Vol.2021-IOT-55 No.2
Vol.2021-SPT-44 No.2
2021/9/6

エリアラベルを用いた分散型SDN・HARPモデルでのルーティング手法の提案

津田 英明[2,a)]　今泉 貴史[1,b)]

概要：　SDN を実現するプロトコルの 1 つである OpenFlow にはスケーラビリティの課題があり，我々の研究室ではこの課題を解決するため HARP モデルを提案している．HARP モデルはネットワークをエリアに分割し，エリア内を制御するエリアコントローラとエリアコントローラを制御するメインコントローラで役割を分散して制御することでスケーラビリティの課題を克服している．HARP モデルの課題として，エリア間のリンク状態に変更が生じた場合にエリアコントローラに負荷がかかったり，同一エリアを複数回通るルーティングは行えない点がある．

本研究では，HARP モデルで課題となっている，トポロジ変化への対応とエリアを複数回通過するフローを実現するため，MPLS を用いてフレームを宛先エリアごとにラベリングしてルーティングする手法を提案する．提案手法により，HARP モデルのスケーラビリティ向上とより柔軟なネットワーク制御が行えるようになる．

1. はじめに

SDN (Software-Defined Network) はパケットやフレームの転送処理をソフトウェアで制御することにより，ネットワークを動的に設計・構築・運用する技術及び概念である．サーバの仮想化技術の普及によって急速に変化するネットワーク環境をプログラムで柔軟に制御することができ，データセンタでの利用が進んでいるほか，第 5 世代移動通信ネットワークでは重要な要素技術として考えられている [1].

SDN を実現するプロトコルの 1 つに OpenFlow がある．ネットワークを転送機能 (スイッチ) と制御機能 (コントローラ) に分離し，ソフトウェアで制御する．各フレーム・パケットのアドレス，ポート番号，VLAN タグ，MPLS ラベルなどの特徴をフローとしてまとめて扱い，フロー単位で転送処理を設定することでネットワークを構築する [2]. OpenFlow は ONF (Open Networking Foundation) によって標準化されており，現在バージョン 1.5 までリリースされている [3]．しかし，既存のネットワークから OpenFlow への移行は進んでいない．進んでいない要因の 1 つに，OpenFlow にスケーラビリティの問題があることが挙げられている [4].

OpenFlow のスケーラビリティの問題として，

- 処理が集中することによるコントローラの負荷の増加
- ネットワークの拡大によってスイッチの保持する情報量の増加

が挙げられる．これらの問題を解消するため，いくつかの分散型 SDN アーキテクチャが提案されている [5], [6], [7].

HyperFlow[5] は，複数のコントローラでネットワークを管理し，各コントローラはネットワークを構成するスイッチの一部を制御する．これによりコントローラの負荷が分散される．コントローラ間でネットワーク全体の情報を同期することで論理的に集中した制御を行うことができる．しかし，各コントローラはネットワーク全体の情報からスイッチを制御するため，スイッチの情報量は考慮されていない．また，コントローラ間の情報の同期には遅延が発生するため，ネットワークの拡張には制限がある．

Kandoo[6] も，複数のコントローラでネットワークを管理するため，コントローラの負荷は分散される．Kandoo のコントローラは，HyperFlow と違い，コントローラを管理するコントローラ（ルートコントローラ）だけがネットワーク全体の情報を持つ．それ以外のコントローラ（ローカルコントローラ）はネットワークの一部の情報だけを持つため，論理的に分散している．自身のみでルートを構築

[1] 千葉大学統合情報センター
Institute of Management and Information Technologies, Chiba University
[2] 千葉大学大学院融合理工学府
Graduate School of Science and Engineering, Chiba University
[a)] hidetsuda@chiba-u.jp
[b)] imaizumi_takashi@faculty.chiba-u.jp

できない場合はルートコントローラが行う．このときローカルコントローラはスイッチのプロキシのようになる．

HARP モデル [7] は，Kandoo と同様にネットワークを論理的に分散させて管理している．HARP モデルではネットワークをエリアという単位に区切り，エリアを管理するエリアコントローラとエリアコントローラを管理するメインコントローラで処理を分散させている．しかし，トポロジの変化によってコントローラの負荷が増大する問題や，エリアを複数回通過するフローの実現に課題がある．トポロジ変化によって負荷が集中すれば，スケーラビリティに影響を及ぼす可能性があり，また，エリアを複数回通過できないとサービスチェイニングが行えず，SDN が目指すネットワークの柔軟な制御に対応できなくなる．

本論文では，HARP モデルで課題となっているエリアコントローラの負荷を軽減しエリアを複数回通過するフローの実現し，HARP モデルのスケーラビリティ向上とより柔軟なネットワーク制御を行うことができるようにする．

2. HARP モデル

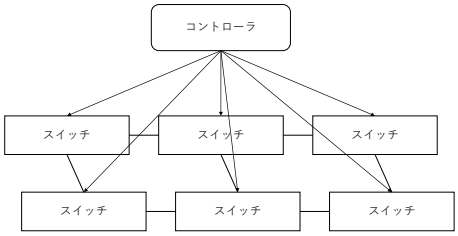
2.1 HARP モデルの概要

HARP モデル [7] は OpenFlow のスケーラビリティ問題に対処するために考案された階層型 SDN アーキテクチャである．HARP モデルはネットワークをエリアという単位に分割し，エリアを管理するエリアコントローラとエリアコントローラを管理するメインコントローラで分散制御する．

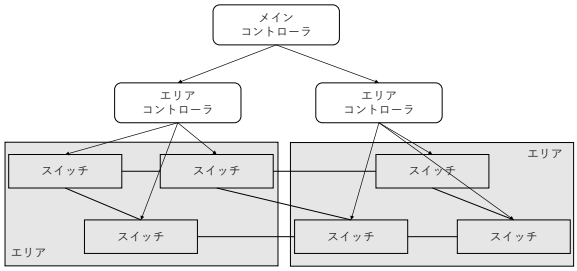
エリアコントローラは管理するエリアのネットワークトポロジとノードの情報（MAC アドレス，IP アドレスなど）を保持しており，エリア内のルーティングについて責務を負う．エリア外のノードへ転送する場合は，メインコントローラに問い合わせ，返答からエリア内部のルーティング（と，隣接エリアへのフレームの送出）を行う．エリア内のトポロジ検出には LLDP (Link Layer Discovery Protocol) を用いる．管理する各スイッチの全ポートから LLDP パケットを送信し，パケットを受け取るとエリアコントローラが内容を解析してトポロジを検出する．エリア外から入ってくる LLDP についてはメインコントローラへ問い合わせを行うことで，メインコントローラがエリア間のトポロジを把握することができる．

メインコントローラはエリア間の接続と全ノードの所属エリアの情報をエリアコントローラからのメッセージから受け取り，エリアをまたぐルーティングについてエリアコントローラに命令を送信することで実現する．

メイン・エリアコントローラ間でのメッセージのやり取りは OpenFlow プロトコルで行う．メインコントローラは通常の OpenFlow スイッチを制御するように，OpenFlow プロトコルを用いてエリアコントローラを制御する．このためメインコントローラからはエリアコントローラは巨大な OpenFlow スイッチのように振舞っているように見える．OpenFlow を用いることにより，メインコントローラ・エリアコントローラ間でやり取りするための API などを新たに実装する必要がなくなる．また，メインコントローラは従来の OpenFlow コントローラにメインコントローラに必要な部分を追加するだけで実装が行える．さらに，プロトコルを OpenFlow で統一したことにより，エリアコントローラの管理するエリアを分割して，サブエリアコントローラを置くことができる．このエリアコントローラとサブエリアコントローラは基本的機能は同じであるので，少し改修するだけで多階層化が行える．



(a) OpenFlow



(b) HARP モデル

図 1: 単一なコントローラによる OpenFlow と HARP モデルの比較

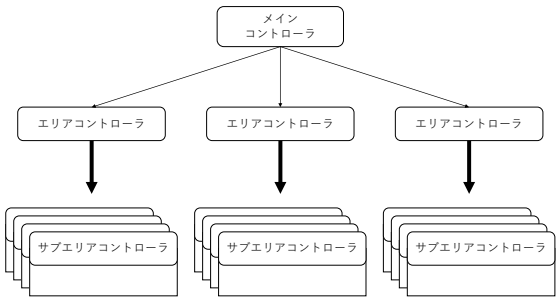


図 2: 多階層の HARP モデルの概念図

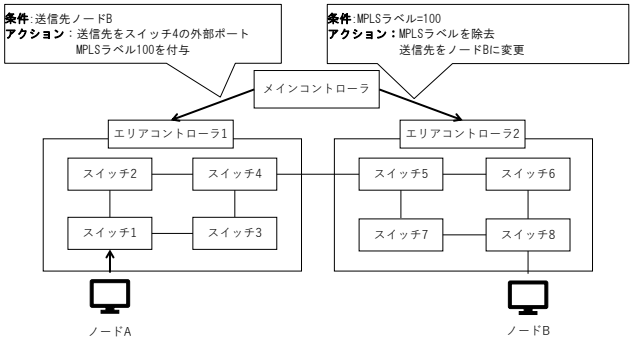


図 3: メインコントローラからエリアコントローラへのメッセージ

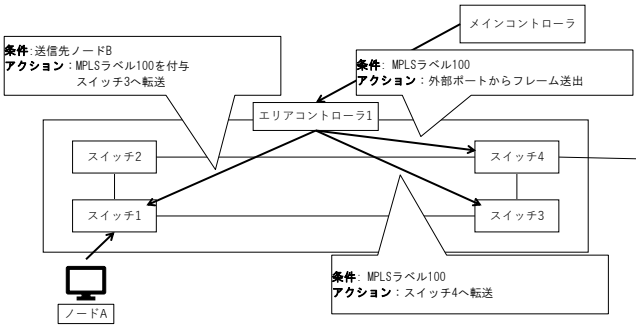


図 4: エリアコントローラからスイッチへのメッセージ

2.2 エリアをまたぐルーティング

図 3 のネットワークでノード A からノード B へ通信を行う場合を考える.

(1) ノード A からのフレームがスイッチ 1 に到達すると，スイッチ 1 はエリアコントローラ 1 へ問い合わせる.

(2) エリアコントローラ 1 はノード B へのルート情報を保持していないため，メインコントローラへ問い合わせる.

(3) メインコントローラは宛先のエリアに基づき，「MPLS ラベルで 100 を設定しスイッチ 4 の外部ポートを宛先にする」ようにエリアコントローラ 1 へ命じる．また，エリアコントローラ 2 へ「MPLS ラベルを除去し，ノード B を宛先にする」ように命じる.

(4) メッセージを受け取ったエリアコントローラ 1 は，メッセージの内容から外部ポートまでのルートを算出し，フレームを転送する各スイッチへメッセージを送信する．エリアコントローラ 2 も同様にスイッチへメッセージを送信する.

(5) スイッチの設定が変更され，フレームはスイッチ 1 からスイッチ 4 の外部ポートまで転送される.

(6) スイッチ 4 の外部ポートから転送されたフレームは，スイッチ 5 へ到達すると MPLS ラベルを除去され，宛先をノード B へ変更され通常通りルーティングが行われる.

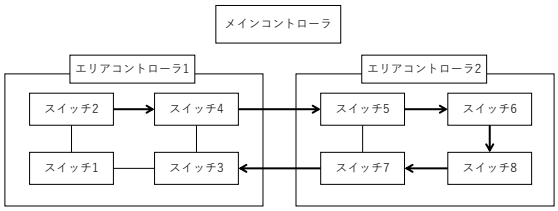


図 5: エリアを複数回通過するフローの例

2.3 HARP モデルの課題

2.3.1 エリアコントローラの負荷

隣接エリアとの接続が不通になるなどでエリア間のトポロジ状態に変化が生じた場合，隣接エリアへ向かうフローは経路に変更が必要になる．変更を行うために変更が必要なフローを削除しなければならず，削除が必要なフローを求める必要がある．また，削除したフローが通信の途中であればすぐにエリアコントローラへ問い合わせが発生し，エリアコントローラは問い合わせに対応しなければならない．さらに，通信が途中であるフローは複数存在することが考えられ，エリアコントローラへ同時に問い合わせが発生することが考えられる．これらの動作によりエリアコントローラへの負荷が増大し，エリアコントローラのスケーラビリティに影響が及ぶ.

2.3.2 エリアを複数回通過するルーティング

従来の OpenFlow モデルでは問題にならないが，HARP モデルではエリアを複数回通過するフローを適用するとループが生じる.

HARP モデルではネットワークをエリアで分割し，エリアの外へ向かうフレームは外へ送出された時点で目的地へ到達したとみなす．また，エリアの外から受け取ったフレームはノードから発出されたフレームと同じ扱いとなる．エリアを複数回通過する場合，エリアへの何回目の訪問なのかを区別する必要があるが，HARP モデルのエリア外のフレームに対する扱い方から区別することができず，ループが生じることとなる.

3. 提案手法

2.3 で示した HARP モデルの課題を解決するために，本論文では隣接エリアの把握手法とエリア外へのフレーム転送手法について新たに提案する．提案手法により，エリアコントローラがメッセージを送信する回数を減らし，またフレームがエリアへの何回目かの訪問なのかを区別する.

3.1 隣接エリアの把握

既存手法ではネットワークのトポロジの検出には LLDP

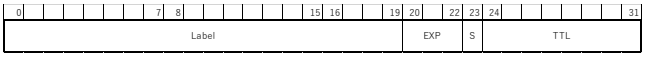


図 6: MPLS のフレームフォーマット

8Octet	6Octet	6Octet	2Octet	4Octet		4Octet
プリアンブル	宛先MACアドレス	送信元MACアドレス	タイプ	宛先エリアを示すMPLSラベル	ペイロード	FCS

図 7: 宛先エリアの MPLS が付与されたフレームの例

を用いる．エリア外からの LLDP についてはエリアコントローラはメインコントローラへ問い合わせを行うことで，メインコントローラだけがエリア間のトポロジを知っていることになっている．

提案手法では，各エリアコントローラは LLDP パケットに自エリアの番号を割り当てた VLAN タグをフレームにつけて管理する各スイッチの全ポートから送信する．エリア外から到着した LLDP パケットについて，エリアコントローラは VLAN タグの情報を見ることで，隣接しているエリアの番号とエリアと接続しているスイッチのポートを把握することができる．ここで得た情報から，エリアコントローラは隣接エリアまでのルーティングを，メインコントローラへ問い合わせることなくプロアクティブに設定することが可能となる．エリア内のルーティングは各エリア内で自由に決めることができる．

エリア外からの LLDP パケットは従来通りメインコントローラへ伝達することで，メインコントローラもエリア間の接続を把握することができる．

3.2 エリア外へのフレーム転送

既存の手法ではエリア外へ転送するフレームについて宛先をノード単位で管理し，エリアの概念は用いていなかった．本手法では，フレームに宛先エリアとネクストホップエリアの情報を付与してルーティングを行う．情報の付与は MPLS(Multi Protocol Label Switching)[8], [9] で行う．MPLS を用いてエリア単位でルーティングを行って宛先エリアまで転送し，目的のエリアに到達するとに MPLS が外される．宛先エリアでは通常のルーティングを行う．

3.2.1 宛先エリアのラベル付与

エリア外が宛先となっているフレームがエリアコントローラへ到達すると，エリアコントローラは宛先情報をメインコントローラに問い合わせる．メインコントローラは宛先情報から宛先エリアを決定し，エリアコントローラに対して「MPLS で宛先エリア番号を付与」する FlowMod メッセージを送信する．FlowMod メッセージを受け取ったエリアコントローラは，フレームを問い合わせたスイッチに対して，「MPLS で宛先エリア番号を付与」する FlowMod メッセージを送信する．これ以降スイッチはフレームに対して宛先エリアラベルを付与する．

宛先エリアのラベルは宛先エリアに到達すると外される．すなわちエリアの番号と宛先エリアとして付与されているラベルの番号が一致していればラベルが除去される．なお，宛先エリアを複数回通過する場合，宛先エリアに到達してもラベルは外さずに，宛先エリアラベルと他の条件にマッチするフローエントリによって処理される．

宛先エリアラベルが付与されたフレームを図 7 に示す．

「MPLS で宛先エリア番号を付与」するフローエントリはエントリのタイムアウトを設定することができる．タイムアウト時間には idle_timeout と hard_timeout がある．idle_timeout を設定すると，フローにマッチするフレームが来ない時間が idle_timeout 時間を超えるとフローエントリが削除される．hard_timeout を設定すると，hard_timeout を超えたフローエントリが削除される．エリアコントローラでは hard_timeout を設定することができ，タイムアウトさせることによって宛先エリアが変更になっていたときにフローエントリを更新できる．スイッチでは idle_timeout と hard_timeout を設定することができ，タイムアウトさせることによってスイッチのフローテーブルの圧迫を防ぐことができる．ただし，スイッチのフローエントリのタイムアウトよりも前にエリアコントローラのフローエントリがタイムアウトし削除されないように hard_timeout を設定する必要がある．

3.2.2 ネクストホップエリアのラベル付与

宛先エリアが隣接エリアの場合，隣接エリアまではプロアクティブにルーティングの設定が可能である．しかし，宛先エリアが隣接エリアではない場合や隣接エリアへ直接転送せずに迂回する場合，宛先エリアへ転送するため次に転送すべき隣接エリアを知る必要がある．そのため，提案手法では宛先ラベルが付与されているフレームに対して次に転送すべき隣接エリアを示したラベル追加で付与する．本論文では次に転送すべき隣接エリアをネクストホップエリアと呼ぶ．

エリアコントローラは未知の宛先エリアの情報を受け取るとメインコントローラへ問い合わせる．メインコントローラはエリア間のトポロジーからルートを算出してネクストホップエリアを割り出し，エリアコントローラに対して「MPLS でネクストホップエリアの番号を付与」する FlowMod メッセージを送信する．FlowMod メッセージを受け取ったエリアコントローラは，スイッチに対して「MPLS でネクストホップエリアの番号を付与」する FlowMod メッセージを送信する．これ以降スイッチはフレームに対してネクストホップエリアラベルを付与する．

このやりとりは，宛先エリアのラベル付与と同時に行うことが可能である．メインコントローラは「MPLS で宛先エリアのラベルを付与し，MPLS でネクストホップエリアのラベルを付与」という FlowMod メッセージをエリアコントローラに対して送信することで，メイン・エリアコントローラ間のメッセージのやりとりの回数を減らすことが

8Octet	6Octet	6Octet	2Octet	4Octet	4Octet		4Octet
プリアンブル	宛先MACアドレス	送信元MACアドレス	タイプ	ネクストホップエリアを示すMPLSラベル	宛先エリアを示すMPLSラベル	ペイロード	FCS

図 8: 宛先エリアとネクストホップエリアの MPLS が付与されたフレームの例

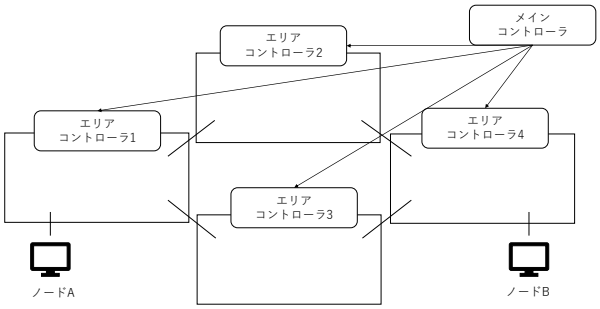


図 9: エリアが 4 つあるネットワーク

できる．

ネクストホップエリアのラベルはネクストホップエリアに到達すると外される．すなわちネクストホップエリアの番号とネクストホップエリアラベルの番号が一致していれば外される．その後，宛先エリアが隣接エリアではない場合は再度ネクストホップエリアラベルが付与される．宛先エリアが隣接エリアであればフレームには変更を加えず事前のルーティング設定に従って転送される．

ネクストホップエリアラベルが付与されたフレームを図 8 に示す．

宛先エリアラベルの付与と同様に，「MPLS でネクストホップエリアの番号を付与」するフローエントリはエントリのタイムアウトを設定することができる．エリアコントローラでタイムアウトさせることによって，ネクストホップエリアが変更になっていたときにフローエントリを更新できる．また，スイッチでタイムアウトさせることによってスイッチのフローテーブルの圧迫を防ぐことができる．ただし，スイッチのフローエントリのタイムアウトよりも前にエリアコントローラのフローエントリがタイムアウトしないように hard_timeout を設定する必要がある．

3.2.3 提案手法の動作例

図 9 のネットワークについて，ノード A からノード B へのフレームの転送を考える．なお，エリアの番号はエリアコントローラの番号と同じとし，各エリアでは隣接エリアまでのルーティングはすでに設定されているものとする．

(1) ノード A からフレームを受け取ったスイッチは宛先のノード B の情報を持たないのでエリアコントローラへ問いあわせる．
(2) エリアコントローラはエリア外にあるノード B についての情報を持っておらず転送できないため，エリアコントローラはメインコントローラへ問い合わせる．

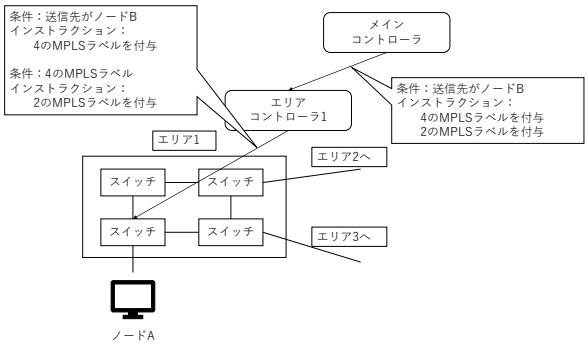


図 10: メインコントローラ・エリアコントローラが送信するメッセージ

(3) メインコントローラはノード B が所属するエリアの情報と，エリア間のトポロジから，「送信先がノード B にマッチする場合，4 の MPLS ラベルを付与し，2 の MPLS ラベルを付与する」という FlowMod メッセージをエリアコントローラ 1 へ送る．
(4) メッセージを受け取ったエリアコントローラは，メッセージを分解し，「送信先がノード B にマッチする場合，4 の MPLS ラベルを付与する」，「4 の MPLS ラベルにマッチする場合，2 の MPLS ラベルを付与する」という 2 つのメッセージに分解してパケットインを起こしたスイッチに送る．
(5) 2 の MPLS ラベルが付与されていることから，エリアコントローラ 2 が管理するエリアへフレームを転送する．
(6) エリア 2 へ到達したフレームは，エリア番号 2 の MPLS ラベルを外され，4 の MPLS ラベルからエリアコントローラ 4 が管理するエリアへ転送される．
(7) エリア 4 へ到達したフレームは，4 の MPLS ラベルを外され，従来通りのフレームの転送が行われる．

宛先から送信元へのフロー（つまり逆向きのフロー）を設定する場合，手順 3 でメインコントローラが「送信先がノード A にマッチする場合，1 の MPLS ラベルを付与し，2 の MPLS ラベルを付与する」という FlowMod メッセージをエリアコントローラ 4 に送信することで設定可能である．

また，エリアコントローラ 1，2 の間が不通となった場合，メインコントローラは「4 の MPLS ラベルがある場合，3 の MPLS ラベルを付与する」というメッセージをエリアコントローラ 1 へ送信する．メッセージを受け取ったエリアコントローラは，フローの起点となるノード A が接続しているスイッチに受け取ったメッセージを送信する．これにより，迂回路を設定することができる．

4. 考察

4.1 エリアコントローラの負荷について

従来手法ではエリア外へのルーティングは，フレームが転送される全てのスイッチに対して宛先ごとにフローエントリを設定していた．エリア間の接続断などによりトポロジが変化した場合，トポロジが変化した接続を通過する全てのフローについてのフローエントリを削除しなければならない．エリアコントローラがスイッチに対して削除するフローエントリの総数 f は，トポロジが変化した接続を通過するフローの数 t とフローが通過するスイッチ s より $f = ts$ となる．

一方，提案手法ではノードや隣接エリアからフレームを受け取るスイッチでの削除だけで済むので，最大 $f = t$ のフローエントリ数となる．特に，影響がある宛先エリアが複数のフローで一致する場合，変更が必要なフローエントリの総数は，影響のある宛先エリアの数 a より $f = a$ となる．

また，エリア間のトポロジの変化の影響を受けるフローが通信の途中であった場合，エリアコントローラはメインコントローラへ問い合わせる．従来手法ではメインコントローラの返答内容（「宛先がノード○にマッチする場合，ポート○から転送」といった内容）から経路を計算して，各スイッチに対して適切な FlowMod メッセージを送信する．一方，本提案手法では隣接エリアまでの経路は事前に計算してあるため，エリア間トポロジの変更の際には経路計算を行わず，メインコントローラからの返答内容（「MPLS ラベルが○にマッチする場合，MPLS ラベル○を付与」といった内容）を変更せずにスイッチへ FlowMod メッセージを送信できる．

4.2 同一エリアを複数回通過するフローについて

先行研究では，該当フレームがエリア外へ行くのか，エリア内で処理をするべきなのかが区別できず，ループが起こってしまっていた．本提案手法により，エリア外へ向かうフレームは宛先エリアとして自エリア以外の MPLS ラベルが付与され，エリア内で処理するべきフレームは宛先エリアとして自エリアの MPLS ラベルが付与されるようになった．エリア内で処理するべきフレームについては，「自エリアの MPLS ラベルが付与されている場合，○○をする」という内容のフローエントリを追加することで実現が可能となる．

4.3 制限

本提案手法では 1 つのフレームに対して複数回のマッチを行えることが前提となっている．複数回のマッチを行うためには OpenFlow のバージョンが 1.1 以上である必要があり，スイッチ・コントローラはともに OpenFlow のバージョン 1.1 以上に対応していなければならない．

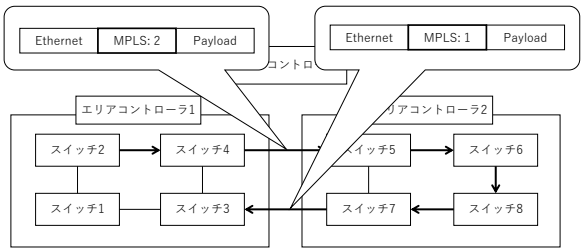


図 11: エリアを複数回通過する場合のフレームの様子

本提案手法では MPLS ラベルが最大 2 つ付与される．MPLS ラベルは 1 つあたり 4 オクテットの長さがあり，ラベルが最大まで付与されるとフレームの長さが 8 オクテット分増える．8 オクテット増えることによって MTU を超えるようなフレームが生成されることが考えられ，フレームを転送できなくなる恐れがある．あらかじめ 8 オクテット分を見込んで MTU を設定する，フレームの分割を行うなどの対応が必要となる．特に，IPv6 では経路の途中でパケットの分割が行われないため，8 オクテット分を見込んだ MTU の設定が必要である．

4.4 関連研究との比較

Fabric

Casado ら [10] は Fabric という SDN で MPLS を用いる方法を提案している．Fabric にはパケットの送信元と送信先となるホスト，受信側と送信側の両方の役割を果たすエッジスイッチ，そしてエッジ間を結ぶコアファブリックの 3 種類のコンポーネントがある．エッジスイッチとコアファブリックのコントロールプレーンは分離される．エッジスイッチではフィルタリングなどのネットワークポリシーが適用され，コアファブリックではフォワーディングの機能に特化している．コアファブリックでは転送に MPLS を用い，従来の LSP でパスを設定する方法が提案されている．

Fabric は HARP モデルと違いエッジとコアファブリックに分離することでネットワークがシンプルに構成されている．一方で，エッジとコアファブリックの分離によりエッジとコアファブリックでの一貫性を保つための仕組みが必要となる．また， Fabric では MPLS の経路については従来の LSP を用いており実装が容易である一方で，同一 IP アドレスが存在するようなネットワークにおいて柔軟に制御することは難しく，アドレス変換を行う必要などが生じる．

5. おわりに

本論文では，HARP モデルで課題となっていたエリアコントローラの負荷を軽減しエリアを複数回通過するフローの実現する手法を提案した．フレームの宛先となるエリアと次に転送すべきエリアの情報をラベルを用いて付与することでルーティングを行い，エリアコントローラのメッセージの数を減らし，エリアを複数回通過するフローが実現できるようになった．更にメインコントローラとエリアコントローラのメッセージを一部統一することで，エリアコントローラが再度の経路計算を行わず，メッセージの数を減らすことでエリアコントローラへの負荷を軽減した．

今後の課題として，本提案手法が多階層のコントロールプレーンにおいても作用するように検討する必要がある．多階層にした場合，メインコントローラからサブエリアコントローラに対して直接サブエリア番号を伝えることができない．サブエリア番号が重複してしまうと正しくルーティングが行えなくなるため，サブエリア番号が重複しないための方法を検討する必要がある．

参考文献

[1] 服部　武，藤岡雅宣：5G 教科書：LTE/IoT から 5G まで，インプレス (2018).
[2] あきみち，宮永直樹，岩田　淳：マスタリング TCP/IP OpenFlow 編，オーム社 (2013).
[3] Open Networking Foundation: OpenFlow Switch Specification Version 1.5.1(Protocol version 0x06) (2015).
[4] Zhang, Y., Cui, L., Wang, W. and Zhang, Y.: A survey on software defined networking with multiple controllers, *Journal of Network and Computer Applications*, Vol. 103, pp. 101–118 (online), DOI: https://doi.org/10.1016/j.jnca.2017.11.015 (2018).
[5] Tootoonchian, A. and Ganjali, Y.: Hyperflow: a distributed control plane for openflow, *Proceedings of the 2010 internet network . . .*, pp. 3–3 (2010).
[6] Hassas Yeganeh, S. and Ganjali, Y.: Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications, p. 19 (online), DOI: 10.1145/2342441.2342446 (2012).
[7] 宮本翔平，今泉貴史：ネットワークの分割を用いたコントロールプレーン分離型 OpenFlow モデルの提案, *FIT2013*, pp. 303–308 (2013).
[8] Viswanathan, A., Rosen, E. C. and Callon, R.: Multiprotocol Label Switching Architecture, RFC 3031 (2001).
[9] Tappan, D., Rekhter, Y., Conta, A., Fedorkow, G., Rosen, E. C., Farinacci, D. and Li, T.: MPLS Label Stack Encoding, RFC 3032 (2001).
[10] Casado, M., Koponen, T., Shenker, S. and Tootoonchian, A.: Fabric: a retrospective on evolving SDN, *Proceedings of the first workshop on Hot topics in software defined networks*, pp. 85–90 (2012).