

OpenFlow によるデータセンタネットワークのフロー到達性検証の 一考察

牛 躍川[†] 橋本 直樹[†] 廣津 登志夫[‡]

法政大学大学院情報科学研究科[†] 法政大学情報科学部[‡]

1. はじめに

サーバの仮想化やクラウドの急速な進化などにより、統合管理や運用自動化が進む中で、トラフィックの急速な増減に伴う通信トポロジの安全な変更を自動化することが大きな課題となっている。柔軟にネットワークを運用する技術として、OpenFlow[1] が注目されている。OpenFlow はネットワーク制御をソフトウェアで行うことによってネットワーク全体の挙動を集中管理することを可能にしており、データセンタネットワークにおける活用が広まっている。このような環境では、複数のテナントネットワークが集約されているため、通信の到達性と分離性が正しく設定されているかどうかの検証を行う必要がある。しかし、OpenFlow プロトコルには開通時の到達性を確認するための機能がなく、設定した経路上で正しく通信が行われることを確認することが出来ない。そこで、本研究では OpenFlow ネットワークにおいて IP 通信を行う際の異常性の検出と到達性の確認を自動で行う枠組みの構築を目指して、OpenFlow のフローエントリを用いた基本的な到達性検証を試みた。ここでは、OpenFlow スイッチが持つフローテーブルを用いることによって各接続の到達性と正当性を自動的に検証する。

2. OpenFlow

OpenFlow は ONF が標準化を行う集中制御型のネットワークアーキテクチャであり、コントロールプレーンを OpenFlow コントローラ、データプレーンを OpenFlow スイッチとし、コントローラとスイッチ間の通信を OpenFlow プロトコルで定義している。OpenFlow では、マッチング条件とマッチ時に実施すべき処理の定義をまとめたフローエントリに従ってパケットを処理する。

複数のフローエントリの集合をフローテーブルと呼び、OpenFlow スイッチは複数のフローテーブルを持つ。これらにより、OpenFlow はフローを特定する条件とフローに合致するパケットへの処理内容を柔軟に設定することができる。

3. 関連研究

文献[2]では、マルチテナント型のデータセンタネットワークにおけるネットワーク機器設定の事前検証を自動化する手法が提案されている。機器設定の適用前と適用後における設定項目間の依存関係を検証ルールに基づいてグラフ形式で抽象化し、それによってテナント間および各テナント内のノードの接続ミスがないことの検証を行っている。

4. 物理ネットワークのトポロジ管理

与えられたトポロジから、その設定した物理ネットワークのノード(スイッチ)間の接続性情報を管理する。それぞれのノードとリンクを頂点と辺として抽象化し、ネットワークをグラフとして表現する。ここで用いるグラフは $G = (V, E)$ として定義し、 $V = (v_1, \dots, v_N)$ はグラフ中の頂点の集合であり、 $E = (e_1, \dots, e_M)$ は辺の集合である。グラフ化したネットワークを接続行列で表現し、これを辿ることで各ノード間の接続性を確認できるようにする。OpenFlow の制御においては、ノードへ接続されている物理ポート番号の情報が必要となるため、辺 e に双方の接続ポート番号の情報を付与する。接続行列上の管理においては、列のノードの接続ポート番号を行列で保持するようにしている。

5. フローの検証

OpenFlow ネットワークでは、フローの経路制御に関する情報はコントローラがフローエントリをスイッチに追加することによって設定される。コントローラから各スイッチにフローエントリが設定されたときに、期待するフローが構成されたネットワーク上で正しく通るか検証する。コントローラが持つエントリ情報を集約して 1 つのテーブルにし、物理ネットワークのトポ

A Reachability Validation Method for a Datacenter Network using OpenFlow

[†]Graduate school of Computer and Information Sciences, Hosei University

[‡]Faculty of Computer and Information Sciences, Hosei University

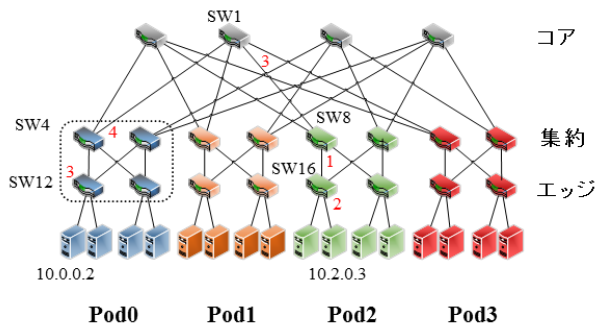


図 1. 検証に用いるネットワークトポロジ

ロジを元に各エントリの出力ポートにつながっているスイッチを取得する。宛先 IP アドレスに対する転送先スイッチとの接続関係を調べ、これをフローの両端のエッジスイッチ間で探索することによって設定したフローの到達性を検証する。

6. Fat-tree における検証例

4 節で述べた提案手法を用いて Fat-tree トポロジを用いた検証の例を示す。まず、トポロジの設定と投入されるフローエントリについて述べ、更に検証例を示す。

6.1 ネットワークトポロジ

本研究ではノード数 16、スイッチ数 20 で構成された Fat-tree トポロジ(図 1)を対象として検証を行う。ここでは、各サーバに 10.0.0.0/8 のプライベート IP アドレスを以下のように割り当てる。Pod に 0 から 3 の Pod 番号 p 、Pod 内のエッジスイッチに 0 または 1 のスイッチ番号 s を割り当て、エッジスイッチごとに左側のノードを 2、右側のノードを 3 とノード番号 n を割り当てたとき、 $10.p.s.n$ をサーバの IP アドレスとして割り当てる。また、スイッチにはコア層の左から順番に 0 から 19 までのスイッチ ID をつける。

6.2 フローエントリの設定

ここでは、通信制御の例として以下のようなルーティングを想定したフローエントリを設定する。このネットワークでは、宛先 IP アドレスに従ってルーティングを行うことによってマルチパスの活用を行う。上向きのルーティングにおいて、エッジ層のスイッチから集約層のスイッチにパケットを送出する際には、宛先 IP アドレスの 3 オクテッド目によって転送先のスイッチを決定し、集約層のスイッチからコア層のスイッチへの送付においては、宛先 IP アドレスの 4 オクテッド目によって転送先を決定する。

6.3 検証例

例として、10.0.0.2 の IP アドレスを持つサーバ

表 1. フローエントリ

SwitchID	Match Field	Action
12	ethType=0x0806, 0x0800 dstIP=10.2.0.3	output:3
4	ethType=0x0806, 0x0800 dstIP=10.2.0.3	output:4
1	ethType=0x0806, 0x0800 dstIP=10.2.0.3	output:3
8	ethType=0x0806, 0x0800 dstIP=10.2.0.3	output:1
16	ethType=0x0806, 0x0800 dstIP=10.2.0.3	output:2

表 2. 接続行列

	SW1	SW4	SW8	SW12	SW16
SW1	×	1	3	0	0
SW4	4	×	0	1	0
SW8	4	0	×	0	1
SW12	0	3	0	×	0
SW16	0	0	1	0	×

から 10.2.0.3 の IP アドレスを持つサーバへのフローを検証する際の手順を以下で説明する。最初に 10.0.0.2 のサーバに接続されているスイッチ 12 のフローエントリ(表 1)を参照し、宛先 IP アドレス 10.2.0.3 へのパケットは 3 番ポートに出力するという情報を得る。次に、物理ネットワークの接続行列(表 2)でスイッチ 12 に関する接続関係を参照することによって 3 番ポートの先にスイッチ 4 が接続されていることがわかる。さらにスイッチ 4 のフローエントリに関しても同じように参照し、スイッチ 4 の出力ポート 4 にスイッチ 1 が接続されているという情報を得る。このような接続行列とフローエントリを参照する手順を宛先のノードが接続されているエッジスイッチが見つかるまで繰り返す。

7. まとめ

本稿では、OpenFlow ネットワークにおけるフロー到達性の検証について述べた。現状では最も基礎的な追跡のみであり、今後、VLAN や複数のテナントなどより複雑な構成の検証に発展させる予定である。

参考文献

- [1] OpenFlow
<https://www.opennetworking.org/sdn-resources/openflow>
- [2] Yosuke Himura, and Yoshiko Yasuda, "Static Validation of Network Device Configurations in Virtualized Multi-tenant Datacenters" IFIP/IEEE IM 2013, May 2013