1Z-05

OpenFlowコントローラの負荷を考慮した同期手法の提案

降簱裕太[†1]　岸知二[‡1]

[†1] 早稲田大学　創造理工学研究科　経営システム工学専攻

1.研究背景と目的

ネットワーク技術の発達に伴い、ネットワークの大規模複雑化が進んでいる。従来のネットワークでは、ネットワークを管理するために個々の機器を設定していた。そのため、近年の大規模複雑化したネットワークに対応するのは難しいという問題点がある。この問題に対して、ソフトウェアでネットワーク制御をするSDN(Software Defined Network)が一つの解決策としてあり、特にSDNの標準の一つであるOpenFlowに注目が集まっている。

OpenFlowではノード間の通信を制御するOpenFlowコントローラ(以下、コントローラ)とパケットの転送を行うOpenFlowスイッチ(以下、スイッチ)に分けて制御を行う。各スイッチは機能をコントローラによって管理される。そのためネットワーク技術者はコントローラによって複数のネットワーク機器を一元管理でき、従来よりも柔軟にネットワークを管理・構成をすることができるようになった。

しかしこうした構成をとることにより、耐故障性に関する新たな問題が生じる。コントローラによる集中管理でネットワークを動的に管理することができる一方で、コントローラが単一障害点となる恐れがある。そのためコントローラの冗長性確保は必要不可欠である[1]。

コントローラの冗長化を考える際に、考慮しなければならないことが二つある。一つは現在スイッチを管理しているコントローラ(以下、運用系コントローラ)からホットスタンバイしているコントローラ(以下、待機系コントローラ)への切り替えを速やかに行うことである。運用系コントローラが故障してもネットワーク内にはパケットが流れ続けているため、早急な切り替えが求められている。もう一つは、コントローラの持つ情報を同期しておくことである。OpenFlowは動的にネットワークを管理できる性質からクラウド環境やデータセンタでの活用が現在の主流である。こういった環境ではパケットのフロー情報は頻繁に変更が起こると想定されるからである。

また、一般には切り替えや同期によって、コントローラ、スイッチ、各回線に負荷がかかる。コントローラの切り替えや同期が行われている間もネットワーク内にはパケットが流れている。そのため、負荷がかかることによってネットワーク全体の性能が低下してしまう恐れがある。本研究ではコントローラの同期の際にかかる負荷を考慮した同期手法を提案する。

2. OpenFlow概要

OpenFlowではコントローラがスイッチの機能を全て決めている。その際、コントローラは以下の三つの情報でスイッチの機能を規定する。一つ目がマッチングルールである。これは受け取ったパケットに対してアクションを起こすか否かを決定するためのものである。マッチングルールとしてスイッチの物理ポート番号や送信元のMACアドレスなどを参照することが可能である。二つ目がアクションである。マッチングルールに合致したパケットが来た際に、どのような処理を行うかが記述されている。行う処理としてはパケットの転送、パケットの中身を書き換え、パケットを廃棄などが可能である。三つ目が統計情報である。受信パケット数や受信バイト数などがここに記述される。これらをまとめたものをフローエントリと呼び、この情報はスイッチのフローテーブルに格納される。スイッチはこのフローエントリにしたがってパケットの処理を行う[2]。

3. 提案手法

本研究では同期にかかる負荷として同期のオーバーヘッド[3]を下げる同期手法を提案する。同期のオーバーヘッドとは、次の同期までの間隔の時間に対する同期にかかった時間の比率のことである。同期のオーバーヘッドを下げるためには同期用の回線を太くすることで転送速度を早くすることで同期の時間を減らす、同期の頻度を減らして同期の間隔を延ばす、同期する

An OpenFlow Controller Synchronization method considering network load

†1 Department of Industrial and Management Syetems Engineering CSE Graduate School, Waseda University

データ量を減らして同期にかかる時間を減らすことなどが挙げられる。コントローラの同期手法として 2 点の拡張を施す。一つは同期をフローエントリの削除をトリガーとして行うことで同期の頻度を下げ、同期するデータを削除されたものに限定することである。もう一つがコントローラ切り替え時にスイッチのフローエントリを参照して同期を行うことである。これによって削除に対する同期の不足を補う。それぞれの詳細は 3.1、3.2 で述べる。

3.1.削除同期

同期するタイミングを特定イベント後に限定することで同期の頻度を下げ、同期のオーバーヘッドを下げる。運用系コントローラと待機系コントローラの同期をスイッチのフローエントリ削除のタイミングで行うことで同期の頻度を下げることができる(図 1)。フローエントリは一般的に一定時間で削除されるので、スイッチの機能を拡張し、スイッチからフローエントリの削除を運用系コントローラに通知したタイミングで同期を行う。またこのとき同期する情報を削除されたものに限定することで同期の際のデータ量も減らすことが可能である。前述の通り、コントローラは新しい情報を獲得したタイミングでスイッチに書き込むことを想定している。そのためフローエントリが削除されたときに削除されたものだけを同期すれば、運用系コントローラが故障しない限り同期は可能である。

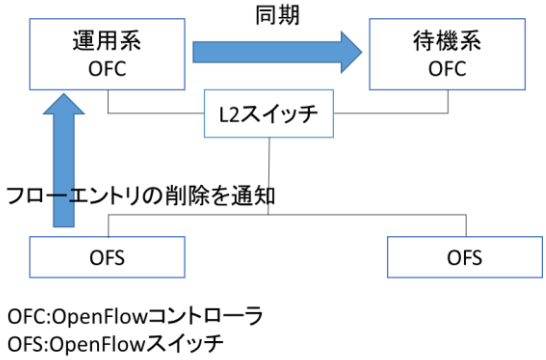

図 1. 削除同期イメージ

3.2.スイッチからの同期

追加されたフローエントリのように削除同期では同期できないフローエントリは運用系コントローラと待機系コントローラで同期がされていない。この場合に運用系コントローラが故障してしまうと待機系コントローラの情報が足りなくなる。そこで、切り替え時に待機系がスイッチから同期されていないフローエントリと同期を取る(図 2)。5.1 とあわせることで同期されていないものがなくなる。同期の際、フローエントリの情報を参照することで、まだ同期されていないフローエントリを見つけ出す。フローエントリを構成する情報の中の統計情報には、フローエントリが出来てからの経過時間が存在する。削除同期の行われた時間と比較することで、同期すべき情報かどうかを判別することが可能である。

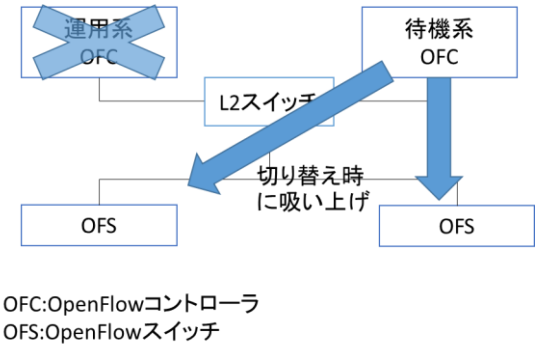

図 2.スイッチからの同期イメージ

3.3.モデル検査による検証

提案手法によって正しく同期できるかどうかモデル検査技術によって確認した。運用系コントローラ、待機系コントローラ、スイッチがそれぞれ相互に接続されているモデルにおいて提案手法によって同期が行えることを確認した。

4. 結論

本研究は OpenFlow コントローラの冗長化に際しての負荷を考慮した同期手法を提案した。同期の負荷を下げる方法として同期の頻度を下げること、同期するデータ量を減らすことの二つの観点から負荷の軽減を行うための手法を提案した。それぞれの観点に OpenFlow の構成に着目した手法をとったことで OpenFlow ならではの負荷を考慮した同期手法である。

今後の課題として負荷を定量的に測定し評価することが必要である。

参考文献

[1] Keisuke Kuroki,Masaki Fukushima Michiaki Hayashi, Nobutaka Matsumoto,"Redundancy Method for Highly Available OpenFlow Controller", International Journal On Advances in Internet Technology, volume 7, numbers 1 and 2, 2014 ,2014-6-30

[2]高宮安仁、鈴木一哉, クラウド時代のネットワーク技術 OpenFlow 実践入門, 技術評論社, 2013

[3]田村芳明, 柳沢佳里, 佐藤 孝治, 盛合 敏, " Kemari:仮想マシン間の同期による耐故障クラスタリング", 情報処理学会論文誌. コンピューティングシステム 3(1), 13-24, 2010-03-16