論 文 の 内 容 の 要 旨

論文題目　大規模ネットワーク管理のための HTTP/SNMP リバースプロキシと管理トラヒックの公平性に関する研究

氏　名　宋　泰永

インターネットの普及と伴ない,大規模化し複雑化した分散型のネットワーク管理を中央集中型の SDN (Software-Defined Networking) や NFV(Network Functions Virtualization)により合理化することをネットワーク事業者が検討するようになった.ネットワーク管理者が直接ネットワーク機器にターミナル接続しコマンドを入力して管理する時代から，中央のコントローラ上の管理アプリケーション(以下では Network Management System あるいは NMS と略称する）がネットワーク内のトラヒックフローの変更や不具合を自動的に探知し，ネットワークリソースとして仮想化されたサーバや firewall を動的に配置して問題解決するといったネットワーク管理の自動化が進んでいる.この結果,中央のコントローラが数万台以上の機器を管理するようになっている.しかし,このような新しい管理に対応していない機器は依然として数多く使われていて,その全てを一気に新しい機器へ置換することは難しい.本研究では既存のネットワーク機器の管理手法である SNMP を大規模かつ中央集中化した SDN や NFV のような新しいネットワーク管理技術と共存させるため,新しいキャッシュ管理手法に基づくプロキシを提案する.

Hamid 等が提案したプロキシ Tambourine は，管理者がインターネットを介して大規模なネットワーク管理する場合に HTTP とネットワーク管理プロトコル間の相互変換を行うプロキシであり，HTTP で送る管理用のリクエストを SNMP のリクエストに変換しネットワーク機器(以下 SNMP 機器と呼ぶ）に転送していた．その際，HTTP リクエストに

対するSNMP機器からの管理情報をOID(Object Identifier)ごとにキャッシュし，頻繁な管理リクエストを処理することに起因するSNMP機器の負荷を下げ，かつ各情報の更新周期に合致した最適なキャッシュのTTL(Time-to-live)を自動学習することに成功していた．しかし，HTTPリクエストを頻繁に値が更新するOIDに集中させることにより，SNMP機器をDDoS (Distributed Denial of Service)攻撃できる脆弱性があった．

このため，本論文では，下記の4要求条件に基づく新たなキャッシュ制御手法を考案している．条件1として，Tambourineは，SNMP機器のCPU使用率の増加分が一定値(*TH*)以下になるよう管理トラヒックを制御する．このためには，管理トラヒックによるSNMP機器のCPU使用率を計算しキャッシュの*TTL*を設定する必要がある．条件2として，同一リクエスト間隔のNMS間で情報取得の公平性を実現しなければならない．NMSからのHTTPリクエストに対しTambourineが得たSNMPレスポンスの数の平均値でその標準偏差を割って相対化した変動係数(*R*)を公平性の評価尺度とした．*R*が小さければ，そのグループ内のNMSは公平であると見なすことができる．条件3として，異なるリクエスト間隔のNMSグループの間でも情報取得の公平性を実現する．前述のように，*R*はNMSからのHTTPリクエスト総数に対しTambourineが得たSNMPレスポンスの数の標準偏差を平均値で相対化したものであるから，*R*が同じであれば，リクエスト間隔に依らず，NMSグループ間の相対的なばらつきが同じであることを意味する．このための評価尺度として上記の*R*を用いることができる．最後に，条件4として，頻繁な問い合わせるNMSに対しては，上記の条件を満足する範囲内で，最大限応じなければならないとした．

そのための評価尺度として，NMSからTambourineへのHTTPリクエストがキャッシュにヒットした場合，SNMP機器から実際に得たOID情報はそれ以前に取得した値であるため，取得ギャップが生ずることに注目し，取得遅れの平均値を鮮度(*Fr*)として定義し評価した．提示手法は，TambourineからSNMP機器へ送るSNMPリクエスト頻度とSNMP機器のCPU使用率間の関係を事前実験により求め，SNMP機器へのSNMP頻度(リクエスト/秒)からSNMP機器のCPU使用率を推定することを基本とする．このための前処理として，PCからAlaxala3640sとCatalyst 2950にSNMPリクエストを任意の一定間隔で送り，各機器の過去1分間のCPU使用率を測定し近似式を求めた．

条件1に対しては，NMSからの問い合わせに対し，SNMP機器のCPU利用率が高い輻輳状態ではキャッシュから値を返し，他の場合はSNMP機器から情報を取得して返すが，その際，TCP Renoに類似したキャッシュの更新間隔(*TTL*)調整手法を提案した．ここで，CPU使用率増加が予め指定した閾値(*TH*)以上になることを輻輳と定義している．また，TCPのREDに類似したリクエストの制御手法を提案し，リクエスト送信頻度の高いNMSほど，CPU使用率の計算の際，輻輳と判断する確率を非線形に上げ，これにより，NMS

間で公平に SNMP 機器からレスポンスを取得でき，条件２と３を満足させることができることを示した．最後に，条件４の評価尺度として，NMS の問い合わせ間隔と取得データの遅延との関係を *Fr* で評価した．

評価は，ns-2 で構成したシミュレーションで実施した．条件１は Reno に類似したキャッシュの制御で，SNMP 機器の CPU 使用率が *TH* として設定した 20%近傍に収束することを示した．条件 2 は，HTTP リクエスト間隔が同じ NMS 間では，RED でキャッシュ制御手法行うことにより，公平に SNMP 機器から SNMP レスポンスを得ることができることを示した．また，条件 3 は NMS からの HTTP リクエスト間隔の最大値より大きい RTO を選択し，本論文で修正を施した RED でキャッシュの制御をすることで，異なるリクエスト間隔の NMS グループ間でも公平性が改善されることを示した．最後の条件４に対しては，提案手法により NMS からの HTTP リクエスト間隔と *Fr* が比例することが分かった．これは短いモニタリングほど短い遅延時間で SNMP 機器からレスポンスを得ていることを意味し，条件 4 を満足していることを意味する．以上の評価実験により，提案手法は条件全てを満足する公平なキャッシュの制御を実現できることを示すことができた．