Vol.2016-IOT-32 No.40
2016/3/4

IP Traceback手法の提案とOpenFlowを用いた実装

矢野 隼太[1,a)]　吉浦 紀晃[1,b)]

概要：IP Traceroute は，目的地へ送信したパケットの経路を把握することが目的である一方で，IP Traceback は，到達したパケットの経路を把握することを目的としている．DDoS 攻撃やなりすまし攻撃に利用されるパケットでは，その送信元アドレスを詐称している場合があり，パケットの送信元アドレスから実際の送信元が把握できるとは限らない．よって，攻撃のパケットが到達した場合，そのパケットの経路を把握することで，攻撃のパケットの送信元を把握することが可能となる．そこで，本稿では，IP Traceback の実現方法を提案し，その実装を Openflow を用いて行う．

Proposal and implementation of IP Traceback method by OpenFlow

Hayata Yano[1,a)]　Yoshiura Noriaki[1,b)]

Abstract: IP traceroute is used to see the route of departing packets, while IP traceback is used to see the route of packets of arriving packets. IP traceback is useful to detect the route of packets of DDoS attack and spoofing attack because the packets fabricate source IP addresses and sources of the packets is not known by the source IP addresses of the packets. Prevention of these kinds of attacks requires the detection of the routes and sources of the attacking packets. This paper proposes the method of IP traceback and implements the method at Openflow controllers.

1. はじめに

IP Traceback とはインターネット通信の送信元を辿る技術である．本来，パケットヘッダの送信元 IP アドレスを確認することで送信元を特定することは容易であるが，サイバー攻撃などの目的で送信元 IP アドレスが詐称されている場合に送信元を特定することは困難である．IP Traceback は，到達したパケットの経路を把握することにより，送信元 IP アドレスが詐称されていても，パケットの送信元を特定する言を目的としている．しかし，IP Traceback は技術面や通信速度の問題などから普及はしていない．

IP Traceroute は，送信するパケットの目的地までの経路を調べることが目的であり，いくつかの方法がある．最も一般的な方法は，送信元から TTL を少なくしたパケットを送信して，そのパケットに対して ICMP メッセージを利用して経路を調べる方法である．これは，traceroute などのコマンドとして実現されている．この IP Traceroute では，経路上のネットワーク機器は何かの情報を保持する必要はなく，ICMP メッセージを送信するだけで済むため技術的な少なく，また，ネットワーク機器へ負荷をかけることも少ない．一方，IP Traceback は，到着したパケットの送信元からの経路を調べることが目的であるため，パケットが通過したことをネットワーク機器が記録しておく必要がある．このため，ネットワーク機器の負荷が大きく，これが普及しない理由の一つである．また，実際に IP Traceback を実現させるためには，ネットワーク機器に IP Traceback の仕組みを導入する必要があるが，これは現実的には難しい．以上が，IP Traceback が普及しない技術的な理由である．

OpenFlow とは SDN(Software Defined Network) を実現するためのプロトコルの一つである．従来，ネットワーク機器を利用する場合，それぞれの機器に静的に設定を行う必要があるが，SDN では，プログラミングにより複数のネットワーク機器への設定を動的に行うことができる．

[1] 埼玉大学工学部情報システム工学科
Departement of Information and Computer Sciences, Saitama University
[a)] hyano@fmx.ics.saitama-u.ac.jp
[b)] yoshiura@fmx.ics.saitama-u.ac.jp

OpenFlow は，OpenFlow コントローラーと OpenFlow スイッチという二つの機械を用いて SDN を実現する．OpenFlow 自体は，このスイッチとコントローラー間での通信のプロトコルを表すが，OpenFlow により作られたネットワークのことも含め OpenFlow と呼ぶ．

OpenFlow では，OpenFlow スイッチがパケットの処理をどのように行うかを示す規則をフローエントリとして保持し，このフローエントリに基づきパケットの転送処理などを行う．このフローエントリは OpenFlow スイッチに予め設定されている場合もあるが，動的にフローエントリを設定することも可能である．

動的にフローエントリが設定される場合には，OpenFlow スイッチにパケットが入ってきたときに，OpenFlow コントローラからフローエントリが提供される．具体的には，OpenFlow スイッチがフローエントリに適さないパケットを取得した場合，そのパケットの情報を OpenFlow コントローラーへ送る．OpenFlow コントローラーは OpenFlow スイッチが取得したパケットの情報から，OpenFlow スイッチに対してパケットに処理方法やフローエントリの追加などを OpenFlow スイッチに命令する．フローエントリの追加などの命令を OpenFlow コントローラから受け取った OpenFlow スイッチは，この命令に基づきパケットを処理する．以後，類似のパケットを受け取った時には，フローエントリに基づき処理を行うため，OpenFlow コントローラへの問い合わせは必要ない．

本稿では，OpenFlow の仕組みを利用して IP Traceback を実現する．具体的には，OpenFlow スイッチに入ったパケットにより，OpenFlow コントローラにパケットの情報が届いたときに，パケットの情報を記録し，その記録された情報により IP Traceback を実現する．この方法であれば，OpenFlow スイッチには IP Traceback のための仕組みを実装する必要はなく，OpenFlow コントローラに IP Traceback のための仕組みを導入するだけで実現可能である．

以後，第 2 章で先行研究について説明し，第 3 章で本稿で提案する方法を説明する．第 4 章で提案方法を実装し，その実験結果について述べる．第 5 章で本稿をまとめる．

2. 先行研究

IP Traceback の先行研究として，3 つの手法を紹介する．

2.1 マーキング方式

マーキング方式では，パケットがルーター (Layer 3 スイッチ) を通過する際に一定の確率でルーターがパケットに自身の情報を埋め込む [2]．パケットの目的地である IT 機器は，送られてきたパケットに埋め込まれた情報を集めることで，パケットの経路と送信元を特定することができる．この手法では余計なパケットを作らないためネットワークに負荷がかからない．また，パケットにはルーターの情報を書き込むスペースは与えられていないため，ID フィールドに書き込んでいる．しかし，この手法では，ID フィールドを書き換えることになるため改竄にあたる可能性がある．また，ID フィールドはフラグメントの再構築をするために使用するため，この手法によりフラグメントされたパケットがそれぞれ別の ID フィールドに書き換えられてしまうと再構築ができなくなってしまう可能性がある．

2.2 ICMP 方式

ICMP 方式では，パケットがルーターを通過する際に一定の確率でルーターがそのパケットと同じ宛先へ ICMP パケットを送信する [1], [4]．この ICMP パケットにはどのパケットに反応して送信されたものかという情報が書かれており，パケットの目的地である IT 機器がこの ICMP パケットを集めることで，送信元を特定することができる．しかし，この手法では余計なパケットを作成し送信するためネットワークに負荷がかかるという問題がある．また，攻撃者が ICMP パケットを偽装し送信した場合，本物を見分ける認証も必要である．

2.3 ロギング方式

ロギング方式では，パケットがルーターを通過する際にルーターがそのパケットの情報を保存する [3]．IP Traceback を行う IT 機器がどのルーターを経由してきたか各ルーターに問い合わせることで攻撃元を特定する．

3. 提案手法

本研究ではロギング方式に着目し，OpenFlow を用いることで OpenFlow コントローラを改良することで実装できる IP Traceback の手法を提案する．また，提案手法を実機での実験により有効性を確認する．

3.1 方針

本研究で提案する手法は以下のとおりである．OpenFlow スイッチをパケットが通過した時，OpenFlow コントローラーがそのパケットの情報と時間から生成したハッシュ値を保存する．パケットの情報そのものではなく，そのハッシュ値を保存するのは，パケットの情報自体が通信の秘密であり，その侵害を極力避けるためである．また，ハッシュ値を保存する理由は，その後，パケットの情報と時間での問い合わせに対して，そのパケット情報と時間から精製したハッシュ値が保存されているか確認し応答するものである．この問い合わせを繰り返すことで最終的な送信元まで問い合わせするというものである．

本稿で提案する手法は OpenFlow スイッチに接続されている機器の MAC アドレスと IP アドレスの組が必要となる．その理由は，パケットが自身のルーターが通ったこと

だけでなく，パケットがどの機器を通ってきたかに応答するためである．

3.2 ハッシュ値を保存するまでの基本設計

初期状態ではフローエントリーを存在させず，すべてのパケットを取得するようにする．スイッチにパケットが到達しパケットの情報をコントローラーへ送られると，コントローラーは以下の情報によりハッシュ値を生成する．【ハッシュ値を生成するための情報のテーブル】生成したハッシュ値は HTTP 通信により他のコンピュータへ送信し保存する．（負荷が重すぎたためコントローラー内に書き込みに変更）ハッシュ値を生成したパケットは宛先に届くように転送を行う．

3.3 MAC アドレスと IP アドレスの組の収集方法

動的と静的に収集するという二つの手法を提案する．

3.3.1 MAC アドレスと IP アドレスの組みを動的に収集する手法

コントローラーがスイッチのポートそれぞれが MAC アドレスと IP アドレスを持つようにふるまうことで MAC アドレスと IP アドレスの組を収集する．具体的にはコントローラーが ARP 処理に応答するようにスイッチに命令することで，コントローラーがスイッチに隣接する機器の ARP テーブルを取得する．ハッシュを取ったパケットはルーティングテーブルに従いパケットヘッダの MAC アドレスを書き換え転送する．ルーティングテーブルについていは静的に設定している．

3.3.2 MAC アドレスと IP アドレスの組を静的に設定する手法

コントローラーにあらかじめ MAC アドレスと IP アドレスが設定されているためそれらを収集する必要はない．コントローラーはスイッチを通るパケットのハッシュ値を生成し，パケットは送信先 MAC アドレスに従い転送する．

3.4 ハッシュ値の生成頻度

スイッチはパケットが分割されていてもそれらすべての情報をコントローラーへ送る．しかし，ハッシュ値を生成する際に使用する UNIX 時間は一秒より短い時間の差を測ることができない．そのため，一秒以内に同じパケットヘッダを持ったパケットが複数通った場合にハッシュ値を重複して作成してしまう．そこで，これを防ぐために生存時間を一秒以上に設定したフローエントリーを使用する．コントローラーはパケットのハッシュ値を取る際に，パケットと同じパケットヘッダをもつパケットは転送処理だけを行うフローエントリーを追加するようにスイッチに命令する．これにより一度ハッシュ値を取ったパケットはフローエントリーが存在する間コントローラーに情報が行くことはないため，ハッシュ値を重複して作成することはない．フローエントリーは作成されてから生存時間以上の時間がたつと消滅する．つまり，フローエントリーの生存時間がそのままハッシュ値の生成頻度となる．

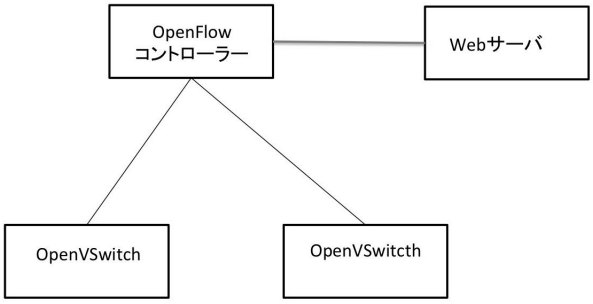

図 1 システム構成図

3.5 IP Traceback の方法

IP Traceback を利用する方法を説明する．IP Traceback を利用する人は，到着したパケットの経路を調べるために Web サーバに問い合わせを行う．Web サーバへの問い合わせの内容は，経路を調べたいパケットの送信元 IP アドレスと送信先 IP アドレスとパケットの到着時刻である．これらの情報を受け取った Web サーバは，ハッシュ値を生成し，Web サーバが保存しているハッシュ値と一致するものを探す．ハッシュ値と一致するものがあれば，問い合わせのあったパケットが管理下のネットワークを通過したこと，つまり，経路であったことを答える．

4. システム構成

提案するシステム構成を説明する．図 1 が基本的な構成である．一般的な OpenFlow によるネットワークに加えて，IP Traceback の機能を提供する Web サーバを用意する．この Web サーバは，IP Traceback の情報，つまりハッシュ値を保持する．また，この Web サーバは，IP Traceback を行いたい人からの問い合わせを受け付ける．IP Traceback のための仕組みを，OpenFlow コントローラーと Web サーバに組み込む必要がある．しかし，OpenFlow スイッチには不要である．

5. 実験

本稿で提案したシステムを実際に構築し，このシステムによる負荷を測定した．実験は，2 つのネットワークを用意した．OpenFlow スイッチが 1 台の場合と 2 台の場合を用意し，さらに，OpenFlow スイッチのポートに MAC アドレスと IP アドレスを対応づけることで，静的に MAC アドレスを収集させる場合に対応させ，ポートに MAC アドレスと IP アドレスを与えないネットワークを用意することで，動的に MAC アドレスを収集する場合に対応させた．図 2〜図 5 がネットワーク図である．

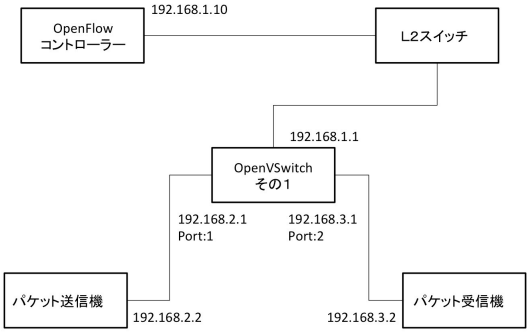

図 2 1 台の OpenFlow スイッチの場合の実験図 (動的に MAC アドレスを収集)

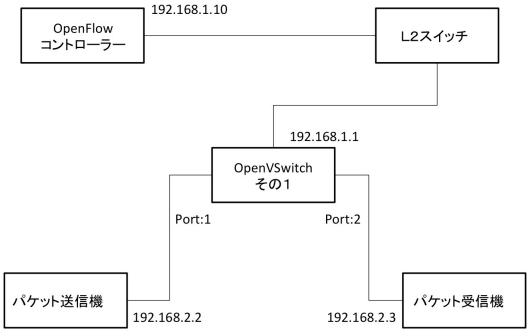

図 3 1 台の OpenFlow スイッチの場合の実験図 (静的に MAC アドレスを収集)

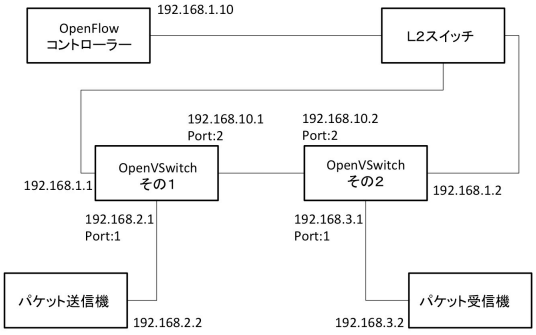

図 4 2 台の OpenFlow スイッチの場合の実験図 (動的に MAC アドレスを収集)

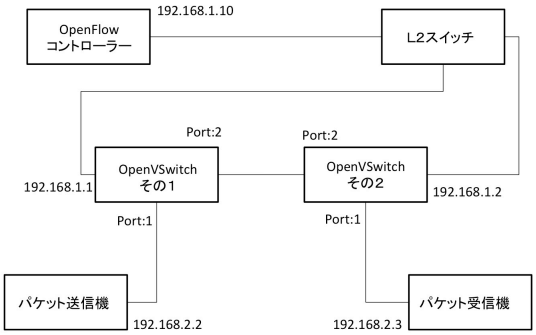

図 5 2 台の OpenFlow スイッチの場合の実験図 (静的に MAC アドレスを収集)

full_no_flow

15.5MB	13Mbps	22.5MB	18.1Mbps	12.286ms
14.9MB	12.5Mbps	20.2MB	16.3Mbps	1.021ms
15.375MB	12.881Mbps	22.05MB	17.921Mbps	6.09395ms
15.4MB	12.9Mbps	22.2MB	18Mbps	6.3105ms

full_1s

301MB	253Mbps	120MB	100Mbps	7.453ms
238MB	196Mbps	117MB	97.8Mbps	0.001ms
278.85MB	233.66Mbps	119.47MB	99.801Mbps	0.71133ms
280MB	235Mbps	120MB	100Mbps	0.001ms

full_60s

300MB	252Mbps	120MB	100Mbps	1.892ms
232MB	194Mbps	117MB	98.2Mbps	0.001ms
275.54MB	230.74Mbps	119.55MB	99.788Mbps	0.38659ms
276MB	231Mbps	120MB	100Mbps	0.001ms

表 1 ハッシュ値を生成し保存する場合 (OpenFlow スイッチ 1 台)

no_write_no_flow

20MB	16.7Mbps	29.9MB	24.5Mbps	5.821ms
19.5MB	16.3Mbps	29.4MB	24.1Mbps	1.053ms
19.757MB	16.559Mbps	29.741MB	24.318Mbps	3.41688ms
19.8MB	16.6Mbps	29.8MB	24.3Mbps	3.288ms

no_write_flow_1s

303MB	254Mbps	120MB	100Mbps	3.789ms
226MB	190Mbps	116MB	97.5Mbps	0.001ms
275.67MB	230.86Mbps	119.29MB	99.749Mbps	0.45873ms
278MB	233Mbps	120MB	100Mbps	0.001ms

no_write_flow_60s

305MB	256Mbps	120MB	100Mbps	1.891ms
252MB	209Mbps	117MB	97.7Mbps	0.001ms
280.08MB	234.58Mbps	119.5MB	99.711Mbps	0.39412ms
281MB	235Mbps	120MB	100Mbps	0.001ms

表 2 ハッシュ値を生成するが保存しない場合 (OpenFlow スイッチ 1 台)

no_cap_no_flow

24.6MB	20.6Mbps	40.1MB	33Mbps	4.011ms
23.3MB	19.5Mbps	36.1MB	29.8Mbps	0.943ms
24.221MB	20.297Mbps	39.828MB	32.651Mbps	2.18262ms
24.3MB	20.3Mbps	39.9MB	32.7Mbps	2.084ms

no_cap_flow_1s

303MB	254Mbps	120MB	100Mbps	1.891ms
228MB	191Mbps	117MB	98Mbps	0.001ms
277.26MB	232.15Mbps	119.54MB	99.812Mbps	0.32241ms
281.5MB	236Mbps	120MB	100Mbps	0.001ms

no_cap_flow_60s

305MB	256Mbps	120MB	100Mbps	1.891ms
229MB	191Mbps	117MB	98.1Mbps	0.001ms
278.72MB	233.47Mbps	119.56MB	99.785Mbps	0.39396ms
280MB	234.5Mbps	120MB	100Mbps	0.001ms

表 3 ハッシュ値を生成しない場合 (OpenFlow スイッチ 1 台)

simple_back_no_flow

44.5MB	37.3Mbps	95.2MB	79.3Mbps	11.036ms
40.4MB	33.9Mbps	87.1MB	72.4Mbps	0.916ms
42.685MB	35.779Mbps	94.16MB	78.2Mbps	2.59579ms
42.65MB	35.8Mbps	94.85MB	78.95Mbps	0.954ms

simple_back_flow_1s

301MB	253Mbps	120MB	100Mbps	1.891ms
238MB	195Mbps	118MB	98.7Mbps	0.001ms
276.89MB	231.99Mbps	119.55MB	99.841Mbps	0.36751ms
279.5MB	234Mbps	120MB	100Mbps	0.001ms

simple_back_flow_60s

301MB	253Mbps	120MB	100Mbps	1.891ms
221MB	185Mbps	117MB	97.8Mbps	0.001ms
274.38MB	229.86Mbps	119.49MB	99.74Mbps	0.38764ms
276.5MB	232Mbps	120MB	100Mbps	0.001ms

表 4 静的に IP アドレスと MAC アドレスの組みを設定し，ハッシュ値を生成し保存する場合 (OpenFlow スイッチ 1 台)

no_back_no_flow

80.3MB	67.2Mbps	120MB	100Mbps	1.891ms
71.9MB	60.3Mbps	117MB	98Mbps	0.001ms
76.492MB	64.099Mbps	119.5MB	99.715Mbps	0.67882ms
76.55MB	64.2Mbps	120MB	100Mbps	0.917ms

no_back_flow_1s

298MB	250Mbps	120MB	100Mbps	1.891ms
240MB	201Mbps	117MB	98.1Mbps	0.001ms
278.2MB	233.09Mbps	119.38MB	99.7Mbps	0.36445ms
280.5MB	235.5Mbps	120MB	100Mbps	0.001ms

no_back_flow_60s

303MB	254Mbps	120MB	100Mbps	1.891ms
235MB	197Mbps	118MB	98.6Mbps	0.001ms
277.31MB	232.2Mbps	119.56MB	99.853Mbps	0.34543ms
278MB	233Mbps	120MB	100Mbps	0.001ms

表 5 静的に IP アドレスと MAC アドレスの組みを設定し，ハッシュ値を生成し保存する場合 (OpenFlow スイッチ 1 台)

full_no_flow

7.75MB	6.5Mbps	11.8MB	9.12Mbps	20.888ms
7.52MB	6.29Mbps	8.98MB	7.05Mbps	1.291ms
7.6598MB	6.4178Mbps	11.2578MB	8.7091Mbps	8.83345ms
7.665MB	6.42Mbps	11.5MB	8.93Mbps	8.0965ms

full_1s

253MB	212Mbps	120MB	101Mbps	39.813ms
237MB	198Mbps	117MB	97.7Mbps	0.001ms
245.68MB	205.96Mbps	119.05MB	99.713Mbps	1.55693ms
246MB	206Mbps	119MB	100Mbps	0.0015ms

full_60s

257MB	216Mbps	120MB	101Mbps	1.891ms
242MB	203Mbps	118MB	98.6Mbps	0.001ms
249.18MB	208.95Mbps	119.53MB	99.9Mbps	0.36921ms
250MB	209Mbps	120MB	100Mbps	0.001ms

表 6 ハッシュ値を生成し保存する場合 (OpenFlow スイッチ 2 台)

no_write_no_flow

9.07MB	7.6Mbps	13.4MB	10.4Mbps	16.915ms
8.7MB	7.29Mbps	10.8MB	8.57Mbps	1.034ms
8.9484MB	7.4986Mbps	12.836MB	10.0216Mbps	6.89232ms
8.945MB	7.495Mbps	13.1MB	10.3Mbps	5.8305ms

no_write_flow_1s

252MB	211Mbps	120MB	100Mbps	3.792ms
239MB	201Mbps	117MB	97.8Mbps	0.001ms
246.37MB	206.54Mbps	118.93MB	99.683Mbps	0.40607ms
247MB	207Mbps	119MB	99.9Mbps	0.001ms

no_write_flow_60s

256MB	215Mbps	120MB	100Mbps	1.891ms
241MB	202Mbps	117MB	97.8Mbps	0.001ms
250.58MB	210.14Mbps	119.31MB	99.689Mbps	0.39045ms
251MB	211Mbps	120MB	100Mbps	0.001ms

表 7 ハッシュ値を生成するが保存しない場合 (OpenFlow スイッチ 2 台)

no_cap_no_flow

12.5MB	10.5Mbps	21.1MB	16.9Mbps	8.953ms
12MB	10.1Mbps	17.1MB	13.9Mbps	0.946ms
12.411MB	10.405Mbps	19.387MB	15.601Mbps	5.22802ms
12.4MB	10.4Mbps	20.45MB	16.4Mbps	5.703ms

no_cap_flow_1s

253MB	212Mbps	120MB	100Mbps	2.222ms
241MB	202Mbps	117MB	98.2Mbps	0.001ms
247.28MB	207.22Mbps	119.04MB	99.775Mbps	0.39047ms
247MB	207Mbps	119MB	100Mbps	0.001ms

no_cap_flow_60s

261MB	219Mbps	120MB	100Mbps	1.891ms
244MB	205Mbps	118MB	98.5Mbps	0.001ms
250.74MB	210.2Mbps	119.29MB	99.796Mbps	0.35589ms
251MB	211Mbps	119MB	100Mbps	0.001ms

表 8 ハッシュ値を生成しない場合 (OpenFlow スイッチ 2 台)

simple_back_no_flow

23.6MB	19.7Mbps	47.9MB	39.4Mbps	4.239ms
22.2MB	18.6Mbps	37.6MB	30.8Mbps	0.917ms
23.027MB	19.294Mbps	46.112MB	37.837Mbps	2.00594ms
23MB	19.3Mbps	47.1MB	38.7Mbps	1.9615ms

simple_back_flow_1s

227MB	190Mbps	120MB	100Mbps	1.892ms
196MB	165Mbps	116MB	97.3Mbps	0.001ms
208.65MB	174.87Mbps	119.07MB	99.677Mbps	0.42878ms
209MB	175Mbps	119MB	100Mbps	0.001ms

simple_back_flow_60s

255MB	214Mbps	120MB	100Mbps	1.891ms
242MB	203Mbps	117MB	97.9Mbps	0.001ms
249.77MB	209.44Mbps	119.42MB	99.701Mbps	0.38246ms
250MB	210Mbps	120MB	100Mbps	0.001ms

表 9 静的に IP アドレスと MAC アドレスの組みを設定し，ハッシュ値を生成し保存する場合 (OpenFlow スイッチ 2 台)

no_back_no_flow

44.5MB	37.4Mbps	120MB	100Mbps	1.891ms
41.9MB	35.2Mbps	117MB	98Mbps	0.001ms
43.209MB	36.219Mbps	119.53MB	99.827Mbps	0.74216ms
43.2MB	36.2Mbps	120MB	100Mbps	0.917ms

no_back_flow_1s

253MB	212Mbps	120MB	100Mbps	1.891ms
241MB	202Mbps	117MB	97.8Mbps	0.001ms
249.01MB	208.71Mbps	119.05MB	99.653Mbps	0.38481ms
250MB	209.5Mbps	119MB	100Mbps	0.001ms

no_back_flow_60s

256MB	215Mbps	120MB	100Mbps	1.891ms
243MB	204Mbps	117MB	98.3Mbps	0.001ms
249.85MB	209.46Mbps	119.37MB	99.823Mbps	0.34613ms
250MB	210Mbps	120MB	100Mbps	0.001ms

表 10 静的に IP アドレスと MAC アドレスの組みを設定し，ハッシュ値を生成し保存する場合 (OpenFlow スイッチ 1 台)

各ネットワークにおいて，Iperf[5] を用いて OpenFlow コントローラの性能を測定した．測定内容は次のとおりである．

- full
  動的に IP アドレスと MAC アドレスの組みを収集し，OpenFlow コントローラでパケットの情報からハッシュ値を生成し保存する．
- no_write
  動的に IP アドレスと MAC アドレスの組みを収集し，OpenFlow コントローラでパケットの情報からハッシュ値を生成するが，保存しない．
- no_cap OpenFlow コントローラでパケットの情報からハッシュ値を生成しない．
- simple_back
  静的に IP アドレスと MAC アドレスの組みを設定し，ハッシュ値を生成し保存する．
- no_back
  静的に IP アドレスと MAC アドレスの組みを設定し，ハッシュ値を生成しない．

さらに，フローエントリをまったく作らない場合，フローエントリの生存時間を 1 秒，60 秒にした場合を測定した．

測定結果を表 1〜表 10 に示す．この図では，no_flow はフローエントリーを作らない場合，flow_1s はフローエントリーの生存時間が 1 秒，flow_60s はフローエントリ 60s はフローエントリーの生存時間が 60 秒の場合を示す．各表は，左側から右側に，TCP での送信したデータ量，TCP でのパケット転送速度，UDP での送信したデータ量，UDP でのパケット転送速度，UDP の接続の遅延を表す．また，上段から，最大値，最小値，平均値，中央値をそれぞれ表している．

6. 考察

表 1〜表 10 までの結果から，次のことが言える．

6.1 OpenFlow スイッチの台数について

OpenFlow スイッチ 1 台と OpenFlow スイッチ 2 台の場合を比較すると，フローエントリを保存しない場合，処理速度が低下している．しかし，フローエントリの生存時間を長くすれば，OpenFlow スイッチ 1 台の場合と 2 台の場合の差が少なくなるが，差がなくなることはない．OpenFlow スイッチの台数が増加すれば，その分，OpenFlow コントローラの負荷が増加するため，この測定結果は自然である．フローエントリを保存しない場合，それぞれの OpenFlow スイッチにパケットが入力されたときに，IP Traceback のための処理がスイッチの台数に比例して OpenFlow コントローラに負荷がかかると思われる．

6.2 MAC アドレスと IP アドレスの取得方法について

IP Traceback のための情報として，OpenFlow スイッチが受け取ったパケットの送信元，つまり，パケットを中継したネットワーク機器の MAC アドレスと IP アドレスを記録する．このとき，Mac アドレスと IP アドレスの取得方法としては，ARP テーブルからそれら 2 つのアドレスを取得する方法がある．この方法の処理負荷を考察する．MAC アドレスと IP アドレスを ARP テーブルから取得する場合，あらかじめ，静的にこれらの情報を与えた場合とを比較する．表 1 と表 4 において，フローエントリを保存しない場合を比較すると，3 倍近い差がでている．よって，ARP テーブルから動的にアドレスを取得することは負荷が大きいことがわかる．また，OpenFlow スイッチが 2 台の場合であっても，同様に，フローエントリを保存しない場合，3 倍近い差が出ている．

しかし，フローエントリを OpenFlow スイッチで保存する場合には ARP テーブルの利用では差が生じない．これは，IP Traceback 自体の処理が減少するためである．実際，表 1 と表 4 のフローエントリを利用する場合を比較すれば，差がないことがわかる．

6.3 IP Traceback による負荷について

本稿で提案した IP Traceback の仕組みを実装しない場合としない場合の負荷を比較する．表 1 と表 5 を比較することで，処理負荷を知ることができる．フローエントリを保存しない場合，転送速度が 5 倍近く，IP Traceback の負荷が大きいことがわかる．一方で，フローエントリを 1 秒間保存する場合には，速度差が少ない．このことから，フローエントリを短時間であっても保存することにより，IP Traceback のための処理を軽減することができる．

6.4 IP Traceback を構成する各処理の負荷について

本稿で提案した IP Traceback の仕組みは，OpenFlow コントローラがパケットを受け取ったあとで，パケットを中継したネットワーク機器の MAC アドレスと IP アドレスを取得し，次にパケットの情報と時刻からハッシュ値を計算し，そのハッシュ値を保存する．このハッシュ値が IP Traceback を行う際に利用される．表 1 は全ての処理を実装した場合，表 2 はハッシュ値の保存をしない場合，表 3 はハッシュ値の計算をしない場合である．これらを比較すると，ハッシュ値の保存と計算それぞれで負荷がかかることがわかる．また，表 1 と表 4 を比較すると，パケットを中継したネットワーク機器の MAC アドレスと IP アドレスを ARP テーブルから調べることで，負荷が推測できる．ARP テーブルを調べることの負荷が，ハッシュ値の計算やハッシュ値の保存の負荷よりも大きいことがわかる．

6.5 フローエントリの保存時間について

各表において，フローエントリを保存しない場合と，フローエントリを保存する場合を比較すると，1 秒であっても保存することにより，IP Traceback のための負荷を軽減することができることがわかる．今回の実験では，Iperf によるパケットの処理速度の測定であり，Web ブラウジングなど，トラフィックの種類によっては異なる結果となりうる．つまり，フローエントリの保存時間が 1 秒では IP Traceback の処理負荷が大きくなる場合があり，様々なトラフィックの種類に対して，実験を行う必要がある．

7. おわりに

本稿では OpenFlow コントローラを利用した IP Traceback を提案した．また，実装し，IP Traceback の処理の負荷を測定した．今後は，より大規模な実験を行い，本稿で提案した IP Traceback の方法を実用化を目指す．

参考文献

[1] draft-ietf-itrace-04 - ICMP Traceback Messages Steve Bellovin
[2] Advanced and Authenticated Marking Schemes for IP-Traceback Dawn Xiaodong Song and Adrian Perrig
[3] draft-hazeyama-traceback-field-trial-00 - A Field Trial of Inter-Domain Traceback Operation in Japan:
[4] 低レート DoS 攻撃と反射型 DoS 攻撃に対応した改良型 ICMP Traceback 高田　友則
[5] Iperf 入手先 ⟨https://Iperf.fr⟩ (2016.01.15).