Vol.2019-IOT-46 No.3
2019/6/14

OpenFlowを用いたDNSアンプ攻撃対策のフラグメント化を考慮した改善

櫻井 理寛[1]　Patipon Suwanbol[2]　山井 成良[3,a)]　北川 直哉[3]　Vasaka Visoottiviseth[2]

概要：
SDN（Software Defined Network）はコントローラと呼ばれるソフトウェアによってネットワークを制御する技術であり，ネットワークを動的に制御できるなど様々な恩恵を受けることができる点で注目されている．しかし，コントローラでの処理はオーバヘッドが比較的大きいため，フラグメント化されたパケットなどコントローラが短時間で大量のパケットを処理する必要がある状況では，十分なスループットが得られない可能性がある．本稿ではSDNを用いたDNSアンプ攻撃対策に焦点を当て，従来提案されてきた対策手法がフラグメント化されたパケットをうまく処理できない点を指摘するとともに，それらの問題点の改善方法を提案する．

キーワード：
SDN，OpenFlow，フラグメント化，分散型サービス不能攻撃，DNSアンプ攻撃

Improvement of Countermeasure for DNS Amplification Attack with OpenFlow Considering Packet Fragmentation

Masahiro Sakurai[1]　Patipon Suwanbol[2]　Nariyoshi Yamai[3,a)]　Naoya Kitagawa[3]
Vasaka Visoottiviseth[2]

Abstract:
Software Defined Network (SDN) is a technology that allows a software program called "controller" to manage networks. This technology attracts much attention nowadays since it has many advantages such as dynamic and flexible network configuration. However, it may decrease the throughput of SDN controlled network especially when the controller deals with many packets such as fragmented packets since the overhead for the controller to process packets tends to be relatively large. In this paper, we focus on DNS amplification attack countermeasures using SDN. First, we point out a problem of an existing countermeasure which did not take fragmented packets into account and then propose an improved method to mitigate this problem.

Keywords:
SDN, OpenFlow, fragmentation, Distributed Denial of Service attack, DNS amplification attack

1. はじめに

SDN（Software Defined Network）[1], [2] はコントローラと呼ばれるソフトウェアによってネットワークを制御する技術であり，ネットワークを動的に制御できるなど様々

[1] 東京農工大学工学部
Faculty of Engineering, Tokyo University of Agriculture and Technology
2-24-16, Nakacho, Koganei, Tokyo 184–8588, Japan
[2] Faculty of Information and Communication Technology, Mahidol University,
999 Phuttamonthon 4 Road, Salaya , Nakhon Pathom 73170 Thailand
[3] 東京農工大学工学研究院
Institute of Engineering, Tokyo University of Agriculture and Technology
2-24-16, Nakacho, Koganei, Tokyo 184–8588, Japan
[a)] nyamai@cc.tuat.ac.jp

な恩恵を受けることができる点で注目されている．従来のネットワークでは，ネットワークを構成するサーバやネットワーク機器の追加，削除などにより，その構成に変更が生じた場合にネットワーク管理者の負担が多い点が問題であった [2]．これに対して SDN ではソフトウェアによってネットワークが管理可能となるため，ネットワーク構成を物理的に変更する必要がなくなり，ネットワーク管理者の負担が軽減される．

SDN ではコントローラでネットワーク全体を一元管理しているため，コントローラのセキュリティ対策は重要である．特に SDN の代表的なプロトコルである OpenFlow[3] では，DoS（Denial of Service）攻撃や DDoS（Distributed Denial of Service）攻撃への対策を怠ると，従来のネットワークよりも悪い影響が出る可能性があることが指摘されている [4], [5]．

そこで，本稿では代表的な DDoS 攻撃である DNS アンプ攻撃 [6] を対象とし，OpenFlow を用いた従来の対策手法 [7] が大量のパケットをコントローラで処理しているためコントローラが過負荷になる問題点を指摘するとともに，それらの問題点を改善し，さらにフラグメント化されたパケットの扱いについても考慮した新たな OpenFlow ネットワークの構築方法の提案およびその評価を行う．

2. DNS アンプ攻撃と既存の対策手法

2.1 DNS アンプ攻撃

DNS アンプ攻撃とは DNS サーバを介して行われる DDoS 攻撃の一種である．攻撃の流れを図 **1** に示す．

DNS アンプ攻撃では，まず攻撃者がボットネットを用いて攻撃対象の IP アドレスを送信元 IP アドレスとして偽装した大量の DNS 問合せメッセージを UDP パケットとして多数の踏み台 DNS サーバに一斉送信する．この問合せメッセージを受信した DNS サーバは問合せメッセージ中の送信元 IP アドレスに基づいて応答メッセージを攻撃対象に返信する．その際，応答メッセージが問合せメッセージに比べて大きければ大きいほど攻撃の効率が高まり，一般には応答メッセージは問合せメッセージの数十倍の大きさになる．

DNS アンプ攻撃はこのように第三者の DNS サーバを介したリフレクション攻撃（Distributed Reflection Denial of Service attack，DRDoS）であるため，その対策は困難である．根本的な対策としては，特に家庭用ルータが持つ，外部からの DNS 問合せメッセージに応答する（オープンリゾルバ）という脆弱性を修正することであるが，これらのルータの管理者は専門的な知識を持っておらず，またその必要性を認識していないことが多いため，実際には困難である．

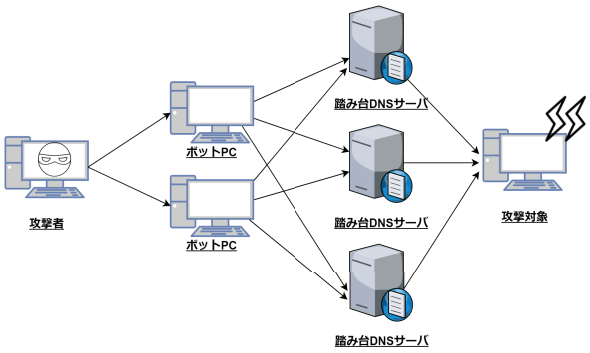

図 **1** DNS アンプ攻撃の流れ

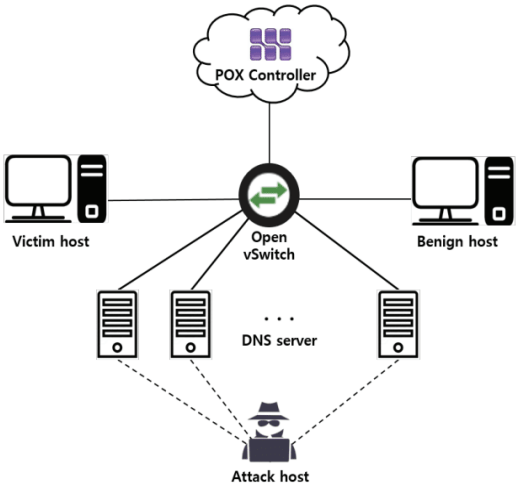

図 **2** 既存の対策手法におけるネットワーク構成 [7]

2.2 DNS アンプ攻撃への既存の対策手法

DNS アンプ攻撃に対する既存の対策手法として，文献 [7] が挙げられる．この手法では図 **2** に示すような OpenFlow を用いたネットワーク構成において OpenFlow スイッチ（Open vSwitch[8]）が全ての DNS 問合せメッセージ，応答メッセージを OpenFlow コントローラ（POX Controller[9]）に送り，組織内ホスト（Victim host）からの問合せに対する正規の DNS サーバ（Benign host）からの応答メッセージのみを転送し，それ以外の DNS サーバ（DNS server）からの応答メッセージを廃棄するように動作する．具体的な動作を以下に示す．

(1) OpenFlow スイッチは組織内ホストから受信した DNS 問合せメッセージおよび全ての受信した DNS 応答メッセージを OpenFlow コントローラに転送する．
(2) OpenFlow コントローラは DNS 問合せメッセージを受け取った場合には (3) に進む．そうではなく DNS 応答メッセージを受け取った場合には (4) に進む．
(3) OpenFlow コントローラは Request History と呼ばれるリストに DNS 問合せメッセージの情報を追加し，OpenFlow スイッチに DNS 問合せメッセージを宛先に転送するように指示する．その後，(1) に戻る．

(4) OpenFlow コントローラは Request History を参照し，DNS 応答メッセージに対応する DNS 問合せメッセージが登録されているか確認する．登録されている場合には (5) に，そうでなければ (6) に進む．
(5) OpenFlow コントローラは Request History から対応する DNS 問合せメッセージの情報を消去する．その後，OpenFlow スイッチには DNS 応答メッセージを宛先に転送するよう指示する．その後，(1) に戻る．
(6) 廃棄する．その後，(1) に戻る．

文献 [7] では仮想ネットワーク mininet[10] でこの手法の性能を評価し，その有効性を確認している．しかし，この手法を実際のネットワークで評価するとコントローラが過負荷になる可能性が高く，改善の余地が残されている．また，後述するように実際のネットワークでは DNS アンプ攻撃で使われるような大きな DNS 応答メッセージがフラグメント化されるのに対して，この手法では mininet 上でフラグメント化が行われなかったため正常に動作した可能性がある．

3. DNS アンプ攻撃対策の改善

3.1 提案手法の概要

文献 [7] の手法で致命的であった処理は，Request History を OpenFlow コントローラ内に用意し，OpenFlow スイッチが DNS 応答メッセージを受け取るたびに OpenFlow コントローラに転送し，OpenFlow コントローラがその都度パケットの転送・破棄の判定を行っていることにある．したがって，この Request History に相当する機能を OpenFlow スイッチ側に用意することでこの問題は解決すると考えられる．そこで我々は DNS アンプ攻撃用のパケットを OpenFlow コントローラに転送せず，OpenFlow スイッチが直ちに廃棄できるように設定する方法を検討した．この方法では以下のようなフローエントリを OpenFlow スイッチに登録すると機能するものと判断していた．

- OpenFlow スイッチ起動時に Victim host 宛の DNS メッセージ（送信元ポート番号 53）を全て廃棄するフローエントリを低優先度で登録する．
- Victim host から DNS メッセージ（宛先ポート番号 53）を受け取った際には，このパケットを OpenFlow コントローラに転送する．
- OpenFlow コントローラは OpenFlow スイッチからパケットを受け取ると，これが DNS 問合せメッセージであることを確認し，これに対する DNS 応答メッセージを Victim host に中継するフローエントリを高優先度で OpenFlow コントローラに作成する．
- フローエントリ作成後に OpenFlow コントローラは受け取った DNS 問合せメッセージを本来の宛先に中継するように OpenFlow スイッチに指示する．

この方法を Raspberry Pi 2 上で実装し，小さなパケット（約 400 オクテット）を用いて Victim host を攻撃する実験を行ったところ，有効に機能することが確認できた．しかし，実際の DNS アンプ攻撃に用いられるような大きなパケット（約 4000 オクテット）を用いて同じ実験を行ったところ，攻撃用パケットが全て OpenFlow コントローラに転送され，うまく機能しなかった．その原因を調査したところ，DNS 応答メッセージがフラグメント化されていたため，OpenFlow スイッチがパケットの UDP ポート番号を無視し，デフォルトの動作により OpenFlow コントローラに転送していることが判明した．

そこで，提案手法では正確性では多少劣るが，フラグメント化されたパケットでもマッチするフローエントリを作成することにより DNS アンプ攻撃を緩和できるようにした．

3.2 OpenFlow における IP フラグメントの処理

IP パケットが経路上の MTU（Maximum Transmission Unit）を超える大きさである場合，送信元あるいは経由するレイヤ 3 機器によりフラグメント化される．この場合，TCP/UDP ポート番号や ICMP のタイプとコードフィールドは最初のフラグメントのみに含まれ，後続のフラグメントには含まれない．したがって，OpenFlow スイッチにより IP フラグメントを処理する場合には，注意が必要である．

OpenFlow スイッチの仕様 [11] では，Version 1.3 より IP フラグメントの処理方法が明記された．OpenFlow スイッチではフラグメントの再構築機能を持つことができ，この機能を使えば全てのフラグメントから元のパケットを再構築した後にフローエントリとの比較を行うことができる．しかし，この機能は補助機能であり，利用可能な OpenFlow スイッチは一般的ではない．

OpenFlow スイッチのソフトウェアでの実装である Open vSwitch でも IP フラグメントの処理モードとして normal, drop, reassemble, nx-match の 4 つが指定可能であるが，再構築を行う reassemble モードは実装されていないことがマニュアル [12] に明記されている．IP フラグメントに対するデフォルトの処理モードは normal であるが，このモードでは各フラグメントの TCP/UDP ポート番号や ICMP のタイプとコードフィールドには常に 0 が設定される．また，nx-match モードでは最初のフラグメントのみ TCP/UDP ポート番号などをフローエントリとの比較に使用できるが，後続のフラグメントでは常に 0 が設定される．したがって，3.1 で述べた方法では normal モード，nx-match モードのいずれの設定でもフラグメントは正しく処理されず，OpenFlow コントローラに転送されることになる．

なお，OpenFlow スイッチでフラグメントから元のパケットを再構築するのではなく，代わりに最初のフラグメントに含まれる TCP/UDP ポート番号などと併せて IP

ヘッダ中の識別子（IP-ID）を記録し，後続のフラグメントは IP-ID に基づいて TCP/UDP ポート番号などの情報を復元してフローエントリと比較する方法 [13] が存在する．しかし，この方法は一般には利用できず，また DNS アンプ攻撃では IP-ID の管理などでオーバヘッドが大きくなり，OpenFlow スイッチが過負荷になる可能性がある．

3.2.1 提案手法の詳細

上記のように OpenFlow では IP フラグメントを UDP ポート番号に基づいて処理できない．したがって，DNS アンプ攻撃のパケットを OpenFlow スイッチで廃棄するには，必然的に IP ヘッダ中のフィールドに基づいたフローエントリを作成する必要がある．

DNS アンプ攻撃では攻撃用パケットは必ず UDP で送られるため，IP ヘッダ中のプロトコルフィールドの値が 17（UDP）であるパケットは原則として廃棄し，Victim host から送信された UDP パケットおよびその宛先から返される UDP パケットだけを中継するように OpenFlow スイッチを設定すればよいことになる．

したがって，提案手法では OpenFlow コントローラおよび OpenFlow スイッチが以下の動作を行うようにする．

(1) 初期設定として，OpenFlow コントローラは OpenFlow スイッチに「宛先 IP アドレスが Victim host，イーサネットタイプが 0x0800(IP)，IP プロトコル番号が 0x11(UDP) の 3 条件を全て満たしていれば廃棄する」というフローエントリを低優先度で登録しておく．また，これにマッチしない UDP パケットは OpenFlow コントローラに転送するように設定しておく．

(2) OpenFlow スイッチは受信した UDP パケットが (1) で登録したフローエントリにマッチすればそのパケットを廃棄し，(3) で登録したフローエントリにマッチすればそのパケットを Victim host に中継する．それ以外の UDP パケットは OpenFlow コントローラに中継する．

(3) OpenFlow コントローラは OpenFlow スイッチから転送された UDP パケットに対する物理ポート番号が Victim host が接続されているもので，かつ宛先ポート番号が 53 であるかを調べ，両方の条件が満たされている場合には OpenFlow スイッチに「宛先 IP アドレスが Victim host，送信元 IP アドレスがこのパケットの宛先 IP アドレス，イーサネットタイプが 0x0800(IP)，IP プロトコル番号が 0x11(UDP) の 4 条件を全て満たしていれば Victim host が接続されている物理ポートに中継する」というフローエントリを高優先度で追加する．なお，このフローエントリは一定時間（後述の実験では 15 秒）経過後に自動的に削除されるように設定する．

ここで Victim host が EDNS（Extension mechanisms for DNS）[14] を用いずに問合せをおこなったり，EDNS でバッファサイズを 512 バイトに指定したりする方法で DNS 応答メッセージが 512 バイトを超えないことが保証される場合，DNS 応答メッセージのフラグメント化が行われないため送信元 UDP ポート番号に基づくフローエントリが機能することが期待できる．

なお，提案手法では DNS 以外の UDP 通信も廃棄の対象になるが，DNS と同様に Victim host から先に UDP パケットが送信される通信については上記と同様の動作により OpenFlow スイッチにパケットを中継させることが可能である．

4. 性能評価

4.1 性能評価の実験環境

提案手法の有効性を評価するため，**図 3** に示す実験ネットワークにおいて Victim が DNS アンプ攻撃を受けながら dig コマンドにより Benign（DNS サーバ）に 1 秒間隔で 50 回 DNS 問合せメッセージを送出し，5 秒以内に応答が得られなかった割合および応答が得られた場合の平均応答時間を測定した．同図中の全てのホスト，OpenFlow スイッチ，OpenFlow コントローラには Raspberry Pi 2 Model B を使用し，OpenFlow スイッチには Open vSwitch[8]，OpenFlow コントローラには Trema[15], [16] を用いた．したがってネットワーク帯域は全てのリンクで 100Mbps である．Attacker が踏み台用 DNS サーバに送信元 IP アドレスを偽装した DNS 問合せメッセージを送信する際には，Scapy[17] を用いた．Scapy ではパケット送出速度（単位：packets per second (pps)）を指定することが可能であるが，必ずしも指定した通りの送出速度が得られるとは限らないため，実際の送出速度も測定した．踏み台用 DNS サーバから Victim に送られる DNS 応答メッセージの大きさは Attacker からの問合せにおけるレコードタイプに依存し，A の場合に約 400 オクテット，ANY の場合は約 4000 オクテットである．

この性能評価実験では，比較の対象として何も対策を行わないで OpenFlow スイッチが全てのパケットを中継する場合（No protecting）および文献 [7] の方法を用いた場合（Existing）についても同様の測定を行った．但し，後者の方法では Request History は更新せず常に Victim・Benign 間の DNS メッセージを中継するようにし，それ以外の送信元から Victim 宛に送られるパケットは送信元・宛先 IP アドレスに基づいて OpenFlow スイッチが廃棄するようにした．

4.2 DNS アンプ攻撃下における DNS 問合せ失敗率

まず，Attacker が DNS アンプ攻撃を実行している間，Victim が Benign に対して 1 秒間隔で DNS 問合せメッセージを送出し，5 秒以内に応答がなかった割合（問合せ失敗率）を測定した．この実験では攻撃者の偽装 DNS 問合せ

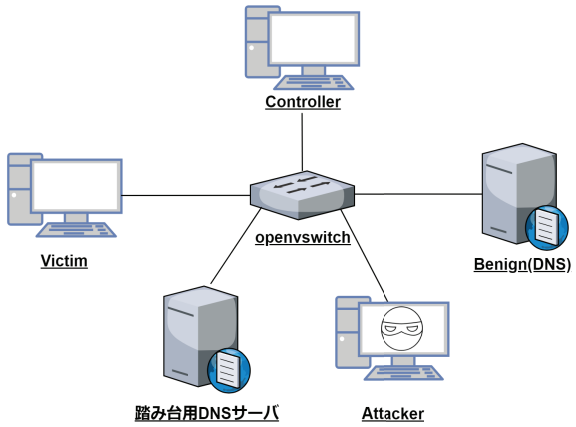

図 3　性能評価実験におけるネットワーク構成

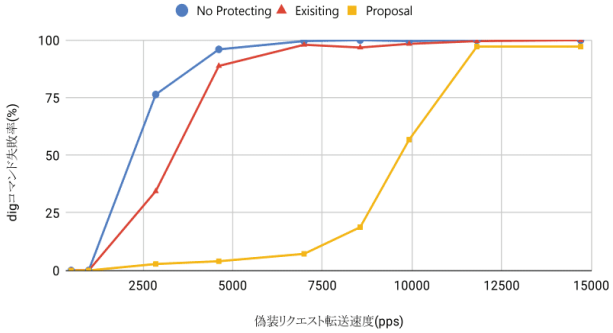

図 4　A タイプによる攻撃下での問合せ失敗率

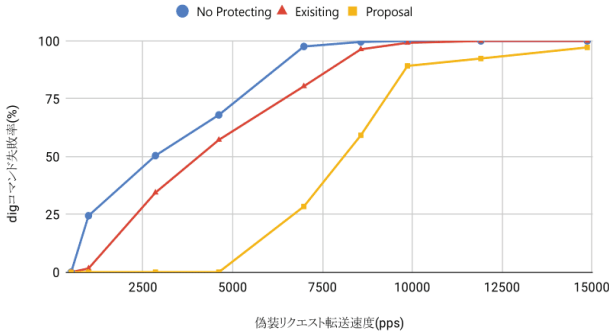

図 5　ANY タイプによる攻撃下での問合せ失敗率

メッセージを約 500-約 15000pps（実測値）の間で変化させ，各点について 5 回計測を行った．踏み台用 DNS サーバからの応答が A タイプの場合（約 400 オクテット）の結果を図 **4** に，ANY タイプの場合（約 4000 オクテット）の結果を図 **5** に示す．

これらの結果から，いずれの場合でも提案手法は既存の対策手法に比べて DNS アンプ攻撃に対する耐性が高いことが確認できる．図 4 と図 5 との違いは以下のように説明できる．No protecting と Existing の場合，A タイプによる攻撃では小さな攻撃用パケットが OpenFlow コントローラに転送されるため OpenFlow スイッチあるいはネットワークの帯域には余裕がある段階で OpenFlow コントローラが過負荷になり，失敗率に大きな影響を与えた一方，Proposal では 7000pps までは OpenFlow コントローラが過負荷にならず，失敗率があまり大きくならない．また，

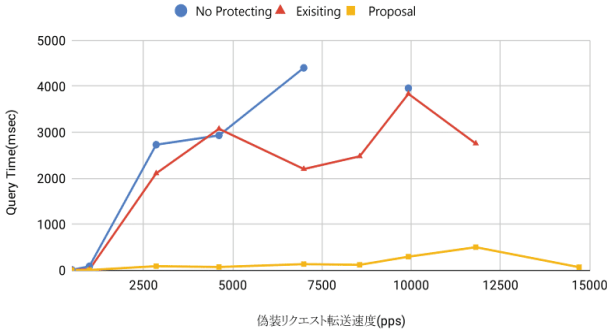

図 6　A タイプによる攻撃下での平均応答時間

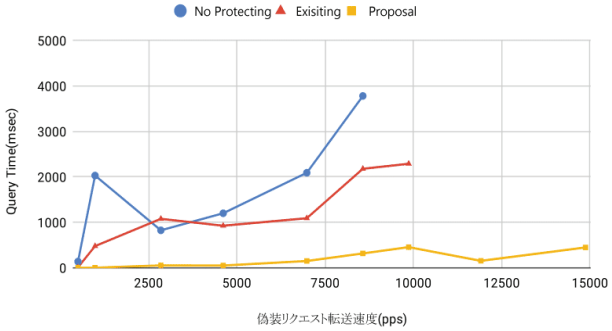

図 7　ANY タイプによる攻撃下での平均応答時間

ANY タイプによる攻撃では OpenFlow スイッチが先に過負荷になるか，ネットワークの帯域を超えるトラフィックが発生するため，7000pps 程度の比較的低速な攻撃でも Proposal に影響が出る一方，OpenFlow コントローラはそれほど負荷が高くないため A タイプによる攻撃の場合と比較して失敗率が高くならない．

4.3　DNS アンプ攻撃下における平均応答時間

次に，前節と同じ条件で実験を行い，5 秒以内に応答があった場合における平均応答時間を測定した．踏み台用 DNS サーバからの応答が A タイプの場合の結果を図 **6** に，ANY タイプの場合の結果を図 **7** に示す．なお，これらの図において No protecting と Existing では欠落している部分があるが，これらの部分は計測中に正常な DNS 応答メッセージを受信できなかったことを示す．

これらの図からわかるように，Proposal ではどの頻度の攻撃においても十分に小さな平均応答時間が得られていることがわかる．No protecting と Existing ではいずれも ANY タイプによる攻撃下のほうが平均応答時間が小さい傾向にあるが，その理由としては ANY タイプによる攻撃下では OpenFlow コントローラの負荷やネットワークの帯域に比較的余裕があり，OpenFlow スイッチから OpenFlow コントローラへのパケット転送が成功すれば OpenFlow コントローラが遅滞なく処理をできるためと考えられる．

5.　まとめ

本稿では SDN を用いた DNS アンプ攻撃対策に焦点を当

て，従来提案されてきたポート番号に基づく攻撃用パケットの廃棄がパケットのフラグメント化によりうまく機能しないことを指摘するとともに，IP ヘッダ中の IP アドレスとプロトコル番号に基づいて問合せ先以外からのホストからのパケットを攻撃とみなして OpenFlow スイッチで廃棄する方法を提案した．また，性能評価実験により提案手法が既存の対策手法より攻撃への耐性が高いことを確認した．

今後の課題としては，実環境での性能評価が挙げられる．また提案手法では全ての UDP パケットを廃棄の対象としていることから，この処理が悪影響を及ぼすかどうか調査することも実用化に向けての重要な課題である．

参考文献

[1] E. Haleplidis, K. Pentikousis (Eds.), S. Denazis, J. Hadi Salim, D. Meyer, O. Koufopavlou: Software-Defined Networking (SDN): Layers and Architecture Terminology, RFC7426, IETF, January 2015.

[2] Diego Kreutz, Fernando M. V. Ramos, Paulo Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig: “Software-Defined Networking: A Comprehensive Survey,” *Proceedings of the IEEE,* Vol.103, No.1, pp.14–76, Januray 2015.

[3] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner: “OpenFlow: enabling innovation in campus networks,” *ACM SIGCOMM Computer Communication Review,* Vol.38, No.2, pp.69–74, April 2008.

[4] Sandra Scott-Hayward, Gemma O’Callaghan, Sakir Sezer: “Sdn Security: A Survey,” *Proceedings of 2013 IEEE SDN For Future Networks and Services (SDN4FNS),* pp.1–7, 2013.

[5] Zhiyuan Hu, Mingwen Wang, Xueqiang Yan, Yueming Yin,Zhigang Luo: “A comprehensive security architecture for SDN,” *Proceedings of 18th International Conference on Intelligence in Next Generation Networks,* pp.30–37, February 2015.

[6] ICANN Security and Stability AdvisoryCommittee: SSAC Advisory SAC008 DNS Distributed Denial of Service (DDoS) Attacks (online), available from ⟨https://www.icann.org/en/system/files/files/dns-ddos-advisory-31mar06-en.pdf⟩ (accessed 2019-05-19).

[7] Soyoung Kim, Sora Lee, Geumhwan Cho, Muhammad Ejaz Ahmed, Jaehoon (Paul) Jeong, Hyoungshck Kim: “Preventing DNS amplification attacks using the history of DNS queries with SDN,” Computer Security – European Symposium on Research in Computer Security (ESORICS 2017), *Lecture Notes in Computer Science,* Vol.10493, pp.135–152, 2017.

[8] A Linux Foundation Collaborative Project: Open vSwitch (online), available from ⟨https://www.openvswitch.org/⟩ (accessed 2019-05-19).

[9] McCauley *et al.*: Installing POX — POX Manual Current documentation (online), available from ⟨https://noxrepo.github.io/pox-doc/html/⟩ (accessed 2019-05-19).

[10] Mininet Team: Mininet: An Instant Virtual Network on Your Laptop (or Other PC) (online), available from ⟨http://mininet.org/⟩ (accessed 2019-05-19).

[11] Open Networking Foundation: OpenFlow Switch Specification, Version 1.3.0 (online), 2012, available from ⟨https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf⟩ (accessed 2019-05-19).

[12] —: ovs-ofctl(8), Open vSwitch Manual (online), available from ⟨http://www.openvswitch.org/support/dist-docs/ovs-ofctl.8.txt⟩ (accessed 2019-05-19).

[13] KDDI 株式会社：OpenFlow スイッチおよびプログラム，特開 2015-070434.

[14] J. Damas, M. Graff, P. Vixie: Extension Mechanisms for DNS (EDNS(0)), RFC6891, IETF, April 2013.

[15] —: Trema (online), available from ⟨https://github.com/trema/trema⟩ (accessed 2019-05-19).

[16] 高宮安仁，鈴木一哉，松井暢之，村木暢哉，山崎泰宏：Trema で OpenFlow プログラミング (オンライン)，入手先 ⟨https://yasuhito.github.io/trema-book/⟩（参照 2019-05-19）.

[17] Shipra Bansal, Nitin Bansal: “Scapy — A Python Tool For Security Testing,” Journal of Computer Science & Systems Biology, Vol.8, No.3, pp.140–159, 2015.