DoSリフレクション攻撃の分析と防御法

野口 大貴[1,a)]　後藤 滋樹[1,b)]

概要：DRDoS（Distributed Reflection Denial of Service）攻撃による被害が増加している．DRDoS 攻撃は，攻撃者がリフレクターを利用して攻撃対象に向けて増幅されたパケットを送信して帯域負荷を増大させる．本研究では，ハニーポットで観測された DRDoS 攻撃のクエリパケットの時系列分析を行う．分析により攻撃クエリの検知に有用な時間の閾値を定める．この閾値を防御機構に組み込んでネットワークに内在するリフレクターの攻撃活動への加担を防ぐ手法を提案する．防御の方法は OpenFlow スイッチング技術による防御ネットワークを構築し，疑わしいクエリパケットの経路情報を活用して送信元情報の詐称を判定してフィルタリングを行う．この提案手法を実装した場合の負荷実験を行い，ルータでフィルタリングしても高負荷になりにくいことを実証した．

キーワード：MWS データセット，ネットワークセキュリティ，DRDoS 攻撃，OpenFlow

Defense against DRDoS Attacks by OpenFlow Switches

Daiki Noguchi[1,a)]　Shigeki Goto[1,b)]

Abstract: DRDoS attacks are serious problems. Victim servers suffer from heavy loads and network bandwidth consumption. Attackers use a variety of open servers around the world as reflectors which are forced to send huge amplified packets to victim servers. This paper analyzes malicious query packets captured by honey-pots. We investigate time intervals between attacking packets to find the threshold value between malicious and benign time intervals. This paper proposes a new protection mechanism based on the time interval. Our new method can prevent reflectors from participating in attacking mechanisms. We use OpenFlow switch to detect malicious query packets and filter them out. It is shown by the experiments that our filtering is light weight.

Keywords: MWS Dataset, Network Security, DRDoS Attack, OpenFlow

1. はじめに

DRDoS（Distributed Reflection Denial of Service）攻撃による被害が増加の傾向にある．DRDoS 攻撃（第 3 節）は，DDoS（Distributed Denial of Service）攻撃の中でもリフレクターと呼ばれるサーバを攻撃者が利用する．リフレクターとして選ばれるサーバは，多くの場合にクライアントに対し正常にサービスを提供するサーバであり，特に受信したクエリ情報に比べて大きなデータサイズで返答する機能を持つ特徴がある．攻撃者がこの機能を悪用し，送信元を攻撃対象に詐称したクエリを多量に送信することで攻撃対象のネットワーク機器への膨大な負荷が問題となる．

Akamai 社の Q3 2015 State of the Internet [1] によれば，DRDoS 攻撃は 2014 年第 3 四半期に比べて 462.44%増加しており，2015 年第 2 四半期から 40.14%増加し，SSDP や NTP を用いた攻撃が拡大している．この理由として，ボットネットの構築に比べてリフレクターは既存の脆弱なネットワーク機器を利用するため効率的であり，攻撃元を特定されにくい利点があるとしている [2]．リフレクターにされる機器には管理の行き届いていないものがあり [3]，その

[1] 早稲田大学 169-8555 東京都新宿区大久保 3-4-1
School of Fundamental Science and Engineering, Waseda University 3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555 JAPAN
[a)] noguchi@goto.info.waseda.ac.jp
[b)] goto@goto.info.waseda.ac.jp

対策としては世界中に遍在しているこれらのネットワーク機器に関して調査を行い，疑わしい場合はネットワーク管理者への報告を通して改善を試みる取り組みが行われている [4].

本研究では，ハニーポットで観測された DRDoS 攻撃のクエリパケットの特徴を分析し，その分析により攻撃クエリの検知に要求される待ち時間の閾値を定める．その閾値を OpenFlow[5] を用いた防御機構に組み込み，ネットワークに内在するリフレクターの攻撃活動への加担を防ぐ方法を提案する．

2. 関連研究

従来の DRDoS 攻撃についての研究では，攻撃者によって送信されるクエリパケットの特徴分析や，被害防止のためのネットワーク上での防御機構の提案などが行われている．

Christian[6] は，攻撃に用いられる UDP ベースのプロトコルの脆弱性について分析した上で，大規模 ISP やダークネットで観測された通信データ，および囮（おとり）として稼働させた脆弱な NTP サーバなどで観測された通信データから DRDoS 攻撃を抽出し分類している．攻撃者とリフレクター間の通信を pairflow という特徴量の集合で示し，また BAF というリフレクターへのクエリパケットと応答パケットの UDP ペイロードの比率を用いて通信データを分析している．Timm[7] らは BAF のみを指標として解析を行うと多くの正常な通信を悪性と判定してしまうことを指摘している．本研究では，クエリ及び応答パケットの UDP ペイロードサイズの比率ではなくパケットの到着間隔による攻撃通信の検知を目指す．すなわち連続するパケット群の時間間隔を指標としてデータ分析を行う．

ISP における DRDoS 攻撃対策ではイングレスフィルタリング [8] という攻撃者による送信元 IP アドレスが詐称されたパケットの送信を防ぐ技術がある．これはルータで受信したパケットの送信元 IP アドレスが管理ネットワーク上で割り当てられているか否かを判定し，不正なものは廃棄する．この手法では外部ネットワークからの詐称されたパケットを判別することは困難であり，世界中のネットワーク網での対応は未だ進んでいない [9]．また大容量のトラフィックが通過する ISP のエッジルータにおける DRDoS 攻撃対策には限界がある．本来の通信環境を保障できない等のリスクを伴う [10]．本研究ではリフレクター近傍で観測される外部ネットワークからのクエリパケットの送信元 IP アドレスの詐称を判定をしてフィルタリングを行う．この方法で ISP のエッジルータでの攻撃対策による負荷を軽減することができる．

3. DRDoS 攻撃

DRDoS（Distributed Reflection Denial of Service）攻撃は DDoS（Distributed Denial of Service）攻撃の一種であり，攻撃者がリフレクターを利用して攻撃対象に増幅されたパケットを送信することで帯域負荷を増大させる．リフレクターは世界中に遍在し，それらはクエリパケットに比べて大きなデータサイズで返答する機能を持つサーバであることが多い [2]．攻撃者から送信元を攻撃対象に詐称したクエリパケットを受信した場合に，リフレクターの側では悪意のあるパケットかどうかの判別が難しい．正常な通信とみなして返答してしまい，気づかずに攻撃プロセスに加担してしまうという問題がある．さらに，クエリに対して多くの情報で応答を行うリフレクターの特性上，攻撃者が送信するパケットサイズは最小限に留めても攻撃として有効であるから，攻撃者の負担すべきリソースが TCP SYN Flood 攻撃などの DDoS 攻撃に比べ節約できる．小規模の攻撃者グループでも深刻な被害を及ぼすことができる [11].

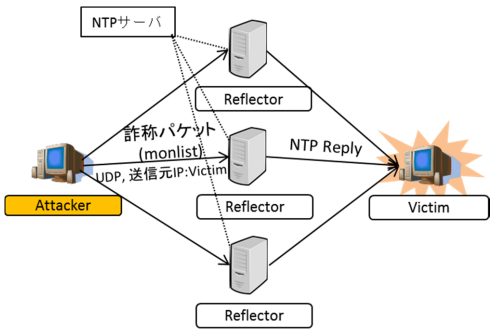

図 1　DRDoS 攻撃の例：NTP Amp 攻撃

4. OpenFlow

本研究では OpenFlow の技術を用いる．OpenFlow は，インターネットの再構築を目的としてスタンフォード大学を中心に研究開発しているオープンソースの技術である．OpefFlow は従来技術のスイッチの機能を再構成，細分化して独立させることにより，既存ネットワーク上で実験的なプロトコルを実装することを可能にしている．OpenFlow は通信の処理を行う基本単位をフローとして扱い，プログラムによって制御することが可能である．また，主にレイヤ 4 までのパケットヘッダ情報の書き換えが可能であり柔軟な転送処理を実現させている．日本では独立行政法人情報通信研究機構（NICT）が持つ研究開発テストベッドネットワーク JGN2plus [12] において実証実験が行われ，日米間を繋ぐ大規模なネットワーク上で OpenFlow の動作が確認された．これまで OpenFlow の標準化は OpenFlow スイッチングコンソーシアムを中心に進められてきたが，OpenFlow の実用化をより促進するために，2011 年 3 月 21 日に ONF（Open Networking Foundation） [5] が発足した．今後の OpenFlow の技術的仕様に関する議論は ONF で行われる．

表 1　サービスごとのクエリパケット総数とユニークホスト数

サービス	クエリパケット総数	ユニークホスト数	パケット/ホスト比
CharGen	83,023,544	4,304	19,289
DNS	32,058,041	415	77,248
NTP	72,449,934	104,446	693
SSDP	3,223,379	6,415	502

5. 分析対象とするデータ

第 3 節で説明した DRDoS 攻撃のクエリパケットを含むデータセットについて説明する．具体的なデータセットは，ハニーポットによる収集データである PRACTICE Dataset 2015 である．このデータセットには以下に述べる特徴がある．

5.1 PRACTICE Dataset 2015

PRACTICE Dataset 2015 [13] は，総務省委託研究「国際連携によるサイバー攻撃予知・即応技術の研究開発（H23-H27）」の支援を受け，横浜国立大学情報・物理セキュリティ研究拠点が開発して運用中の DRDoS ハニーポット [14] のトラフィックデータである．観測期間は 2015 年 5 月 31 日から 2015 年 6 月 6 日の一週間である．ハニーポット自体は前もって IP を固定しシミュレートしているため攻撃者にリフレクターとして探索されるまでは十分に時間がある．ハニーポットでシミュレートしているサービスは CharGen，DNS，NTP，SSDP の 4 種類である．

5.2 データセットの特徴

表 1 に 5 月 31 日から 6 月 6 日までのクエリパケット総数及びユニークホスト数を，攻撃の種類ごとに示す．パケット/ホスト比は小数点以下を切り捨てた．SSDP は他のサービスと比較してクエリパケット総数が少ないことが分かる．CharGen，DNS，NTP のクエリパケット総数の平均である 62,510,506 パケット（小数点以下切り捨て）と比較しても 5%程しかない．パケット/ホスト比が似ている NTP と比べて，SSDP が小規模であったことが分かる．NTP Amp 攻撃が DDoS 攻撃の件数では上位を占めている．CharGen や DNS ではホストに対するパケット数が大きい．その理由としてスキャン活動が活発であった可能性がある．スキャンの目的はリフレクターとなるサーバを探索することである．大量のスキャンパケットが送信されるとスキャン結果を受信するホストに負荷がかかる．その結果ホスト数が少なくなり表 1 の結果になったと推測される．

次に，IP ペイロード長の累積度数分布，及び観測日時における観測パケット数の分布を各サービスごとに図 2，図 3 に示す．累積度数分布（Cumulative Distribution Function，CDF）は累積度数の割合を確率として表現したもので，0 から 1.0 までの値で示され攻撃ごとの IP ペイロード長の分布を図示できる．IP ペイロード長はユニークホストごとに，受信したパケットのうち最も多く出現した長さである．各ユニークホストにおける IP ペイロード長の平均を表 2 に示す．この分布は最も多く出現したパケット長をもつパケット数の全体に対する割合である．なお CDF に図示した IP ペイロード長の中で，大半を占めるものを示したものが表 3 である．図 2 から，各サービスごとに IP ペイロード長が一定区間にまとまっていることがわかる．ここで NTP に着目すると広い範囲に分布しているように見えるが，プロットの個数が後半にかけて少なく，200 バイト付近は CDF の差も小さく外れ値とみなすとすれば，全体では 100 バイト以下にクエリパケットが収まっていることがわかる．これは小さいサイズの攻撃クエリを用いる DRDoS 攻撃の特徴をよく示している．最も多く分布する IP ペイロード長は最小のサイズではないことから，攻撃者が効率的に攻撃を行う共通の最適なサイズがサービスごとに存在することがわかる．また，表 2 から，ユニークホストごとのパケットごとのサイズの分布の範囲は小さい．図 2 のようなカウントの方法は特徴を表現するために適切である．

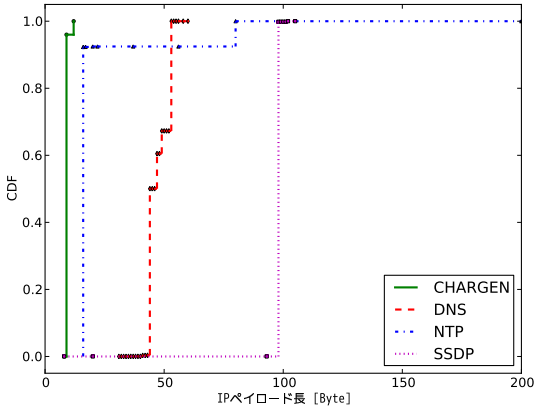

図 2　サービスごとの IP ペイロード長の累積度数分布

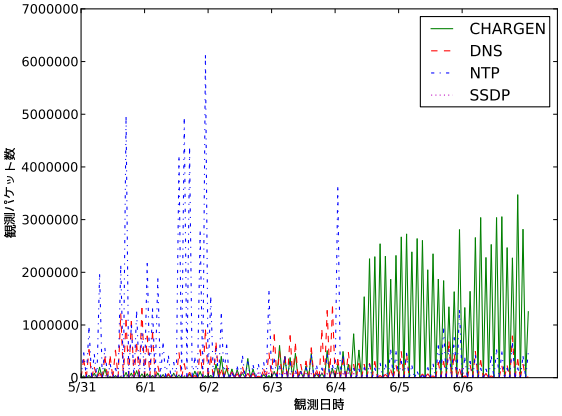

図 3　サービスごとの観測パケット数の分布

6. データ分析と考察

本研究は，リフレクター付近で DRDoS 攻撃によるクエ

表 2　各ユニークホストの IP ペイロード長の分布平均

サービス	最多分類サイズ [Byte]	割合 [%]
CharGen	9	99.90
DNS	44	97.81
NTP	16	99.99
SSDP	98	99.96

表 3　CDF におけるサービスごとの IP ペイロード長の割合

サービス	IP ペイロード長 [Byte]	割合 [%]
CharGen	9	95.98
	12	4.01
DNS	44	49.69
	53	32.72
NTP	16	92.32
	80	7.53
SSDP	98	99.85
	102	0.14

表 4　ユニークホストごとの隣接する到着間隔の比率が 0.9 以上であるパケットの割合

攻撃の種類	クエリパケット数	全パケットに対する割合 [%]
CharGen	20,656,329	24.8
DNS	7,854,492	24.5
NTP	17,606,713	24.3
SSDP	26,862	0.08

リパケットを検知して防御ネットワーク内で詐称判定を行う手法を提案する．検知に至るまでのクエリパケットの観測時間を適切に設定して，可能な限り負荷を軽減することが望ましい．そこで，検知に用いる閾値を選定する基準として，既存の攻撃ログを分析して攻撃の傾向に従った最適な値を求める．具体的には，ユニークホストごとのクエリパケット同士の到着間隔の特徴分析を行い，リフレクター付近でクエリパケットを最初に観測してから次に来るパケットを期待できる最大時間を選定する．

パケットの到着間隔の特徴を見るために，ユニークホストごとにパケットの到着間隔を計算する．隣接する長時間の間隔に対する短時間の間隔の比率が 0.9 以上であるパケットの割合を表 4 に示す．クエリパケット数はユニークホストごとに検出したパケット数の合計値である．

表 4 から，SSDP の全パケットに対する割合が小さい．この原因としてスキャンパケットの個数が影響していることが考えられるが，表 1 に示すように SSDP と NTP は同条件でありスキャンの可能性は低い．SSDP では，隣接する到着間隔の比率が 0.9 未満の連続したパケットが他のサービスに比べ多いと考えられる．CharGen，DNS，NTP では約 24%と一定である．CharGen や DNS はユニークホスト数が比較的少なく，各攻撃で連続したパケットの到着間隔が一定ではないことがわかる．この結果を踏まえて，適切に各サービスにおいて攻撃とみなされる連続したパケット群を判別するために，以下の事項に留意して分析を行う．

- 連続するクエリパケットの到着間隔の安定性（連続するパケット群において，隣接するパケット到着間隔同士の変動が無いか）を分析する
- ユニークホストごとに連続するパケット同士が別の攻撃であると判定する最大時間を「限界攻撃間隔」と定義する
- 複数クエリが同一の攻撃と認識され始めた最小値である 0.016 秒から，0.064，0.256，1.024，4.096，16.384，65.536，262.144，1048.576，4194.304 秒 を限界攻撃間隔とする
- 限界攻撃間隔ごとに同一のユニークホストへの独立した攻撃と判別する
- 各限界攻撃間隔ごとに分析を行う

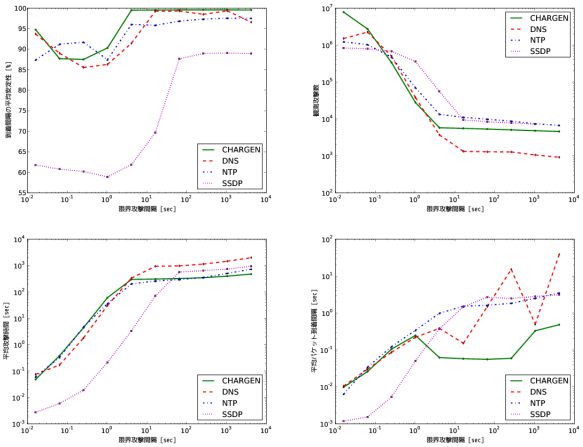

図 4　限界攻撃間隔と各値の関係

$$\sum_{n=1}^{all} \frac{\sum_{m=2}^{c_n} \frac{\mathrm{Min}\left(\frac{\sum_{k=1}^{m} i_{nk}}{m}, \frac{\sum_{k=1}^{m-1} i_{nk}}{m-1}\right)}{\mathrm{Max}\left(\frac{\sum_{k=1}^{m} i_{nk}}{m}, \frac{\sum_{k=1}^{m-1} i_{nk}}{m-1}\right)}}{c_n - 1}}{all}$$

図 5　到着間隔の平均安定性

到着間隔の平均安定性は all=観測攻撃数，c_n=n 番目の攻撃における到着間隔数，i_{nk}=k 番目の到着間隔として，図 5 の式で定義する．

α=限界攻撃間隔とする．図 4 では，α で区切られる 3 パケット以上の攻撃に対し，隣り合う到着間隔の安定性の平均を示した．α=0.016 では高レートの攻撃のみを対象とし，微小時間以上遅延して到着したパケットを別の攻撃とみなすため安定性は高くなるが，攻撃時間と到着間隔の平均は小さくなる．また，同一ユニークホストにおける攻撃において，限界攻撃間隔以上の遅延を含む 1 つの攻撃が分離されるため, 観測攻撃数が多くなる．

α = 0.064 から 16.384 にかけて CharGen，DNS，SSDP は下方にピークが現れており，異なる間隔を同じ攻撃とし

て認識するため安定性が下がり，短期間の攻撃において隣り合う攻撃がマージして観測攻撃数全体が下がる傾向にある．

NTP では一番目に上方ピークがあり，二番目に下方ピークが観測される．第 5 節の図 3 の結果から他のサービスと比較して非常に高いピークが観測日時全体で存在していることから，攻撃の発生時間が偏っていることがわかる．一時的に安定性が上がり，α=1.024 で急降下した要因として，α=0.064 以上のパケット群を含む攻撃が α=1.024 以上の間隔で局所的に発生し，攻撃時間が α=0.016 の攻撃に比べ到着間隔が比較的一定であり，それらがマージした個数が多かったために安定性が大きく下がったものと推測される．

α=1.024 以降は，低レートの攻撃が認識され，観測攻撃数は短期間の攻撃のマージにより低下する一方で，攻撃時間の平均は上がっていく．安定性が全体的に上る要因は，高レートの攻撃における到着間隔の誤差に対して通信環境に起因するジッタが大きく影響する．低レートの攻撃では到着間隔が長いためにジッタの影響を受けづらいことが挙げられる．また，α の増加に伴い平均攻撃時間が長くなり 1 件の攻撃中に発生するパケット数が増加することにより，観測されるパケットの到着間隔の変動が平均安定性へ影響しにくい．観測攻撃数，平均攻撃時間，安定性は最終的に収束していき，攻撃検知数が限界に近づいていることがわかる．図 4 の右下のグラフにおいて DNS が α=16.384 から特徴的な形を示すのは，α が小さい場合に独立した攻撃とみなされていた長期間・短期間の低レート・高レートの攻撃がマージを繰り返したためと考えられる．

提案手法で DRDoS 攻撃を検知する場合には，可能な限り検知ルータにかかる負荷を軽減するために必要以上にパケットを待ち受ける時間を長く設定することを回避したい．しかし，待ち受ける時間を短時間にしてしまうと低レートの攻撃を検知できない．高レートの攻撃は低レートの攻撃の検知に要求される時間内で検知できるため，低レートの攻撃を基準とする方が良い．α = 1.024 以下の高レート攻撃の安定性の低下は 1.024 より大きな α を設定した時の低レート攻撃の検知による安定性の上昇により解決できるとすると，α=4.096，16.384，4.096，65.536 に対し小数点以下を切り上げし，5，17，5，66 秒を CharGen，DNS，NTP，SSDP について検知待ち受け時間の最大値として採用し，提案手法として実装する．

7. 提案手法

本研究で提案する方法は，OpenFlow を用いたリフレクター付近において外部からの攻撃クエリを OpenFlow ネットワーク内で詐称判定を行い，フィルタリングする．

本提案手法は，第 6 節で求めた閾値を元に，内部ネットワークに脆弱なリフレクター機器を持つような境界ルータで攻撃の検知を行い，中継ルータにおいて無関係な通信に干渉せずに，OpenFlow ネットワーク内で経路情報による詐称判定を行い，被害ホストとの正常な通信を除外したフィルタリングを行うことを実現する．想定するネットワーク構成を図 6 に示す．

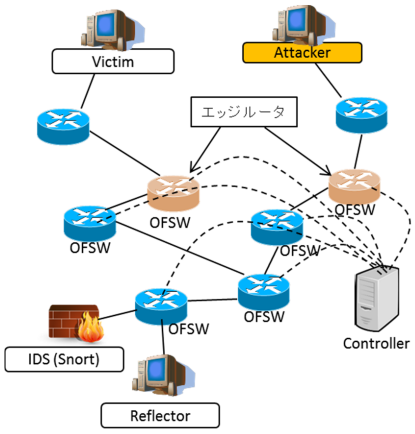

図 6　想定する OpenFlow ネットワーク

提案手法の動作は次の 2 段階で構成される．

(1) 境界ルータにおける攻撃クエリの検知
(2) OpenFlow ネットワーク内での攻撃トレース及びフィルタリング

ここで挙げた 2 項目について以下に説明する．

7.1 境界ルータにおける攻撃クエリの検知

図 6 に示すように Reflector を管轄するルータで，第 6 節で求めた閾値を用いて外部から来る攻撃クエリをサービスに応じた閾値を用いて検知する．閾値は第 1 パケットが観測されてから第 2 パケットが到着するまで期待される最大時間である．実際には第 2 パケット到着時間は閾値よりも短くなると予想される．

図 7 は第 6 節で求めた閾値を検知待ち受け時間の最大値とした場合に，各サービスにおける攻撃ごとの攻撃時間及び到着間隔の平均をプロットしたものである．閾値未満での到着間隔に攻撃が集中していることが分かる．予想される到着間隔は閾値よりも短時間であると予想される．

7.2 OpenFlow ネットワーク内での攻撃トレース及びフィルタリング

OpenFlow スイッチはレイヤ 4 層までのヘッダフィールドを元に制御を行う．提案手法ではヘッダに情報を添加する必要がある．ここでマーキングで扱うフィールドとしては Type of Service，Identification，Flags，Fragment Offset とする．Sanap ら [15] は，DRDoS 攻撃に対しトレースを行う手法の中で，書き換える IP ヘッダ内の領域に ToS，Identification を採用し，Durresi らの主張 [16] に基づいて正当であるとしている．Durresi らは ToS フィールドにつ

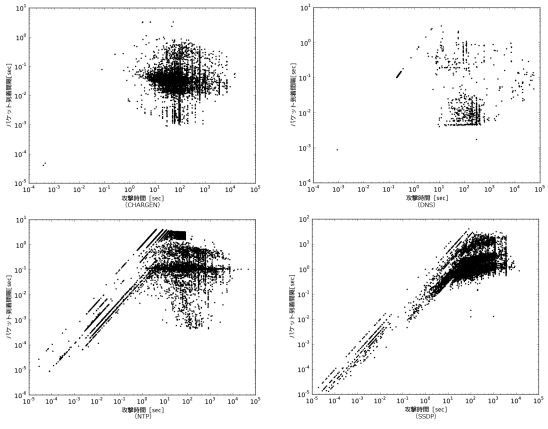

図 7　サービスごとの攻撃時間と到着間隔の関係

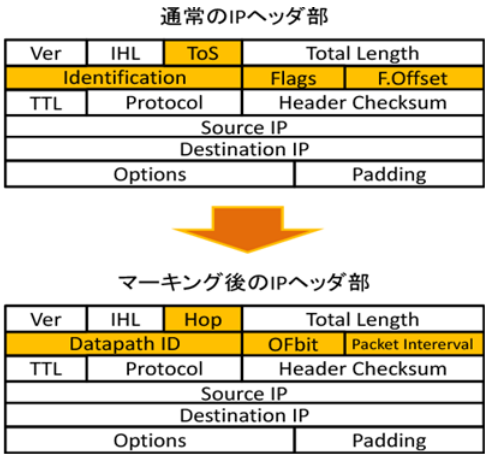

図 8　攻撃クエリにおける IP ヘッダ部のマーキング範囲

いて，特殊な制御に利用されるが利用率は低く，マーキングを行ってもルーティング機能には支障が出ないとしているが，その根拠となる研究成果 [17] は 2001 年に発表されたものであり，2002 年には ToS フィールドの改訂が RFC [18] により発行され，2012 年に Chaudhuri ら [19] は ToS フィールドを用いた DiffServ が重要視されて派生研究がおこなわれていることを主張している． また，2014 年に Tuyen ら [20] の研究でも同様のことが指摘されて，ToS フィールドの利用を控えているが，Flags フィールドの第 1 ビットを利用することで彼らの提案手法を確立している．本研究においても OpenFlow ネットワーク内の QoS 制御と互換性を持たせるために Flags フィールドの第 1 ビット（予約ビット）を利用する．この時，IP フラグメント機能についても同様に互換性が取れることに留意する．また，Flags フィールドの第 2，第 3 ビットを合わせて制御フィールドと定義する．Flags フィールド全体を OF ビットと定義する．

OpenFlow ネットワーク内に設置された OpenFlow スイッチは予約ビットが 1 である場合に，本手法によるパケットであると認識し，独自のルーティング手順を行う．詐称判定で用いられる情報は実際に通過した経路の情報であり，送信元 IP アドレスを詐称されたパケットの転送経路は被害ホストからリフレクターへの経路に完全に一致しないことが前提となる．よって，OpenFlow ネットワーク内で検知可能であるためには，ネットワーク内で正規パケットと詐称されたパケットのどちらか一方のみが通過するルータが存在することを前提とする．また，攻撃側及び被害側ネットワークが共に外部に存在することを前提とする．

IDS は検知後に，OpenFlow コントローラに小数点以下を切り捨てた 2 パケット間の到着間隔の 2 倍の値と，検知した攻撃クエリを 1 パケット送信する．コントローラは検知された攻撃クエリの IP ヘッダにマーキングを加えたパケット（図 8）をエッジルータに転送する．

各 OpenFlow スイッチは常時，以下の動作をするようフローエントリが登録されているものとし，項番の若い方を優先する．

(1) 予約ビット=0：通常通りルーティングを行う
(2) 制御フィールド=1：Identification=自身の Datapath-ID，制御フィールド=2 とし転送する．
(3) 制御フィールド=2：Packet-In を行う．
(4) 制御フィールド=3，TTL=ToS：Packet-In を行う．
(5) 制御フィールド=3：通常通りルーティングを行う．
(6) 制御フィールド=0：通常通りルーティングを行う．

OpenFlow コントローラは常時，Packet-In されたパケットに対し以下の動作をするようにし，昇順に高優先度とする．

(1) 予約ビット=0：通常の動作を行う
(2) IDS からの Packet-In：Fragment Offset=2 パケット間の間隔，制御フィールド=1，予約ビット=1，ToS=0 としてエッジルータに転送．IDS に接続されたスイッチに制御フィールド=0，予約ビット=1，プライオリティ=最大，Action：Packet-In としてフローエントリを追加．また，送信元 IP=IP ヘッダ内送信元 IP，受信ポート=UDP ヘッダ内受信ポート，Action：Drop としてフローエントリを追加．
(3) 制御フィールド=0：寿命=N，プライオリティ=最大，送信元 IP=IP ヘッダ内送信元 IP，Action：送信元 IP=ToS・Identification・Fragment Offset で示される送信元 IP としてフローエントリを追加．
(4) 制御フィールド=2：寿命=Fragment Offset，プライオリティ=最大，Action：通常ルーティングとして受信スイッチにフローエントリを追加する．エントリの寿命が切れ，Flow Removed により判明するエントリ合致パケット数が正数の時，ToS が 0 以外であれば制御フィールド=3 とし Datapath-ID=Identification のエッジルータへ転送，そうでなければドロップする．正数でなければ ToS=TTL として受信スイッチから転送．
(5) 制御フィールド=3：寿命=N，プライオリティ=最大，

送信元 IP=IP ヘッダ内送信元 IP，Action：送信元 IP=受信スイッチ IP としてフローエントリを追加し，ToS・Identification・Fragment Offset をゼロクリアし IP ヘッダ内送信元 IP を書き込む．制御フィールド=0，送信元 IP=受信スイッチ IP として転送．同一エントリが既に存在すれば，ドロップする．

OpenFlow スイッチでは，プライオリティを設定することができ，フィルタリングエントリより優先度の高いルールを適用することで例外を設定できる．攻撃経路ではない最もリフレクターに近いルータを通る被害ホストからのパケットは必ずマーキングが施されており，リフレクター側の境界ルータで攻撃クエリと違う処理を行うことが可能であるから，フィルタリング中も被害ホストとリフレクターとの通信は確保される．

正規通信をフィルタリングから除外するためのエントリ寿命 N は，攻撃時間が予想以上に長引く場合に備えて，ある程度の時間ごとに IDS でクエリパケットの受信を確認し次第，寿命の更新を行うようにする．ここで，中継するルータ及びリフレクター側ルータのエントリ寿命を同一にすることで，仕様上は誤動作が起きないことに留意する．また，本手法は極力 OpenFlow スイッチに動作を任せることでコントローラの負担を減らしている．

OpenFlow ネットワーク全体の端に存在する境界ルータで攻撃経路上にないところでマーキングを行う方法もあるが，各境界ルータからリフレクターに向けての経路が，ある場所でマージする可能性があり，その場所でマーキングを行うことで大量のトラフィックを処理する最境界ルータの負担を減らすだけでなく，冗長なマーキングルータを減らすことができる．

8. 負荷実験

8.1 実験の概要

本研究で提案する OpenFlow による攻撃回避を実装して，被害ホストとリフレクターの経路を確保した状態で被害ホストからリフレクターへの通信に対する負荷テストを行う．攻撃者からの攻撃クエリは hping3 [23] によって再現する．hping3 は，icmp プロトコルによって様々なパケットを生成する．本実験では hping3 を用いて UDP ストリームを発生させて，リフレクター付近の境界ルータでフィルタリングする．また，被害ホストとリフレクターとの通信は iperf [24] によって再現する．iperf は主にスループット計測で利用されるトラフィックを生成するツールである．

8.2 実験環境

実験の環境は，Oracle VM VirtualBox [25] を用いて仮想ネットワーク環境を構築した．仮想ネットワーク環境には Open vSwitch [26] を 7 台，Ryu SDN Framework [21] コントローラを 1 台，ホスト 3 台を設置し，それぞれが Mininet [27] で仮想的に接続されている．Ryu は NTT 研究所が開発したオープンソースの OpenFlow コントローラである．コントローラプログラムは Python によって記述され，OpenFlow の様々なバージョンに対応するため幅広いシステム開発が期待できる．Mininet とはスタンフォード大学で開発が行われているネットワークシミュレータであり，1 台の仮想マシン内で仮想スイッチや仮想ホストを構築することができ，ホスト内ではアプリケーションを実行できる．実験に使用した OS は Ubuntu 14.04 LTS，メモリを 4096MB，CPU は Intel Core i7-2600 3.40GHz を 2CPU 割り当てた．

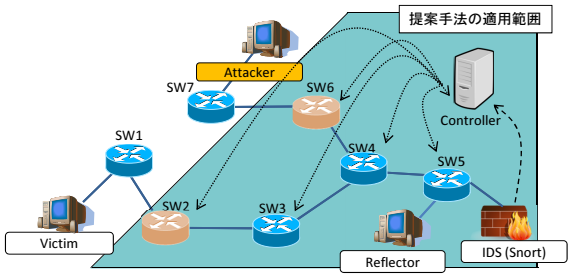

図 9 提案手法の実験構成図

8.3 実験内容

実験では，図 9 の Reflector において Victim からの UDP ストリームを iperf により生成させ，パケット到着間隔の変動を 1000 秒の間観測し，以下の条件で Open vSwitch にかかる負荷を評価する．

- CASE 1．攻撃及びマーキングを行わない場合
- CASE 2．SW3 及び SW5 でのマーキングを行う場合
- CASE 3．SW3 及び SW5 でのマーキングを行い，Attacker から hping3 で UDP ストリームを Reflector に向け送信し続ける場合

CASE 3 では hping3 にて通信帯域 1Mbps で 0.005 秒おきに 1 パケット送信し，NTP による通信を想定し第 5 節の結果から UDP ペイロード長を 36 バイトとした．

8.4 実験結果

各条件ごとの観測結果を図 10 に示す．CASE 1 及び CASE 2 ではジッタの変動は無いことがわかる．よって，中間地点によるマーキングでルーティングに影響が出るとは考えにくい．CASE 3 では他の条件に比べて 0.1 ミリ秒ほどジッタが増大しているが，実際に発生する Victim-Reflector 間の通信は想定した通信帯域より小さくなると予測されるため，特に問題にはならない．図 10 では他条件と比べても差は少なく，防御ネットワークに攻撃クエリが加わることで発生する負荷は微小なものであり，提案手法の

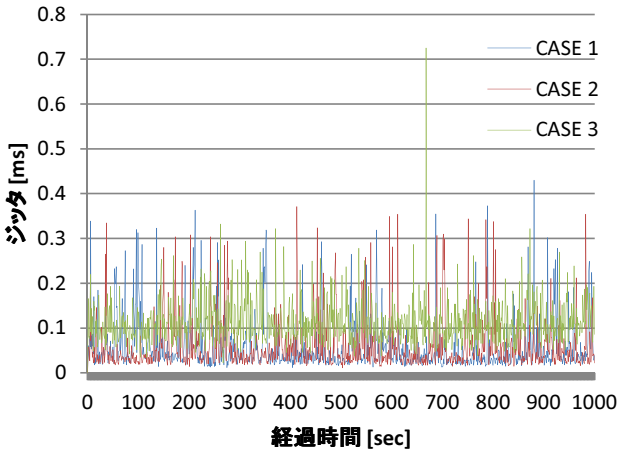

図 10　条件ごとの Victim-Reflector 通信の遅延変動

動作への影響は少ない．ただし，フィルタリングを行うことで相対的な負荷の増大がある．攻撃を受けているリフレクターが OpenFlow ネットワーク上で増加するにしたがって負荷が顕著になる可能性もある．実用化する際には，この点に配慮する必要がある．

9. まとめと今後の課題

本論文では，DRDoS 攻撃にリフレクターとなる機器が加担することを防ぐため，DRDoS 攻撃のクエリパケットの特徴を分析し，得られた結果から OpenFlow による防御ネットワーク内で攻撃クエリの詐称判定を行う手法を提案した．実際に観測した DRDoS 攻撃を分析することで，サービスごとの攻撃特性を見出すことができた．また，経路情報を元に詐称判定を行うことで送信元情報の詐称に対応できることを示した．

実験ではフィルタリング中に被害ホストとリフレクターの通信による遅延変動を観測することで，リフレクターが存在するネットワークの境界ルータで攻撃クエリをフィルタリングしても高負荷になりにくいことを示した．

リフレクターとして使われる機器はネットワーク管理者が発見できない場合が多い．リフレクター側ネットワークで事前に対策を取ることにより，被害ホストだけでなくリフレクターから被害ホストへのすべての経路で通信帯域が守られる．また，本研究で想定した限界攻撃間隔を更に細かく分割することで分析精度の向上が期待できる．

本手法は OpenFlow ネットワーク外に被害ホスト及び攻撃者がいることが前提である．ネットワーク内に攻撃者の存在が確認できる場合についてはイングレスフィルタリング [8] を OpenFlow ネットワークの境界で実装することで攻撃を未然に防ぐことが出来る．また，正常経路と悪性経路が OpenFlow ネットワークのエッジルータまで同一な経路となる可能性がある．この問題は，OpenFlow の標準化が進むに従い実装範囲が広がることを踏まえると憂慮すべきことではない．今後は実ネットワーク上で提案手法を実装して，悪性通信の検知率を測定する．

謝辞　本研究の一部は JSPS 科研費 JP16H02832 の助成を受けたものです．MWS データセットを提供頂いた情報処理学会 MWS コミュニティのメンバー諸氏に感謝いたします．

参考文献

[1] Akamai, “Akamai Releases Q3 2015 State Of The Internet - Security Report”, 入手先 ⟨https://www.akamai.com/us/en/about/news/press/2015-press/akamai-releases-third-quarter-2015-state-of-the-internet-security-report.jsp⟩, 2015.
[2] Impress, “リフレクション攻撃が増加し、DDoS 攻撃は小型化/2015 年のセキュリティを振り返る”, 入手先 ⟨http://internet.watch.impress.co.jp/docs/column/security/20160104_737474.html⟩, 2016.
[3] ASCII, “「家庭には乗っ取られる機器が大量に」アカマイ最新 DDoS 報告”, 入手先 ⟨http://ascii.jp/elem/000/001/045/1045383/⟩, 2015.
[4] shadowserver, “Open Simple Service Discovery Protocol (SSDP) Scanning Project”, 入手先 ⟨https://ssdpscan.shadowserver.org/⟩, 2016.
[5] Open Networking Foundation, “Open Networking Foundation”, 入手先 ⟨https://www.opennetworking.org/index.php⟩, 2016.
[6] Christian Rossow, “Amplification Hell: Revisiting Network Protocols for DDoS Abuse”, NDSS 2014 Symposium, February, 2014.
[7] Timm Bottger, et al., “DoS Amplification Attacks Protocol-Agnostic Detection of Service Abuse in Amplifier Networks”, 7th International Workshop on Traffic Monitoring and Analysis, Volume 9053 of the series Lecture Notes in Computer Science pp. 205–218, April, 2015.
[8] IETF, “RFC 2827 (Ingress Filtering)”, 入手先 ⟨https://www.ipa.go.jp/security/rfc/RFC3704EN.html⟩, 2004.
[9] IIJ, “DNS オープンリゾルバ問題”, 入手先 ⟨http://www.iij.ad.jp/company/development/report/iir/pdf/iir_vol21_internet.pdf⟩, 2013.
[10] IIJ, “DoS/DDoS 攻撃対策（1）～ISP における DDoS 対策の 現状と課題～”, 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/index.php⟩, 2013.
[11] キーマンズネット, “国内初の DDoS 犯人摘発例は高校 1 年生！　有名オンラインゲームが標的に”, 入手先 ⟨http://www.keyman.or.jp/at/30007392/⟩, 2014.
[12] 情報通信研究機構, “JGN2plus”, 入手先 ⟨http://www.jgn.nict.go.jp/⟩, 2015.
[13] MWS 組織委員会, “マルウェア対策研究人材育成ワークショップ 2015 （MWS2015）”, 入手先 ⟨http://www.iwsec.org/mws/2015/about.html⟩, 2015.
[14] 牧田 大佑, 吉岡 克成, 松本 勉, “DNS ハニーポットによる DNS アンプ攻撃の観測”, 情報処理学会論文誌, 55(9), pp.2021–2033, September, 2014.
[15] Yonghui Li, et al., “Traceback DRDoS Attacks”, Journal of Information and Computational Science, Vol.8 (1) pp.94–111, June, 2011.
[16] Arjan Durresi, et al., “Fast autonomous system traceback”, Journal of Network and Computer Applications, Vol.32, Issue 2, pp.448–454, March, 2009.
[17] Stefan Savage, et al., “Network Support for IP Traceback”, IEEE/ACM Transactions on networking, Vol.9,NO.3, pp.226-237, June, 2001.
[18] IETF, “RFC 3260”, 入手先 ⟨https://tools.ietf.org/html/rfc3260⟩, 2002.
[19] Sruti Gan Chaudhuri, et al., “Validation of a DiffServ Based QoS Model Implementation for Real-Time Traffic in a Test Bed”, NCC 2012, pp.1–5, February, 2012.
[20] Dang Van Tuyen, et al., “An Enhanced Deterministic Flow Marking Technique to Efficiently Support Detection of Network Spoofing Attacks”, ATC 2014, pp.446–451, October, 2014.
[21] Ryu SDN Framework Community, “Ryu SDN Framework”, 入手先 ⟨https://osrg.github.io/ryu/⟩, 2014.
[22] Manuel Palacin Mateo, “OpenFlow Switching Performance”, the Univercity Politecnico di Torino, July, 2009.
[23] Salvatore Sanfilippo, “hping”, 入手先 ⟨http://www.hping.org/⟩, 2006.
[24] Sourceforge.net, “iperf”, 入手先 ⟨http://iperf.sourceforge.net/⟩.
[25] Oracle, “Oracle VM VirtualBox”, 入手先 ⟨http://www.oracle.com/technetwork/jp/server-storage/virtualbox/overview/index.html⟩.
[26] Open vSwitch, “Open vSwitch”, 入手先 ⟨http://openvswitch.org/⟩.
[27] Octopress, “Mininet”, 入手先 ⟨http://mininet.org/⟩.
[28] 高田 雄太, 寺田 真敏, 村上 純一, 笠間 貴弘, 吉岡 克成, 畑田 充弘, “マルウェア対策のための研究用データセット–MWS Datasets 2016–”, 研究報告コンピュータセキュリティ, 2016-CSEC-74, CSEC, pp.1–8, July, 2016.