Computer Security Symposium 2014
22 - 24 October 2014

ネットワークノードの連携による緊急時の DDoS パケットの削除

鈴木 涼太　　森田 光

神奈川大学大学院工学研究科
221-8686 神奈川県横浜市神奈川区六角橋 3-27-1

あらまし

DDoS 攻撃に対してはアクセス数の流量制限の対策が一般的である。しかし、通常のアクセスと攻撃とを識別せずに一様の制限を行うので本来機能が著しく損なわれる。本稿では、攻撃によるトラフィックの増大に際して、不明、休眠または低頻度アクセスのクライアントのパケットを抑止するアプローチを取る。つまり、除外するクライアントのパケットを識別して削除する。具体例として、SDN(Software Defined Network) における方法を示す。

Emergency DDoS packet deletion by means of cooperation with network nodes

Ryota Suzuki　　Hikaru Morita

Graduate School of Engineering, Kanagawa University
3-27-1, Rokukakubashi, Kanagawa-ku, Yokohama, 221-8686, Japan

Abstract

Conventional measures for DDoS attacks have been generally using controls on flow data packets. However, since they don't classify the data packets as normal or attack, they impair transmission efficiency of networks. This paper takes an approach to suppress packets of strangers, and dormant or low-frequent clients. Thus, network switches delete attackers' packets. Furthermore, the authors show practical examples by using architecture of SDN – Software Defined Network.

1　はじめに

インターネットを通じ、ウェブサイトやデータベースに向けた DDoS 攻撃 (Distributed Denial of Service attack) が社会的な脅威として顕在化している。ショッピングサイトに輻輳障害を起こし、そのサービスを停止する例がたびたび報道されている。DDoS 攻撃を引き起こす大量の通信パケットが、多数の発信者から特定のターゲットの受信者に送られることを特徴とする。この対策としては、ターゲットの周辺のネット上のパケットに制限をかける（帯域制限）または負荷分散が行われる。しかし、攻撃ターゲット宛のパケットを破棄する対症療法は通常取られない。このため、抜本的な対策が無く、有効な対策が未だ存在しない。

ここでは、通常取られて来なかったパケットのヘッダ内容を活用する方法を提案する。この方法は以下の 3 点の部分からなる。スイッチが

データ転送を担うとし、

1. ターゲットへの Denial の認知
2. スイッチへのパケット制限依頼
3. スイッチのパケット・フィルタリング方法

この提案法を用いることにより、DDoS 攻撃のパケットを排除し、攻撃下でも通常のサービスの持続を期待できる。ここでは、1〜3 の要素をネットワーク上で実現するため、スイッチングハブやルーターなどの通信ノードを柔軟に制御する必要がある。そこで、制御を担う Control Plane とパケットを流す Date Plane に分離し、コントローラーとスイッチの 2 種の機器がパケットのフローを担うアーキテクチャーである SDN(Software Defined Network) における構成を前提に考察をする。

パケットのフィルタリングは、ブラックリストが良く用いられる。しかし、DDoS 攻撃では、送信者が多数なので、攻撃者の IP などをブラックリストに入れるアプローチでは即効性がなく、有効な手立てではなかった。ここでは、通常のサービスを受けるユーザには、特定の情報をパケットのヘッダに含ませ、その他のパケットと区別することで、フィルタによる識別性を向上させる。

本稿では、2 章に提案法を、3 章にシステム構成および制御の流れを、4 章に考察を示し、5 でまとめる。

2 提案法

2.1 記法の定義

- r_i：クライアントごとにランダムに生成する CIM の要素
- CIM_i：CIM の集合で、$\{r_0, r_1, \ldots, r_i\}$ である。

2.2 攻撃対象となったノードへの Denial の認知

攻撃対象となるノード（ここでは、サーバーとする）が攻撃を受信し、サービスが停止する危険性があると判断する材料として、

- コントローラーが、制御下のスイッチにおいて平時観測されるトラフィックを著しく上回るトラフィックを観測した場合
- サーバーの OS が、アプリケーションの処理容量を越えるトラフィックを観測した場合

  例として、HTTP Get Request などの特定のデータフレームが平時の数百倍のパケットを観測した場合など。

これらの手段により DDoS 攻撃によるサービスの停止と判定する。

2.3 スイッチへのパケット制限依頼

各緊急パケットの送信、及びパケットの制限依頼は Control Plane の通信路によって送受信される。2.2 節により対象ノードがサービスを持続させることが不可能であると判断できた場合には、サーバーはネットワークを構成するスイッチを制御するコントローラーに対し、緊急パケットを送信する。緊急パケットを受け取ったコントローラーは配下のスイッチに対して、パケットの制限依頼を送信する。スイッチは受信したのちに直ちにフィルタリングルール（2.4 節）に従い、パケットの棄却・転送を行う。

2.4 スイッチのパケット・フィルタリング方法

DDoS 攻撃をサーバーが観測し、パケットのフィルタリング要請がコントローラーに対し依頼され、コントローラーがスイッチに対してパケットの棄却命令を行う。棄却は以下のパケットのフィルタリング方式を用いスイッチ上で行われる。

パケットのフィルタリングを行う手段として、従来のブラックリスト方式によるフィルタリングに加え、クラス識別マテリアル（Class Identification Materials、以下 CIM) を導入する。

CIM はサーバーからクライアントに対して、事前にアプリケーション通信時に配布される r_i を含む識別子である。サーバーは、CIM を分

類するルールを予め設定し、コントローラーに対し事前に通告、コントローラーはフローエントリとしてスイッチに書き込む。また、コントローラー経由で、CIM を構成する r_i は、各スイッチに配布されるものとする。

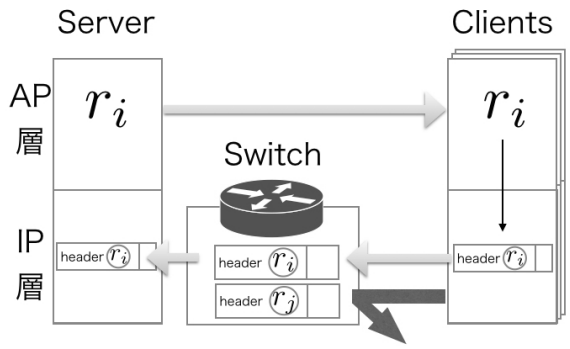

図 1: CIM の要素 r_i の配送 $(i > j \geq -1)$

図1では、サーバーに対し、スイッチを経由しクライアントが接続したときに配布される CIM の要素の配送手順を示す。サーバーからクライアントに対して r_i を配布する時はアプリケーション層を経由し、パケットの送信時には IP 層を通じて転送される。

クライアントは IP プロトコルと接続アプリケーションを用いて接続を行う。接続アプリケーションでは、初回接続時に配布される r_i を受信し、次回通信時に IP パケット内のオプションフィールドに r_i を挿入し、接続先のサーバーに対して転送を行う。通信経路上に存在するコントローラー制御下のスイッチは IP パケットオプションフィールドの r を検査する。

また、CIM においてはいくつかの集合を定義することが可能である。

- 新規アクセス優先

  一定期間はサーバーが任意に設定する。

- メンバーシップ集合

  この集合では、会員制として優先すべきクライアントを設定する。

この 2 種類の CIM の集合を利用し、それぞれの分類に優先度を付与することで緊急時に転送を許可するクライアントの優先度管理を柔軟に行う。4 章で CIM の集合の管理法について考察を行う。

2.4.1 CIM の集合 CIM_i の生成および配送

以下の手順により、CIM の集合を生成する。

1. $CIM_{-1} = \varnothing$
2. $i = 0$
3. $CIM_i = CIM_{i-1} \bigcup \{r_i\}$
4. $i = i + 1$
5. goto 3

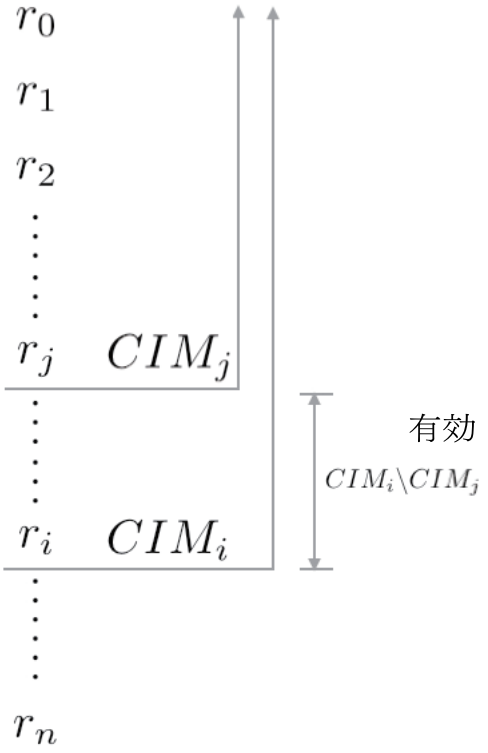

図 2: CIM の時間制御

図2では、サーバーより任意に設定する j から i までの期間にランダムな値 r_i を生成する。CIM 集合 $(CIM_i \backslash CIM_j = \{r_{j+1}, r_{j+2}, \ldots, r_{i-1}, r_i\})$ を有効とする。また、j 以下の集合 CIM_j 及び他の r は棄却対象とする。

以下が CIM の配送からフィルタリングまでの手順である。

1. クライアントのプログラムはサーバーからの r_i を入手し、次回送信時に、IP パケットのオプションフィールドに挿入する。
2. サーバーは緊急時に備え、$i > j \geq -1$ なる j を決めておく。

   ここで、$r \in CIM_j$ の r を IP パケットのオプションフィールドに挿入するパケットを排除するためである。
3. 緊急時には、$r \in CIM_i \backslash CIM_j$ の r を IP パケットのオプションフィールドに挿入するパケットのみを対象にサービスを継続する。

3 SDN における制御の流れ

3.1 システム構成図

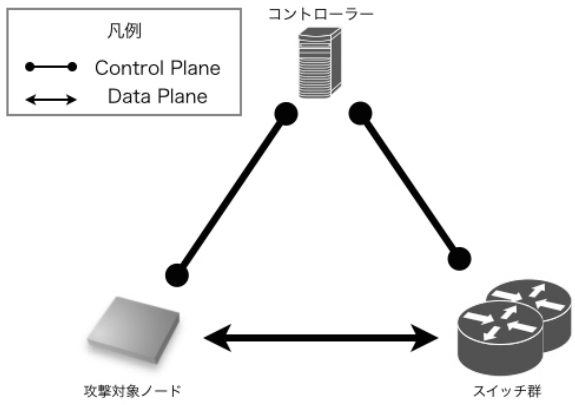

図 3: システム構成図

図 3 では、ネットワークにおいてコントローラー・スイッチ群・攻撃対象ノードの 3 者によって構成される。コントローラーは構成されるスイッチの経路制御・転送制御を管理し、スイッチに対して任意の制御情報を書き込む（Control Plane) 役割を担う。スイッチは複数個存在し、コントローラーから与えられた制御情報に従ってデータの転送 (Data Plane) を行う。攻撃対象ノードは、何らかのサービスの提供を行うサーバー等である。

提案システムでは、SDN の一形態である OpenFlow のモデルに従い、ネットワークにおける L2〜L4 のデータフレームの転送を担うスイッチ（本稿では、L2 におけるフローテーブルを想定する）、コントローラーはスイッチに対してフローエントリ（制御情報に相当）を書き込む。

3.2 シーケンス

攻撃対象ノードのサーバーとクライアント間で以下の事前準備を行う。図 3 では、クライアントはスイッチ群を経由し、サーバーへアクセスを行う。

1. サーバーは CIM を分類に従いアプリケーションソフトを経由し配布する。
2. クライアントは配布されたマテリアルをアプリケーションにおいて受信する。
3. サーバーはコントローラーに対して、CIM を報告するとともに分類ルールを渡す。
4. クライアントは CIM の要素 r_i を IP パケットのオプションフィールドにセットし、IP プロトコルでの通信を行う。
5. サーバーは受信した r_i を CIM テーブルに保存する。

被攻撃時は以下の流れとなる。

1. DDoS 攻撃を 2.2 節の方法により観測し、判断する。
2. 判断に基づき、ただちにサーバーが Control Plane を通じコントローラーに対し緊急時対応であることを通知する。
3. 緊急パケットを受信したコントローラーはただちに配下のスイッチに対して Control Plane を通じ、パケットの棄却依頼となるフローエントリーを書き込む。
4. スイッチはコントローラーから配布された $CIM_i \backslash CIM_j$ に包含される有効な r_i と、受信されたパケットの r が同値であれば転送を行い、それ以外は棄却する。

4　考察

4.1　CIM の集合

4.1.1　新規アクセス優先

図 2 では、時間経過ごとに r_0 から r_i までの CIM の要素を発行している。

サーバーなどのノードに対してクライアントがアクセスした時間によってアクセス管理を行う。サーバーは 0 から i までの期間にランダムな CIM の要素 r_i の発行を行い、クライアントに対して配布（図 1）を行う。サーバーより任意の間隔を設定する j から i までの期間に発行された CIM 集合（$CIM_i \backslash CIM_j = \{r_{j+1}, r_{j+2}, \ldots, r_{i-1}, r_i$ を有効とする。j までに発行された CIM_j に含まれる要素はすべて無効とし、輻輳時・被攻撃時にはこの集合にあるパケットのみスイッチにおいて転送する。

また、攻撃ノード宛で CIM_j のみならず、未知の r や、r が記述されていないパケットも削除する。

4.1.2　メンバーシップ制御

課金制のサービスを行うサーバーなどでは、優先度を階層的に分割して CIM を管理することで、従来から行われている QoS（Quality of Service) 制御を導入することが可能である。被攻撃時、輻輳障害時においては、優先度の階層に応じ、優遇すべきクライアントのパケットの転送を行う。階層の種別を二つ設定する。

- 会員として優遇すべき階層
- 過去継続的に利用を行っているクライアントであり、継続期間に応じて分割する階層

また、時間的な制御 4.1.1 節と併用し、指定した有効期間のみ優先する階層とするクライアントとしての CIM を発行し、更新がなければ、通常のアカウントグループに戻すといった処理が考えられる。目的に応じて CIM_j は非優先、$CIM_i \backslash CIM_j$ を優先と分離すればよい。

4.2　拡張性

エリックチェンら [1] は、ファイアウォールを DDoS 攻撃発信者付近のネットワークに徐々に広げながら攻撃元を隔離する方式を提案している。今回の提案でも、複数のネットワークに拡張することによって、攻撃パケットを隔離するネットワークの範囲を拡大することが可能である。

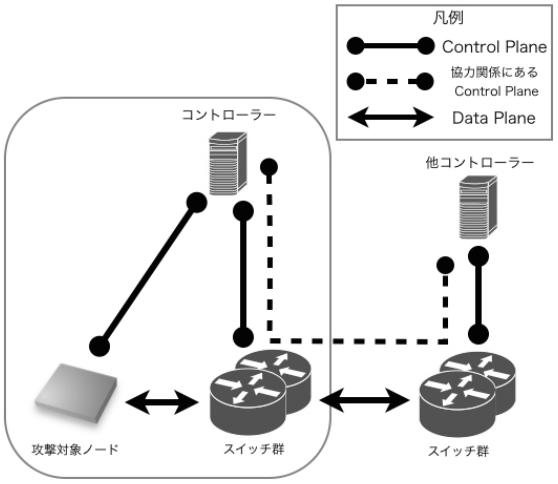

図 4: 隣接するネットワークとの連携

図 4 では、枠内を一つのコントローラーが管理しているネットワーク群とし、枠外を管理外のコントローラーにより管理されるネットワーク群とする。枠内のコントローラーは管理外のコントローラーと協力関係にあり、Control Plane によって協力依頼が可能な通信路によって接続されている。その為、コントローラーによる枠内の制御を外側に対して拡大することが可能である。

被攻撃時には 1 つのコントローラーが管理するネットワークの隣接ネットワークを経由し、攻撃パケットが送信されてくることが想定される。その場合に、隣接ネットワークが同様に Control Plane と Data Plane によって管理されるネットワークだとした時、緊急パケットを隣接ネットワークの管理を行うコントローラーに対し送出することで、パケットが除外されるエリアを広げていく。また、従来方式の隣接する IP ネットワークに対しては、手動にファイアウォール装

置を設定するなどよってパケットの制限を行う必要がある。この時、隣接ネットワークに対しては、緊急パケットのほかに転送を許可するパケットの CIM テーブルを知らせる必要があるため、隣接ネットワークの管理主体が信頼されたものでなければならない。

しかしながら、提案法における CIM の生成法では、CIM テーブルの共有をする際の問題点を内包している。この CIM を複数のネットワークで識別するためには、すべてのスイッチが CIM テーブルを共有しなければならない。その為、テーブル転送時の時間的なロスや、各スイッチにおける保存領域などのリソースの占有などが発生し、CIM の共有方法・管理が問題となる。そこで、4.3 節の方法を用い、CIM の生成を行うことによって、容量の削減を図る。

4.3 ハッシュチェインを用いた効率的な CIM 生成方法

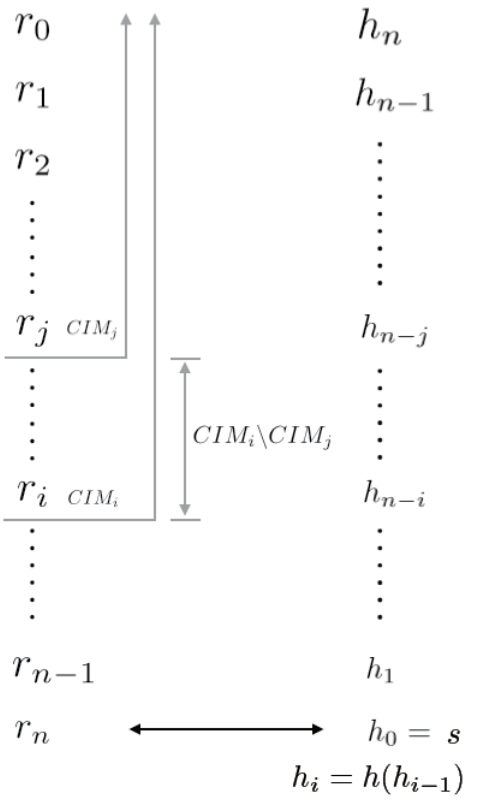

図 5: ハッシュチェインによる CIM 集合の圧縮

図 5 では、ハッシュチェインを用い、生成した r_0 から r_i までの CIM の要素にそれぞれ対応するハッシュ値を逐次ハッシュチェインを行い生成する。この時、r_i から r_0 までの順番に対応するハッシュ値を生成し、それぞれの要素に対応付ける。生成された h_{n-j} のハッシュ値において、$CIM_i \backslash CIM_j$ の CIM 集合が内包される。緊急時はこのハッシュ値を用い、クライアントからのパケットの r_i をサーバー側でハッシュ化、比較を行いパケットの転送・棄却を行う。そのため、先頭のハッシュ値のみを保存していればよいので、情報量が削減される。

これにより、拡張したネットワーク（4.2 節）においては、CIM テーブルをすべて転送するという作業が必要ないため、各スイッチに転送量が少ないために時間を要さず、スイッチ上ではハッシュ化演算を行うだけで CIM_i/CIM_j に包含される各 r_i に対応する値が得られる。その為にメモリーを占有する情報量が削減される。また、照合時には各スイッチでハッシュ値の展開を行いながら照合、または各ハッシュ値を計算しながら照合する手段を各スイッチの能力に応じて選択する。

4.3.1 手順

- ハッシュチェインの生成
  1. $h_0 = s$ （s はランダムに選択されたシード）
  2. $k = 0$
  3. $h_{k+1} = h(h_k)$
  4. if $k == n - 1$ return （ただし、n は適当に与えた上限）
  5. $k = k + 1$
  6. goto 3
- CIM 要素の r_i への適用
  1. $r_0 = h_n$
  2. $i = 1$
  3. $r_i = h_{n-i}$
  4. if $i == n$ goto 7

5. $i = i + 1$
6. goto 3
7. 他のハッシュチェインを生成する方法に移行する。

- $CIM_i \setminus CIM_j$ の要素にある判定

1. r_i と入手したパケットの r とを比較し、一致すれば、棄却しないこととし、return
2. if $i == j + 1$ そのパケットを棄却することとし、return
3. $r_{i-1} = h(r_i)$
4. $i = i - 1$
5. goto 1

4.4 今後の課題

4.4.1 本プロシジャーをターゲットとする攻撃

本稿で提案する方式には、次の 2 つの課題がある。

- CIM の再配布

  1 点目に、優先度の高い CIM を持つクライアントが、CIM を持たない他のクライアントに対して再配布し、その CIM をパケットに設定した上で DDoS 攻撃を行える。この場合、送信者の IP アドレスを CIM と同時に検査することで攻撃パケットを棄却することが可能である。しかし、SSH（Secure SHell) などの手段による遠隔操作または VPN による経路制御により、CIM と IP アドレス両方を持つクライアントによる代理攻撃があった場合、スイッチではパケットの転送を許可してしまうため、本提案によって攻撃パケットを棄却することは不可能である。また、CIM を送信者 IP に関連付ける方法もあるが、検査に計算量が増大する。

- 攻撃対象ノードの乗っ取りによるコントローラーへの任意の緊急通知

  サーバーなどの攻撃対象ノードを踏み台とし、乗っ取ることによってコントローラーに対して任意に緊急通知を行うことができる。そのため、攻撃者により緊急通知が行われた場合、スイッチの構成するネットワークにおいてパケットの制限が機能するため、提案法のシステムの正常な体系を崩す危険がある。

4.4.2 実装レベルでの SDN の制御にむけての攻撃

SDN の一形態の実装として現在 OpenFlow があり、事実上の標準として普及している。

OpenFlow の Control Plane においては、OpenFlow チャネルと呼ばれる TCP セグメントによってフローテーブルの配布、またはスイッチのアクションを行っている。しかしながら、OpenFlow1.1 より SSL/TLS による保護は仕様上必須でなくなったため、実装上の OpenFlow チャネルで Control Plane の保護が行われていない場合、攻撃者が OpenFlow チャネルを盗聴することにより、CIM の漏洩や任意のフローテーブルの更新が行われる危険がある。

4.4.3 実装上の厳密性

提案方式の実装については、独自のコントローラーが開発可能な OpenFlow プログラミングフレームワーク上において実装することが想定されるが、現段階においては、提案方式は未実装である為、実際に DDoS 攻撃を行った際の検知・制限依頼・パケットフィルタリングまでの観測が行われていない。

現在存在する OpenFlow ver1.4 では、スイッチの挙動として、フローエントリにより送信元・宛先 L2〜L4 アドレス・VLAN アドレスに従い、転送・破棄・ヘッダ書き換えの機能を実施する動作が定義されているが、現段階の OpenFlow において実装する際は、コントローラーに対し、流入パケットのパケットインをコントローラーに対し行い、検査をした後にコントローラー上で棄却する方式に変更せざるをえず、スループットの低下が予測される。提案方式を完全に実装

するためには、特定のデータユニットのヘッダ部分を検査し、それによりフローエントリのアクションを実施する機能が必要である。しかしながら、現在の OpenFlow の仕様において、パケットのヘッダ部分によってアクションを実施する機能はフローエントリでは定義されておらず、今後の OpenFlow のバージョンにおいての実装が必要である。

5　まとめ

本稿における提案では、通常のアクセスと攻撃を含むそれ以外のアクセスを、IP パケットに CIM を付加することで、攻撃対象ノードの物理的なリソースに依存せず、正規のクライアントからのパケットの犠牲を最小限度に抑制するパケット制限方式を示した。特に、CIM の導入により、IP レイヤーにおいて階層的優先度の導入や時間管理による制御などの柔軟に個別のクライアントに対してサービスを持続的な提供を可能とした。また、提案法を拡張することにより、一部のネットワークのみでの運用のみならず、隣接するネットワークに向けて拡張することで攻撃パケットに対するパケットフィルタリングの領域を拡大する考察を行った。

参考文献

[1] エリック チェン, 柏大, 冨士仁, 米沢明憲, “Moving Firewall における DDoS 攻撃対策システムの評価,” 信学技報, NS2002-121, IN2002-65, CS2002-76(2002-09)