一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
NS2020-151(2021-03)

探索フェーズにおけるCrossfire攻撃ホストの特定

仲原　愛美†　　上山　憲昭†

† 福岡大学工学部電子情報工学科
hskip1zw 〒 814-0180 福岡市城南区七隈 8-19-1
E-mail: †{tl171250,kamiyama}@fukuoka-u.ac.jp

あらまし　DDoS (distributed denial of service) と呼ばれる，インターネットに接続した多数のホストを利用し特定のサーバに大量のパケットを送信しサーバを意図的に機能不全とする攻撃が頻繁に発生している．しかし近年，特定のサーバではなく複数のターゲットサーバを含むターゲットエリアに至るネットワークのリンクを高負荷とすることでターゲットサーバにパケットが到達不可となりターゲットエリア内のホストを通信不能状態とするCrossfire attack (CFA) の問題が指摘されている．CFAでは攻撃者は攻撃に先立ち，高負荷とするターゲットリンクを選定するために，大量のボットからターゲットエリア内にサーバにtracerouteを行う特徴がある．そこでこのようなCFAの探索フェーズにおいて，tracerouteの発生間隔が閾値以下となるtraceroute送信ホストをボットとして検知し，tracerouteのパケットを規制することが考えられるが，本条件を満たす正常ホストもボットとして誤検知される問題がある．そこで本稿ではtracerouteの受信間隔に対しても閾値を設け，閾値以内の時間間隔でtracerouteを受けたホストをボットの探索フェーズにおけるターゲットサーバとして検知・リスト化し，リスト化されたサーバに対するtracerouteの間隔が第二の閾値以下となったホストのみをボットとして検知することを提案する．また一方の閾値からもう一方の閾値を最適設計するために，正常ホストの誤検知確率を解析的に導出する．そして提案方式の有効性を計算機シミュレーションにより明らかにする．

キーワード　crossfire attack，traceroute，最適検知

Detecting Crossfire Attack Hosts in Preparation Phase

Manami NAKAHARA† and Noriaki KAMIYAMA†

† Faculty of Engineering, Fukuoka University
8-19-1, Nanakuma, Jonan-ku Fukuoka, 814-0180 Japan
E-mail: †{tl171250,kamiyama}@fukuoka-u.ac.jp

Abstract　Attacks called DDoS (distributed denial of service) that use a large number of hosts connected to the Internet to send a large number of packets to a specific server and intentionally cause the server to malfunction frequently occur. However, in recent years, due to the heavy load on the network link to the target area including multiple target servers instead of a specific server, packets cannot reach the target server, and the hosts in the target area cannot communicate. This attack is called a crossfire attack (CFA). In the CFA, an attacker has a feature of executing traceroute from a large number of bots to a server in the target area in order to select the target links with a high load prior to the attack. To cope with the CFA, it might be effective to filter all the traceroute packets sent within a time interval less than the threshold in the phase of searching the target links in the CFA. However, legitimate hosts will be also detected as bots. To prevent this issue, this paper proposes a method to detect the tharget servers which receive traceroute packets within a time interval less than the first threshold and classfiy these identified hosts as the target servers in the searching phase. This paper also proposes to detect the bots which send traceroute packets to the target servers identified within a time interval less than the second threshold. We also derive the formula giving the false identification probability of legitimate hosts and optimally desigin one threshold from the other threshold based on the derived formula. Using a computer simulation, we evaluate the effectiveness of the proposed method.

Key words　crossfire attack，traceroute，optimum detection

1. はじめに

近年，多数のボットから Web サーバ等の攻撃ターゲットサーバに対し大量のパケットを送付することで，ターゲットサーバを機能不全とする DDoS (distributed denial of service) 攻撃が日常的に発生している．特定のホストに対する DDoS 攻撃に対しては，サーバとネットワークとの接続部分にファイアウォールを設置し攻撃パケットを規制したり，ネットワーク事業者が攻撃を受けている Prefix を BGP ルータで規制するなどの対処法が用いられている [4]．しかし近年，特定のサーバではなく複数のターゲットサーバを含むターゲットエリアに至るネットワークのリンクを高負荷とすることで，ターゲットエリア内のホストを通信不能状態とする Crossfire Attack (CFA) の問題が指摘されている [8]．CFA では攻撃対象がリンクのためサーバ側での検知が困難であり，また大量のボットからターゲットリンクに対し，各々は少量のデータのみを流すため CFA を行うボットの特定が困難であり，CFA の効率的・効果的な検知・防御方式の実現が課題である．

CFA では攻撃対象リンクを選定するために攻撃に先立ち，攻撃に用いる大量のボットとターゲットエリア内の複数のサーバに対し traceroute を行うことで，ターゲットエリアと外部のネットワークとをリンク繋ぐを探索する．このような探索フェーズの後，ボットから送信したトラヒックが選定したターゲットリンクを通過するような，ターゲットエリアの周辺に存在する複数のデコイサーバを選択する．そして攻撃フェーズにおいては，多数のボットから複数のデコイサーバに対しトラヒックを流すことでターゲットリンクを高負荷とする [8]．

2. 節で述べるように，これまでに探索フェーズもしくは攻撃フェーズにおいて，CFA を行うボットを特定しパケットを規制することで CFA を防御する様々な方式が検討されているが，防御効果を高めるには CFA の発生を未然に防ぐ必要があり，探索フェーズで検知・防御を行うことが望ましい．しかし正常ホストに対する影響を考慮した方式は見られない．そこで本稿では CFA の探索フェーズにおいて，正常ホストの巻き添えを考慮しつつボットを高精度に特定しその送信パケットを規制することで，CFA の攻撃に用いるボットを効果的に特定する方式を提案する．

上述したように CFA の検知はサーバ側での検知が困難であるため，提案方式では攻撃に先立ち発生する traceroute の発生間隔に閾値を設け，発生間隔をもとに CFA の検知・防御を行う．traceroute の発生間隔を発ホスト側だけでなく着サーバ側も考慮することで，正常ホストの誤検知を抑制しつつボットの検知を行うことが可能である．また正常ホストの誤検知確率を定式化することで，任意に与えた誤検知許容上限を満たす範囲で，ボットの検知確率を最大化する最適閾値設計法を提案する．以下，2. 節では関連研究について，3. 節では CFA について述べ， 4. 節で提案検知方式を述べる．5. 節では提案検知方式の評価について説明し，6. 節で最適閾値設定法について示す．最後に 7. 節でまとめを述べる．

2. 関連研究

CFA の検知や防御技術に関しこれまでに提案されている方式は，主に，攻撃フェーズにおける技術と，探索フェーズにおける技術のいずれかに分類できる．攻撃フェーズにおける技術として，まず CFA の発生を検知するものがある [13] [19]．Narayanadoss らはターゲットリンクのトラヒック量を測定し，ANN, CNN, LSTM などの深層学習を用いて CFA の発生リンクを検知する方式を提案している [13]．また Xue らはルータ間で E2E もしくはホップ間のアクティブ測定を行い，CFA の発生リンクを検知する方式を提案している [19]．ただしこれらの方式では CFA を行っている攻撃フローは特定できないため，CFA の防御は行えない．

さらに攻撃フェーズにおける技術として，攻撃を行っているフローやボットを特定し，SDN を用いて迂回制御等で攻撃を防御する方式が提案されている [3] [14] [15] [16] [17] [20]．例えば Zheng らは Prefix を反復的に絞り込むことで攻撃に加担しているフローを特定する方式を提案している [20]．Wang らは可用帯域やパケット損失率をルータで測定して輻輳リンクを検知し，迂回制御・流入規制により CFA を防御する方式を提案している [17]．また Belabed や Wang らは各ノードにてトラヒック量を測定し，レートの高いフローをリスト化し，SDN コントローラにリストを送り，コントローラはリストアップされたフローの経路を迂回することで，高負荷リンクを経由するフローの経路を切り替えてロードバランスを行うことを提案している [3] [16]．さらに Rafique らは各リンクのトラヒック量を測定し，輻輳の発生が検知されたら，そのリンクを流れるフローを他の経路に SDN を用いて迂回することを [14]，Rasool らは ANN を用いた深層機械学習により，フロー単位の CFA の検知と防御法を提案している [15]．しかしこれらの方式では攻撃が発生してから検知・防御を行うので，CFA を未然に防ぐことはできない．

CFA を未然に防ぐには攻撃者が攻撃ターゲットとするリンクやデコイサーバを探索する探索フェーズにおいて検知や防御を行う必要があるが，探索フェーズにおける方式として，Hirayama らは攻撃者がネットワークマップを作製するために行う Traceroute の各ルータでの検知回数をログとして収集し，その変化パタンから CFA の探索イベントの発生を検知することを提案している [7]．しかしイベントの発生の検知を目的としており，探索を行っているボットを特定し防御することはできない．攻撃者の探索活動を妨害する技術として，Aydeger らは SDN を用いてスイッチが ICMP パケット情報をコントローラに送り，コントローラがターゲットリンクを推定し，そこを経由しない経路を設定することを提案している [1]．また Gillani らは VN を動的に構築することで，CFA のトポロジ推定を無効化することを提案している [5]．また Aydeger らや Kim らは，traceroute パケットを仮想ネットワークで構築した偽のトポロジを経由させることで，攻撃者に誤ったトポロジ情報を学習させることを提案している [2] [10]．しかしこれらの方式では，正常ホストの traceroute も偽の情報を得ることになる問題がある．

3. Crossfire Attack (CFA)

3.1 CFA の攻撃手法

ノードまでの経路情報を取得するツールである traceroute を実行することで，実行したノードから指定したノードまでの経路 (経由するルータ) のリストを得ることが可能である．CFA では攻撃者が攻撃対象リンクを選定するために，探索フェーズにおいて図 1 に示すように，多数のボットからターゲットエリア内の公開サーバや，ターゲットエリア周辺のデコイサーバに対し，traceroute を行うことで攻撃リンクを選定する．CFA における攻撃者の流れは以下のようになる．

（1） 多数のボットからターゲットエリア内の多数の公開サーバ (Web サーバ等) に対し traceroute を行うことでターゲットエリア内のホスト宛てのフローの多くが経由する少数のリンク (攻撃対象リンク) を発見

（2） 多数のボットからターゲットエリアの周囲に存在する

多数の公開サーバ (デコイサーバ) に対し traceroute を行い，攻撃対象リンクをフローが経由するボット・デコイサーバ組を選定

（3） 選定したボット・デコイサーバ組に，検知されない程度の少量のトラフィックを発生 (通常の HTTP request/response など)

上記手順のうち (1)(2) が探索フェーズ，(3) が攻撃フェーズとなる．攻撃フェーズにおいてターゲットリンクが高負荷となり，ターゲットエリア内のホストにパケットが到達不可となりサービス妨害が成立する．

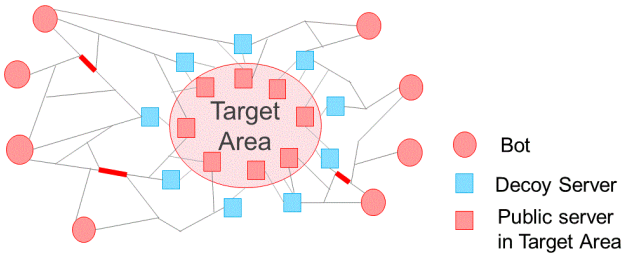

図 1　Crossfire Attack

3.2　検知の困難さ

CFA では，以下のような理由から検知・防御が困難である．

- 従来の DDoS の検知方法の多くは攻撃対象サーバで検知を行うが，CFA ではターゲットエリア内のサーバが直接攻撃を受けるわけではないため，被攻撃サーバでの検知が困難
- 攻撃リンクはターゲットエリアとは異なる AS(Autonomous System) に属しているため，CFA を防御するには AS 間の協力が必要
- 各ボットが各デコイサーバに対して送付するトラフィック量は少量で，かつ正常ユーザと区別がつかないため，トラフィック量やパケットのペイロードによる識別が困難

3.3　CFA の特徴

CFA では攻撃に先立ち，攻撃者は攻撃に用いるボット・デコイサーバ組を選定するため，大量のボットとターゲットエリア内のサーバ/デコイサーバ間に traceroute が発生する．この traceroute はネットワークのリンクが更新される前に選定を終える必要があるため，短い時間内に連続して行われる．また多くの場合，攻撃者はボットマーケット (PPI：pay-per install) を使用するが，コストは使用するボットの数に比例するため，1 つのホストが多数のターゲット/デコイサーバに対し traceroute を実施することになる．この特徴をもとに traceroute の発生間隔に基づく検知方式を提案する．

4.　提案 CFA 検知方式

4.1　簡易方式

攻撃に先立ち，ボット・デコイサーバ組の選定のために短い時間内に連続して大量のボット・ターゲットサーバ間に traceroute が発生する特徴をもとに，発ホスト側検知閾値時間 (T_s) 内に連続して，traceroute を行ったホストをボットとして検知しブラックリスト化する簡易方式を考える．しかしこの方式では，正常なホストも T_s 以内に連続して traceroute を行う可能性が高く，正常ホストがボットとして誤検知される問題がある．

4.2　提案方式

上記の簡易方式における正常ホストの誤検知を抑制するために改善した方式を提案する．ターゲットサーバはボットから短い時間に連続して traceroute を受ける特徴がある．その特徴をもとに，まず着サーバ側検知閾値時間 (T_d) 内に連続して traceroute を受けたサーバをターゲットサーバリスト化し，発ホスト側検知閾値時間 (T_s) 内に連続して，ターゲットサーバリストに含まれているサーバに対して traceroute を行ったホストをボットとして検知しブラックリスト化する．traceroute 間隔は，任意のルータの発もしくは着 IP アドレスをキーとしたハッシュテーブルを用いて traceroute 発生時刻を記録することで検査する．ターゲットサーバとボットの双方で検知を行うことで，正常ホストの誤検知を抑制しつつボットの検知精度の向上が期待できる．

5.　提案方式の性能評価

5.1　評価条件

単一ルータにおける検知性能を評価する．正常ホストの traceroute の発生時間間隔は，WIDE のバックボーンネットワークで取得された公開パケットトレースデータ [18] から作成した発生間隔分布を用いる．まずパケットをキャプチャした pcap ファイルから traceroute の戻りパケット (icmp time exceeded, icmptype: 11) を抽出するが，抽出した ICMP Type 11 のパケットには TTL が 0 となり返送されたものも含まれる．traceroute の戻りパケットには経路上の複数のルータから返信されたものから構成されることから，プローブパケットを宛先とするパケット群のなかから，発 IP アドレスの異なり数が 2 以上のものだけを traceroute の戻りパケットとして抽出する．そして ICMP Type11 パケットのデータ部分に含まれる traceroute のプローブパケットの発着 IP アドレスが同じパケット群を 1 回の traceroute で生成された戻りパケット群と定義し，発 IP アドレスごとに traceroute の発生間隔を抽出する．ただし同一発着 IP のペアに対し，発生間隔が 5 秒以内のものは 1 回の traceroute と定義し，同じ発 IP でも着 IP が異なれば，発生間隔が 5 秒以内であっても異なる traceroute として認識する．作成された，同一ホストの traceroute の発生間隔の累積分布を図 2 に示す．平均実施間隔は 2,069 秒である．

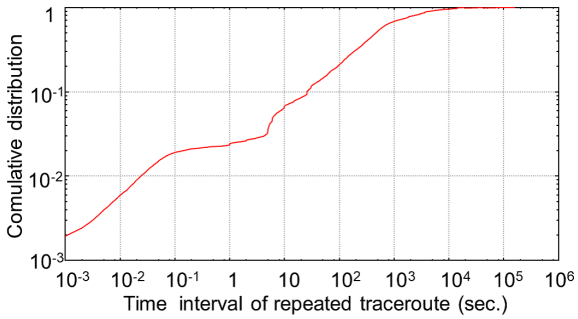

図 2　WIDE の公開トレースデータから作成した
同一ホストの traceroute 発生間隔の累積分布

ターゲットエリア内のサーバ数を 100 とし，パラメタ 0.8 の Zipf 分布に従う人気順位を与え，人気の上位 k 台のサーバをターゲットサーバとする．正常ホスト数を 10,000 とし，正常ホストは，確率 0.5 でターゲットエリア内のサーバを，確率 0.5 でターゲットエリア外のサーバを対象に各 traceroute を実施する．一方，ボット数を 50 とし，ボットは検知回避のため，各ターゲットサーバに対して 1 回のみ traceroute を行う．また各 traceroute の相手ホストとして，確率 0.2 でターゲットサーバ以外の一般サーバを選択し，さらに一般サーバを選んだ場合は確率 0.5 でターゲットエリア内のターゲットサーバ以外の一般

サーバを選択する．各ボットが traceroute を行う平均時間間隔は，ネットワークリンクが更新されると予想される 600 秒 [5] をターゲットサーバの数で除し，ターゲットサーバ選択確率を乗じた値とする．そして各ボットは指数分布に従う時間間隔で tracerouote を反復する．このような条件で，10,000 秒間シミュレーションを実施する．

5.2 評価結果

正常ホストの誤検知確率 False Positive Ratio (FPR) と，ボットを検知できずに見逃す確率 False Negative Ratio (FNR)，全検知ホストにおける検知ボット数の割合であるヒット率の 3 つを評価尺度に用いる．

$T_d = 0.001$，ターゲットサーバの数を 10 に設定し，T_s を変化させた際の提案方式 (DT: Double Threshold) と比較方式 (ST: Single Threshold) の FPR と FNR を図 3 に示す．DT 法，ST 法共に T_s の増加に伴い FNR は 0 に漸近し，T_s がおよそ 100.0 を超えたあたりで両方式とも FNR は 0 となり全ボットが検知される．一方，FPR は ST 法に比べ DT 法は 1/2 程度に抑えており，DT 法は正常ホストの誤検知率を抑えつつボットの検知が可能である．

また図 4 に各方式のヒット率を T_s に対し示すが，T_s の広い領域で DT 法は ST 法と比較してヒット率が増加することが確認できる．ヒット率は FPR が大きいほど全検知ホスト数が多いためヒット率は低下し，FNR が小さいほどボットの検知数が増加するためヒット率は増加する．図 3 に示すように，T_s が 0.01〜0.05 の範囲では FNR が 1 に近く FPR が 0 に近く，0.05〜1.0 あたりでは FPR は増加し FNR は減少するがボットより正常ホストの数が多いことからヒット率は低下する．一方 T_s が約 1.0〜10.0 の領域では，T_s の増加に伴う FPR の増加量より FNR の減少量が大きいためヒット率は増加する．さらに T_s が約 10.0 より大きい領域では，T_s の増加に伴う FPR の増加量が大きく，一方で FNR は 0 に近づき減少量が逓減するためヒット率は減少する．

次に T_s を 100.0，ターゲットサーバ数を 10 に設定し，T_d を変化させた際の DT 法と ST 法の FPR と FNR を図 5 に示す．ST 法は T_d の影響を受けないため FPR，FNR ともに T_d に対し一定である．DT 法は T_d が約 0.003 以上のとき FNR は 0 となり全ボットを検知し，FPR は ST 法と比較して低く抑えられる．

また図 6 に各方式のヒット率を T_d に対し示すが，やはり T_d の広い領域で DT 法は ST 法と比較してヒット率が増加することが確認できる．DT 法では T_d の増加に伴いヒット率は減少するが，図 5 に示すように，DT 法の FNR は T_d が 0.0001 の時点で既に 0 に近く，T_d の増加に伴う FNR の減少量は小さいのに対し，DT 法の FPR は T_d が約 3.0 までは T_d の増加に伴い増加し，以後，一定となるため，DT 法のヒット率は T_d が約 3.0 までは t_d の増加に伴い減少し，以後，一定となる．

T_s を 100.0，T_d を 0.001 に設定しターゲットサーバ数を変化させたときのヒット率を図 7 に示す．ST 法ではターゲットサーバリストを用いないためターゲットサーバ数に対しヒット率は一定となる．DT 法はターゲットサーバ数の増加に伴い，ボットの traceroute 発生間隔が短くなるためボットの検知率が増加し，ヒット率は増加する．

以上のことから，traceroute の発生間隔に基づき発着両側の検知閾値を考慮した DT 法では，発ホスト側のみを考慮した方式と比較して，FNR の増加を抑えながら FPR の大幅な抑制が可能であることが確認できる．

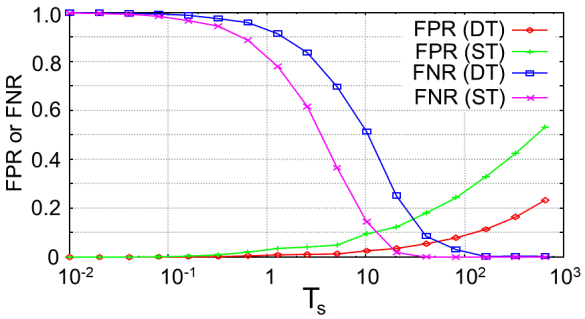

図 3　T_s に対する FPR と FNR

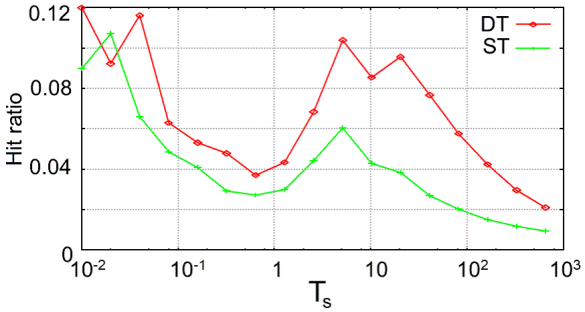

図 4　T_s に対するヒット率

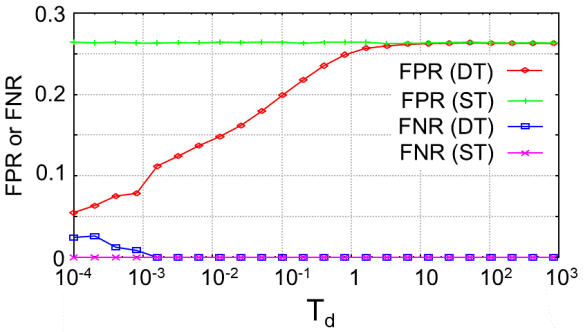

図 5　T_d に対する FPR と FNR

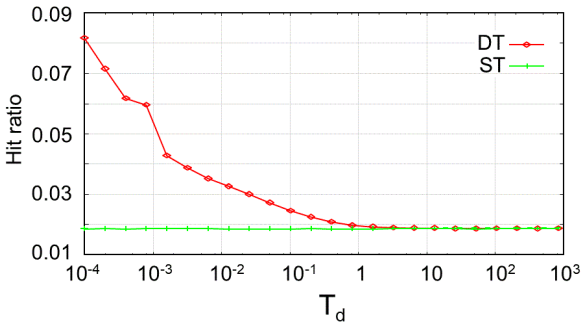

図 6　T_d に対するヒット率

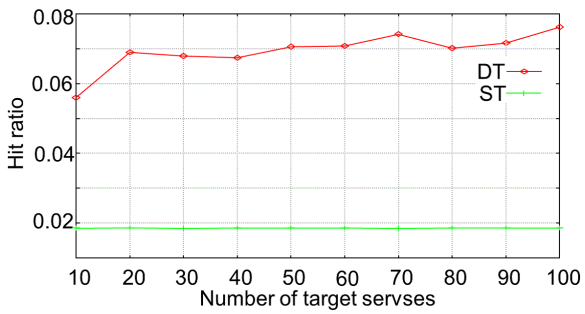

図 7　ターゲットサーバ数に対するヒット率

6. 最適閾値設定法

6.1 閾値の最適設計問題

提案方式においては，2つの閾値 T_s と T_d をどのように設定するかが問題である．そこでまず正常ホストの誤検知確率の許容最大上限を設定し，それを満足する発ホスト側検知閾値 T_s と着ホスト側検知閾値 T_d を最適閾値することを考える．そのため閾値に対する正常ホストの誤検知確率 FPR を簡易な式で得られる必要がある．そこで本設では FPR を導出する．

Ψ を正常ホストの誤検知確率 FPR と定義すると，Ψ は T_s と T_d に依存するため $\Psi = f(T_s, T_d)$ と表記できる．ただし $f(T_s, T_d)$ は T_s と T_d の増加に対し単調に増加する関数である．また T_s と T_d の増加に伴いボットを見逃す確率 FNR は単調に減少するため，FNR の観点からは T_s と T_d は大きいほうが望ましい．そこで1つ目の最適閾値設計法として，任意に与えた T_d と FPR の制約上限 $\hat{\Psi}$ に対し，FNR を最小化するよう以下の最適化問題により T_s を最適設計することを提案する．

$$\max \quad T_s \tag{1}$$

$$\text{s.t.} \quad \Psi = f(T_s, T_d) \leqq \hat{\Psi} \ \text{ for given } \ T_d, \hat{\Psi} \tag{2}$$

同様に2つ目の最適閾値設計法として，任意に与えた T_s と FPR の制約上限 $\hat{\Psi}$ に対し，FNR を最小化するよう以下の最適化問題により T_d を最適設計することを提案する．

$$\max \quad T_d \tag{3}$$

$$\text{s.t.} \quad \Psi = f(T_s, T_d) \leqq \hat{\Psi} \ \text{ for given } \ T_s, \hat{\Psi} \tag{4}$$

6.2 サーバがターゲットサーバリストに挿入される確率

正常ホストの traceroute 発生間隔は上述にある実測の分布を付与する．そのため全正常ホストからの traceroute 発生間隔も実測の分布 $q(t)$ を与える．一方，各正常ホストが T_s 以下の時間間隔で連続して traceroute を行う確率は実測の分布より，$F_n(T_s)$ で与えられる．正常ホスト数が N_n のとき，全正常ホストからの traceroute の発生間隔は，各事象が独立に一定の発生率で生じる多数の事象を重畳したものであるため，平均が $1/\lambda_{n,all}$ の指数分布に従うと仮定する ($\lambda_{n,all} = N_n \lambda_n$)．

$\boldsymbol{S}$ をサーバの集合とし，人気順位が k のサーバ S_k が traceroute の相手として選択される確率 z_k は，パラメタ $\theta = 0.8$ の Zipf 分布を想定し

$$z_k = c/k^{\theta} \tag{5}$$

$$c = 1/\sum_{k=1}^{S} 1/k^{\theta} \tag{6}$$

となる．ボットの数と比較して正常ホストは圧倒的に多いと仮定し，サーバにボットから到着する traceroute は無視する．サーバ S_k が任意の正常ホストから測定期間 M 中に j 回の traceroute を受ける確率 $g_k(j)$ は，平均が $B_k = M z_k \lambda_{n,all}$ のポアソン分布に従うので

$$g_k(j) = {B_k}^j e^{-B_k}/j \tag{7}$$

となる．測定期間 M 中に 1 回以上，連続して T_s 時間内に traceroute を受けるとサーバはターゲットサーバリストに挿入されるので，M 中で j 回の traceroute を正常ホストから受けたサーバがターゲットサーバリストに挿入される確率 $w(j)$ は

$$w(j) = 1 - \left\{1 - q(T_d)\right\}^{j-1} \tag{8}$$

となる．$q(T_d)$ は全正常ホストからの traceroute 発生間隔の分布より閾値 T_d 以下となる確率である．

式 (7)，(8) をもとに，サーバ S_k がターゲットサーバリストに挿入される確率 W_k は次式で求められる．

$$W_k = \sum_{j=1}^{\infty} g_k(j) w(j) \tag{9}$$

6.3 正常ホストの誤検知確率

測定開始時点のおいて，既にターゲットサーバリストが作成されていることを想定する．ある正常ホストが連続して 2 回の traceroute をサーバ S_{k1} と $S_{k2}(k1 \neq k2)$ に対して行ったとき，この正常ホストが検知されるのは，以下の両方を満たす場合である．

- S_{k1}，S_{k2} の両方がターゲットサーバリストに入っていること: 確率 $W_{k1} W_{k2}$
- traceroute を行った時間間隔が T_s 以下であること：確率 $F_n(T_s)$

よって1組の連続する 2 回の traceroute によって，ある正常ホストが検知される確率 $D(T_s)$ は

$$D(T_s) = \sum_{k1 \in \boldsymbol{S}} \sum_{k2 \in \boldsymbol{S}, k2 \neq k1} z_{k1} z_{k2} W_{k1} W_{k2} F_n(T_s) \tag{10}$$

となる．測定期間中に q 回の traceroute を行った正常ホストが検知される確率 $P_n(q)$ は

$$P_n(q) = 1 - \left\{1 - D(T_s)\right\}^{q-1} \tag{11}$$

となる．各正常ホストの traceroute 実施間隔の平均値を $1/\lambda$ とすると，ある正常ホストが測定期間 M 中に k 回の traceroute を行う確率 $h(k)$ は平均が $A = M\lambda$ のポアソン分布に従うので

$$h(k) = A^k e^{-A}/k! \tag{12}$$

となる．式 (11)，(12) をもとに，正常ホストの誤検知確率は次式で求められる．

$$p_n = \sum_{k=1}^{\infty} P_n(k) h(k) \tag{13}$$

6.4 評価条件

評価条件の基本は 5.1 節と同様に設定し，ターゲットサーバの数を 10 に設定する．

導出式によって得られた解析値の精度を確認するため，解析値とシミュレーション値の誤検知確率 FPR を比較する．

6.5 評価結果

着ホスト側検知閾値 T_d を 10.0 秒に設定し，発ホスト側検知閾値 T_s を変化させた際の正常ホストの誤検知確率 FPR の解析値とシミュレーション値を図 8 に示す．

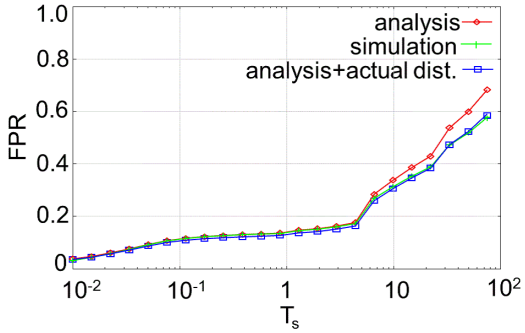

図 8　検知閾値 T_s に対する FPR

正常ホストの誤検知確率は図 2 に示した $F_n(x)$ と似た傾向を示し，T_s の増加に伴い FPR は増加するが，T_s が 7 秒程度より大きくなると FPR の増加率が大きくなる．T_s が 7 秒程度より小さい場合には FPR の解析値はシミュレーション値とよく一致するが，T_s がそれより大きな場合は誤差が見られる．

誤差が生じた原因は，シミュレーションでの正常ホストの traceroute 発生間隔は実際の分布を用いているが，導出式では指数分布に従うと仮定したためであると予想される．そこで，ポアソン分布に従うと仮定した $g_k(j)$，$h(k)$ に関してシミュレーションで用いた実際の分布を用いた場合 (analysis+actual dist.) の FNR も図 8 に示す．この場合，解析値とシミュレーション値の値はほぼ一致する．以上のことから，T_s が 7 秒程度より小さい場合には導出式 (13) の誤検知確率は良好な推定精度が得られることが確認できる．

6.6 閾値の最適設計

6.1 節で述べたように，本稿では 2 つの閾値の最適設計法として，一方を与えたときに，FPR の制約上限を満たす条件で FNR を最小化するようもう一方の閾値を最大化する．ここでは最適設計値の例を見る．そのため任意に与えた FPR の制約上限 $\hat{\Psi}(0.01, 0.05, 0.1)$, 閾値 $T_d(0.001, 0.01, 0.1)$ に対し，FNR を最小化する T_s の最大値 (最適値)T_s^* を計算機シミュレーションにより得た．その際の各 $\hat{\Psi}$ と T_d の組に対する T_s^* と FNR を表 1 にまとめる．同じ $\hat{\Psi}$ において，T_d の増加に伴い T_s^* は減少し，さらに FNR は増加する．また $\hat{\Psi}$ の増加に伴い T_s^* は大幅に減少する．例えば提案方式を用いることで，FPR を 0.1 程度に抑えながら FNR を数%以下に抑制できることが確認できる．

表 1　FPR 制約上限 $\hat{\Psi}$，閾値 T_d に対する最適閾値 T_s^*，FNR

$\hat{\Psi}$	T_d	T_s^*	FNR
0.01	0.001	1.5	0.8978
	0.01	0.666	0.904
	0.1	0.467	0.9156
0.05	0.001	35.0	0.1156
	0.01	11.181	0.1778
	0.1	6.52	0.318
0.1	0.001	12.101	0.0048
	0.01	47.712	0.0006
	0.1	26.396	0.0152

7. ま と め

近年，特定のターゲットエリア内と接続するリンクを高負荷とすることで，ターゲットエリア内に存在するホストへのパケット転送を妨害する Crossfire attack (CFA) の脅威が指摘されている．そこで本稿では，CFA では攻撃に先立ち，短い時間内に連続して多数のボットからターゲットサーバ宛てに traceroute を発生させる特徴に着目し，発着両側の検知閾値を考慮した CFA 検知方式を提案した．そして単一ルータにおけるシミュレーションにより，発ホスト側検知閾値のみを考慮した方式と比較して，ボットを見逃す確率 FNR の増加を抑制しながら，正常ホストの誤検知確率 FPR の大幅な抑制が可能であることを示した．

また提案方式において，正常ホストの誤検知確率の許容最大上限を考慮しつつボットの検知精度を最大化する最適閾値設計法を提案し，そのための FPR の簡易な式を導出した．提案した FPR 導出式は，発ホスト側検知閾値 T_s が約 7 秒より小さい場合には，高精度な近似を与えることを確認した．また最適閾値設計をシミュレーションにより行った結果，FPR と FNR との間のトレードオフの関係を確認した．

今後は攻撃者の挙動やターゲットサーバの選定方法を変更し，改めて導出式を改善し，より精度の高い近似式を導出する．そして提案 CFA 検知方式に，得られた最適閾値を設定した評価を行う．

謝辞　本研究成果は，JSPS 科研費 18K11283 の助成を受けたものである．ここに記して謝意を表す．

文　献

[1] A. Aydeger, K. Akkaya, M. Rahmam, and N. Saputro, Mitigating Crossfire Attacks using SDN-based Moving Target Defense, LCN 2016.

[2] A. Aydeger, K. Akkaya, and N. Saputro, Utilizing NFV for Effective Moving Target Defense against Link Flooding Reconnaissance Attacks, MILCOM 2018.

[3] D. Belabed, M. Bouet, and V. Conan, Centralized defense using smart routing against link-flooding Attacks, CSNet 2018.

[4] C. Dietzel, M. Wichtlhuber, G. Smaragdakis, and A. Feldmann, Stellar: Network Attack Mitigation using Advanced Blackholing, ACM CoNEXT 2018.

[5] F. Gillani, E. AI-Shear, M. Ammar, Q. Duan, S. Lo, and E. Zegura, Agile Virtualized Infrastructure to Proactively Defend Against Cyber Attacks, INFOCOM 2015.

[6] D. Gkounis, X. Dimitropouls, V. Kotronis, and C. Liaskos, On the Interplay of Link-Flooding Attacks and Traffic Engineering, ACM CCR, 2016

[7] T. Hirayama, K. Toyoda, and I. Sasase, Fast Target Link Flooding Attack Detection Scheme by Analyzing Traceroute Packets Flow, WIFS 2015.

[8] M. Kang, S. B. Lee, and V. D. Gligor, The Crossfire Attack, IEEE SSP 2013.

[9] M. Kang, V. Gligor, and V. Sekar, SPIFFY: Inducing Cost-Detectability Tradeoffs for Persistent Link-Flooding Attacks, NDSS 2016.

[10] J. Kim and S. Shin, Software-Defined HoneyNet: Towards Mitigating Link Flooding Attacks, ICDSN 2017.

[11] S. Lee, V. Gligor, and M. Kang, CoDef: Collaborative Defense Against Large-Scale Link-Flooding Attacks, ACM CoNEXT 2013.

[12] X. Ma, B. An, X. Guan, J. Li, and Y. Tang, Protecting internet infrastructure against link flooding attacks: A techno-economic perspective, Elsevier Information Science, 2019.

[13] A. R. Narayanadoss, M. Gurusamy, P. M. Mohan, and T. Truong-Huu, Crossfire Attack Detection using Deep Learning in Software Defined ITS Networks, VTC Spring 2019.

[14] W. Rafique, W. Dou, Z. Liu, X. He, and Y. Sun, CFADefense: A Security Solution to Detect and Mitigate Crossfire Attacks in Software-Defined IoT-Edge Infrastructure, HPCC 2019.

[15] R. Rasool, K. Ahmed, ,Z. Anwar, U. Ashra, W. Rafique, and H. Wang, Cyberpulse: A Machine Learning Based Link Flooding Attack Mitigation System for Software Defined Networks, IEEE Access, 2019.

[16] L. Wang, X. Jia, Y. jiang, Q. Li, and J. Wu, Woodpecker: Detecting and mitigating link-flooding attacks via SDN, Elsevier Computer Networks, 2018.

[17] J. Wang, J. Li, and R. Wen, Detecting and Mitigating Target Link-Flooding Attacks Using SDN, IEEE Trans. Dependable and Secure Computing, 2018.

[18] MAWI Working Group Traffic Archive, WIDE, https://mawi.wide.ad.jp/mawi/samplepoint-G/2019/

[19] L. Xue, E. W. W. Chan, G. Gu, X. Luo, X. Ma, and T. T. N. Miu, LinkScope: Toward Detecting Target Link Flooding Attacks, IEEE Trans. Information Forensics and Security, 2018.

[20] J. Zheng, J. Cao, G. Gu, D. Yau, and J. Wu, Realtime DDoS Defense Using COTS SDN Switches via Adaptive Correlation Analysis, IEEE TIFS, July 2018.