偽装CDNキャッシュサーバによるSlow HTTP DoS攻撃の検知方式

岡部 大樹[1,a)]　稲葉 宏幸[1]

概要：近年，コンテンツ提供者はCDN(Content Delivery Network) を利用してコンテンツの配信を行うことが多くなっていることから，CDN に対するセキュリティ対策が重要になっている．一方で，サイバー攻撃の手法として HTTP サーバに対して少ないデータ量で長時間トラフィックを圧迫することでサーバをサービス不能状態にさせる Slow HTTP DoS 攻撃が確認されている．これまでに，CDN 環境下での DDoS 攻撃の検知手法や，Slow HTTP DoS 攻撃を防御する複数の手法が提案された．しかし，オリジンサーバに対する Slow HTTP DoS 攻撃が行われた場合には，従来手法では攻撃を防げない可能性が考えられる．そこで本研究では DNS への名前解決時のタイムスタンプと，オリジンサーバ接続時のコネクション確立時間のタイムスタンプの時間差を計測し，それが一定時間内に収まればアクセスを許可し，そうでなければ攻撃と判断してアクセスを拒否することで CDN 上のオリジンサーバに対する Slow HTTP DoS 攻撃を検知する手法を提案した．シミュレーション結果より提案手法が一定の効果を持つことが明らかとなった．

Detecting Slow HTTP DoS Attacks Using a Spoofed CDN Cache Server

Daiki Okabe[1,a)]　Hiroyuki Inaba[1]

Abstract: In recent years, as content providers have increasingly used content delivery networks (CDNs) to deliver content, security measures for CDNs have become more important. On the other hand, slow HTTP DoS attacks have been confirmed. To date, several methods have been proposed to detect slow HTTP DoS attacks and DDoS attacks in CDN environments. However, conventional methods may not be able to prevent attacks in the case of a Slow HTTP DoS attack on the origin server. Therefore, in this study, we proposed a method to detect Slow HTTP DoS attacks on origin servers by measuring the time difference between the timestamp of name resolution to DNS and the timestamp of connection establishment time when connecting to the origin server,and if it is within a certain time, access is allowed.The simulation results showed that the proposed method had a certain effect.

1. はじめに

近年，コンテンツ提供者はCDN(Contents Delivery Network) を利用してアクセス集中への対策や大規模コンテンツの安定した配信を図ることが多くなっている．一方で，攻撃対象に対して大量のデータを送信することでネットワークやサーバの過負荷状態を引き起こしサービス不能状態にするDoS(Denial of Service) 攻撃や多数のボット端末を利用して DoS 攻撃を行う DDoS(Distributed Denial of Service) 攻撃が問題となっている．近年では HTTP サーバに対し少ないデータ量で長時間トラフィックを圧迫することで負荷をかける Slow HTTP DoS 攻撃が確認されている．

CDN を用いてコンテンツ提供を行っている場合，アクセスが CDN 上の多数のキャッシュサーバに送信されるため攻撃の多くがキャッシュサーバに分散される．また，オリジンサーバに直接攻撃された場合も，本来キャッシュサーバからのアクセスからのみ要求が届くため，それ以外からのアクセスを破棄できる．しかし，攻撃端末がキャッシュサーバを騙って攻撃した場合には検知ができない．このようなキャッシュサーバを騙る攻撃についてはキャッシュ

[1] 京都工芸繊維大学
Kyoto Institute of Technology
[a)] okabe20@sec.is.kit.ac.jp

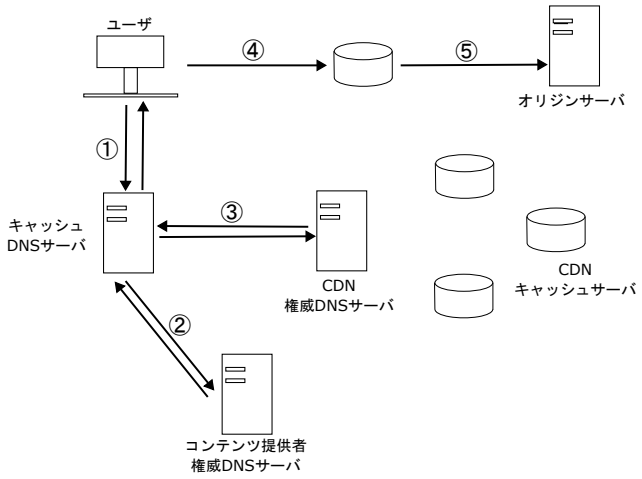

図 1 CDN によるコンテンツアクセスの流れ
Fig. 1 Procedure for accessing content using CDN.

サーバからのアクセス頻度と DDoS 攻撃のアクセス頻度の違いを用いた対策法が研究されている．[1] また一方で，Slow HTTP DoS 攻撃への対策方法の研究も進んでおり，タイムアウト時間を適切に設定することで長時間のコネクションを防ぐ方法や，コネクション数とコネクション継続時間から適切にコネクションを切断する手法が考えられている．[2][3] しかしながらこれらの方法は正常なアクセスを誤って切断してしまう可能性がある．

現状，CDN を用いたネットワーク上でのキャッシュサーバを騙った Slow HTTP DoS 攻撃への対策は行われていない．DDoS 攻撃と異なり，Slow HTTP DoS 攻撃であれば少ないアクセスとデータ量で負荷をかけることができるため，前述した CDN への DDoS 攻撃の対策では防ぐことができないと考えられる．

そこで本論文では，偽装 CDN キャッシュサーバによる Slow HTTP DoS 攻撃の検知方式の提案を行う．加えて提案方式の性能評価及び誤検知率の評価を行う．

2. CDN

CDN(Contents Delivery Network) は web ページや画像，動画といったコンテンツをキャッシュするキャッシュサーバを地理的に分散して配置し，元のコンテンツを管理するサーバであるオリジンサーバに代わってキャッシュサーバから配信を行うネットワークのシステムである．ユーザに近接したキャッシュサーバから配信を行うことでアクセスの高速化が可能となり，コンテンツを多数のキャッシュサーバに分散して配置することでアクセスの集中を防ぐことが可能となる．

CDN を用いたネットワークにおけるユーザのコンテンツアクセス時の名前解決方法を**図 1** に示す．

(1) ユーザはキャッシュ DNS サーバにオリジンサーバの名前解決の依頼を行う．
(2) キャッシュ DNS サーバはコンテンツ提供者の権威 DNS サーバに IP アドレスを問い合わせる．コンテンツ提供者の権威 DNS サーバは CDN ホスト名が登録された CNAME タイプレコードを返す．
(3) キャッシュ DNS サーバは CDN の権威 DNS サーバに問い合わせる．CDN の権威 DNS サーバはユーザに近いキャッシュサーバの IP アドレスを返す．
(4) キャッシュ DNS サーバがユーザ端末に CDN キャッシュサーバの IP アドレスを回答し，ユーザは CDN キャッシュサーバにアクセスする．
(5) コンテンツがキャッシュされていない場合，CDN キャッシュサーバはオリジンサーバに配信要求を行う．

3. Slow HTTP DoS 攻撃

3.1 攻撃の概要

Slow HTTP DoS 攻撃は大量にパケットを送信する Flood 攻撃とは異なり，HTTP のリクエストを長時間維持させ続けることでサーバの処理するリクエストの上限に達しサービス不能状態にさせる攻撃である．HTTP GET リクエストヘッダの一部分を低速で送る Slow HTTP Header 攻撃，HTTP POST リクエストヘッダの一部分を低速で送る Slow HTTP POST 攻撃，サーバに小さな TCP ウィンドウサイズを通知することで低速でレスポンスを送信させる Slow Read 攻撃がある．

3.2 Slow HTTP DoS 攻撃の検証

Slow HTTP DoS 攻撃の攻撃レートを検証するため，Slow HTTP DoS 攻撃の実験を行う．なお，実験は他に影響を与えないようにローカル環境で行う．

3.2.1 検証用ソフトウェア

検証用ツールとして用いる Slowloris.py [4] は Slow HTTP DoS 攻撃の一種である Slow HTTP Headers 攻撃を行うツールである．3.1 で説明したように，コネクションを確立した後に GET リクエストを低速で送信しコネクションを長時間維持させる．これによりサービス不能状態を起こし，正当なユーザがアクセスできないようにする．

このツールは python で記述されており，実行するには”python3 Slowloris.py IP アドレス/URL [オプション]”とコマンドライン上で入力する．-p オプションでポート番号の設定，-s で使用するソケット数の設定ができる．

3.2.2 検証の概要

Slowloris.py を使用し，ローカル環境で Web サーバに Slow HTTP DoS 攻撃を行い，wireshark [5] を用いてパケットを監視し攻撃の様子を観察した．また同時にリソース監視ツールを用いて負荷状況を確認した．

具体的な実験環境を以下に示す．

- Web サーバ側
  - Windows10 上で Oracle VM VirtualBox を用いた仮想環境

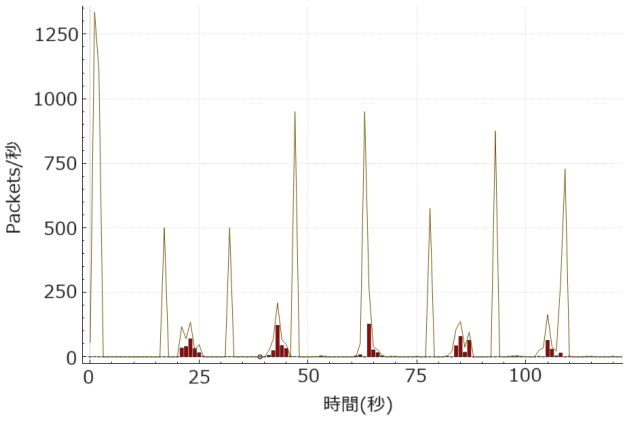

図 2 Slowloris.py による攻撃のパケット量の時間変化
Fig. 2 Changes in the number of packets in Slowloris.py attacks.

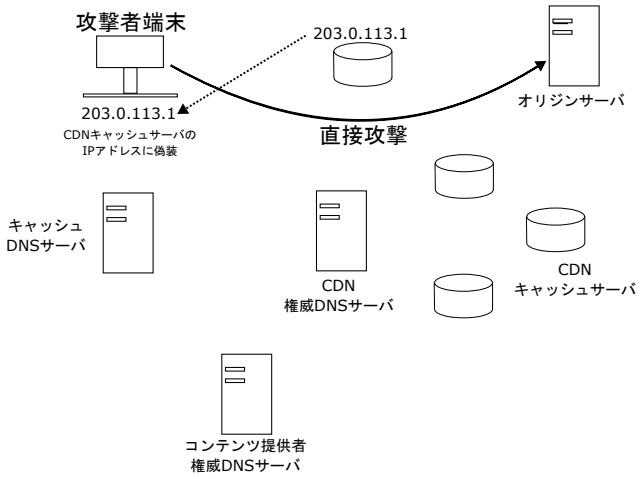

図 3 攻撃時のオリジンサーバへのアクセス
Fig. 3 Accessing an origin server when attaking.

- メモリ:8GB
- OS:Ubuntu 22.04.3
- サーバアプリケーション：apache 2.4.58

• 攻撃端末側
- OS:MacOS Catalina

web サーバ側には”Hello World”のみを記述した html ファイルを置いた．web サーバ側の IP アドレスを 192.168.1.14，攻撃端末側の IP アドレスを 192.168.1.13 に設定して，攻撃端末側で 192.168.1.14 を攻撃するように指定した．

3.2.3 検証結果

図 2 に wireshark を用いた Slowloris.py による攻撃のパケットキャプチャによって得られた，時間ごとのパケット数のグラフを示す．折れ線グラフは時間ごとの攻撃者のパケット数を示す．棒グラフは送信したパケットがサーバ側に届かなかったために再送されたパケット数である．攻撃を開始すると大量の GET リクエストパケットが web サーバ側に送信された．最初に送信を行ってから，10 秒から 15 秒おきに GET リクエストパケットが繰り返し送信された．また，20 秒から 25 秒おきにパケットの一部が再送されていることが分かった．再送されたパケットが発生する原因として，攻撃により帯域が圧迫されたため，攻撃で送信したパケットの一部がドロップしたからだと考えられる．

攻撃が行われている間は web ページに接続しようとしても常に読み込み中の状態となり攻撃によってサービス不能状態になることが確認できた．

4. 提案手法

4.1 提案手法の概要

CDN 環境におけるオリジンサーバへの攻撃検知手法として DNS サーバの連携を利用する方法が提案されている．一般に，キャッシュミス時に発生する配信要求でのアクセス間隔は長く，DDoS 攻撃で発生するアクセス間隔は短い．この 2 つのアクセス間隔の違いから閾値を設定し，閾値の間隔以下で発生した要求は攻撃とみなし破棄する方式が考えられた．[1] しかし，閾値の間隔以上で攻撃が行われる Slow HTTP DoS 攻撃の場合には検知できないという問題がある．また，Slow HTTP DoS 攻撃に対する一般的な対策として，サーバプログラム上でタイムアウト時間を設定することや，1 クライアント当たりの同時セッション数を制限することが挙げられる．[2] しかし，タイムアウト時間を設定する方法では，回線速度が遅い正当なユーザのリクエストが拒否される可能性があるため，タイムアウト時間を短くすることはできない．

そこで，本論文ではこれらの手法の問題点を踏まえて，コンテンツ提供者の権威 DNS サーバのアクセス時間とオリジンサーバの配信要求時のコネクション確立時間の差に閾値を設けてアクセスの許可を判断する手法を提案する．

4.2 提案手法の詳細

次に提案手法の詳細について説明する．正当なアクセスにおけるコンテンツ提供の手順は図 1 のような流れで行われる．

一方，**図 3** で示すように，攻撃者は端末の IP アドレスを CDN キャッシュサーバの IP アドレスに偽装した上で，権威 DNS サーバへの問い合わせを行わず，直接オリジンサーバへ攻撃を行う．

そのため，正当なアクセスであればコンテンツ提供者の権威 DNS サーバへの問い合わせとオリジンサーバへの配信要求時のコネクション確立が短い間隔で発生する．これに対して，攻撃を行った場合にはコンテンツ提供者の権威 DNS サーバへの問い合わせ時刻にかかわらずオリジンサーバへのアクセスが発生するため，正当なアクセスの時とは異なり，必ずしも短い間隔で発生するわけではない．そこでこの違いを利用し，以下のようなステップでオリジンサーバへの攻撃の検知・防御を行う．

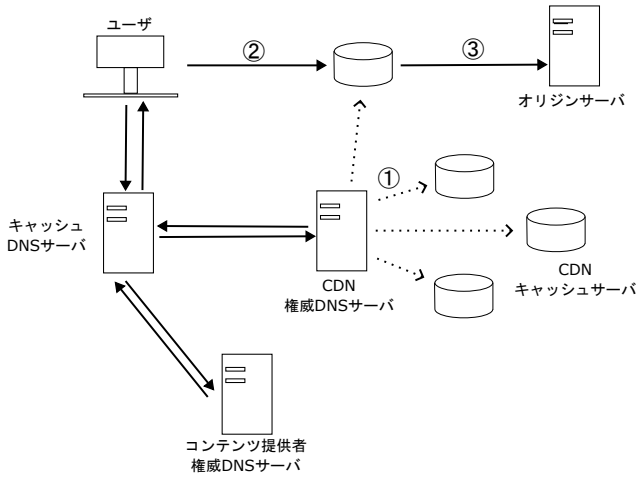

図 4 DNS サーバへの問い合わせとオリジンサーバへのコネクション確立の間の処理

Fig. 4 Processing between querying DNS server and establishing a connection with an origin server.

(1) コンテンツ提供者の権威 DNS サーバへの問い合わせ時に，クエリログから最新の問い合わせ時刻 t_{DNS} を取得する．

(2) オリジンサーバへのコネクション確立が発生した時に，サーバログからコネクション確立確率時刻 t_{con} を取得する．

(3) コンテンツ提供者の権威 DNS サーバへの問い合わせ時刻とコネクション確立確率時刻の差 $t = t_{con} - t_{DNS}$ を取得する．

(4) t が設定した閾値 T よりも小さければ正当なアクセスと判断しアクセスを許可する．そうでない場合アクセスを拒否する．

この提案方法を実装するためにはコンテンツ提供者の DNS とオリジンサーバでタイムスタンプが記録されるように設定する必要がある．また，ユーザのアクセス毎にコンテンツ提供者の権威 DNS サーバに問合せするようにしなければならない．そのため権威 DNS サーバの DNS レコードの TTL の値を小さい値にして，ユーザ側での名前解決においてキャッシュの参照を極力させないようにする必要がある．

5. 閾値の設定

コンテンツ提供者の権威 DNS サーバへの問い合わせ時刻とオリジンサーバの配信要求時のコネクション確立時刻の差を元にアクセス許可の判断をするため，閾値の設定が必要になる．コンテンツ提供者の権威 DNS サーバの問い合わせとオリジンサーバの配信要求時のコネクション確立の間に挟まる処理は**図 4** で示すように，以下の 3 つとなる．

① CDN の DNS でのキャッシュサーバの選択
② ユーザのキャッシュサーバへのアクセス
③ 配信要求時のコネクション確立

ユーザのキャッシュサーバへのアクセスに関しては CDN がユーザ端末に対して地理的に近接するキャッシュサーバを選択しているため，アクセス時間は無視できるものとして考える．そこで閾値を設定する上では CDN の DNS でのキャッシュサーバの選択の時間と配信要求時のコネクション確立時間を考慮しなければならない．

5.1 CDN の DNS でのキャッシュサーバの選択時間

一般的な DNS での名前解決時間は DNS サーバ間を反復的に問合せして行うため長くなる傾向にある．しかし，CDN の DNS でキャッシュサーバを選択する場合は CDN のサービス運営者が保持しているキャッシュサーバからユーザに近接しているキャッシュサーバの IP アドレスを返すだけである．そのため処理時間は比較的短くなると考えられる．ここでは高々 100 ミリ秒以内に処理が行われると仮定する．

5.2 コネクション確立処理時間

HTTP 接続であれば 3 ウェイハンドシェイク，HTTPS 接続では 3 ウェイハンドシェイクの後に TLS ハンドシェイクが行われてコネクションが確立される．3 ウェイハンドシェイクは 3 パケットで終わるが TLS ハンドシェイクはバージョンによっても異なるが 3 パケットから 4 パケットのやり取りが行われる．また，データの暗号化・復号が含まれるため一定の時間を要する．こちらについても高々 100 ミリ秒で処理が行われると仮定する．

上述した 2 つの処理時間は様々な要因である程度変動することが想定される．処理時間の変動も踏まえた上で正当なユーザのアクセスを拒否しないような閾値の設定が必要になる．

一般に閾値を大きくすれば正当なユーザのアクセスが拒否されにくくなるが攻撃も通りやすくなる．逆に閾値を小さくして攻撃を通しにくくすれば正当なユーザのアクセスが誤って拒否される確率が上昇する．閾値の設定にはこのようなトレードオフの関係があると考えられる．

6. 性能評価

6.1 性能評価手法

本章では以下の 2 点:

- CDN のキャッシュサーバに偽装した端末がオリジンサーバに対してどの程度攻撃を成功させることができるのか (攻撃成功率)
- 正当なユーザのアクセスがどの程度誤検知されるのか (偽陽性率)

についてプログラムを作成しシミュレーションを行った．

6.1.1 攻撃成功率

1 時間あたりの正当なアクセスとそれに対する攻撃をランダムに行うシミュレーションを行った．シミュレーショ

ン実験の手順は以下の通りである．

(1) 1 時間を 3600000 ミリ秒として 0 以上 3600000 未満の整数乱数を正当アクセスの回数分生成し，これらをコンテンツ提供者の権威 DNS サーバへの問い合わせ時刻として設定する．

(2) 同様に攻撃タイミングの時刻を設定する．

(3) 問い合わせ時刻から閾値として設定した時間の間に攻撃があれば攻撃成功として，成功した攻撃の個数と攻撃されたアクセスの個数をカウントする．

(4) (1) から (3) を 1000 回繰り返し平均値を求める．

1 つのアクセス内で複数の攻撃が成功していても攻撃されたアクセスの個数は 1 としてカウントする．変化させるパラメータは以下の 3 つである．

- 1 時間当たりのユーザによる正当アクセスの回数 $N_{atk} = \{60, 120, 240\}$
- 1 時間当たりの攻撃回数 $N_{acc} = \{300, 600, 1200\}$
- 閾値 $T = \{50\text{ms}, 100\text{ms}, 150\text{ms}, 200\text{ms}, 250\text{ms}, 300\text{ms}\}$

これらパラメータの全ての組合せについて実験を行い，攻撃成功確率と，攻撃されたアクセス数の割合を算出する．

6.1.2 偽陽性率

CDN が管理する DNS サーバにおけるキャッシュサーバの選択時間とコネクション確立時間について，それらが正規分布に従うと仮定して合計の処理時間を考える．CDN が管理する DNS サーバにおけるキャッシュサーバの選択時間の平均値を μ_{DNS}，標準偏差を σ_{DNS} とする．また，コネクション確立時間の平均値を μ_{con}，標準偏差を σ_{con} とする．

一般に 2 つの正規分布の平均を μ_1, μ_2，標準偏差を σ_1, σ_2 とするとき，2 つの正規分布の和も正規分布となる．この時，平均は $\mu_1 + \mu_2$，標準偏差は $\sqrt{\sigma_1^2 + \sigma_2^2}$ となる．

今回は前章で述べたように 2 つの処理時間をそれぞれ最大で 100 ミリ秒と仮定する．変化させるパラメータは以下の通りである．

- 処理時間の平均値 $\mu_{DNS} = \mu_{con} = \{10\text{ms}, 50\text{ms}, 100\text{ms}\}$
- 標準偏差 $\sigma_{DNS} = \sigma_{con} = \{5\text{ms}, 10\text{ms}, 12.5\text{ms}, 25\text{ms}, 50\text{ms}\}$
- 閾値 $T = \{50\text{ms}, 100\text{ms}, 150\text{ms}, 200\text{ms}, 250\text{ms}, 300\text{ms}\}$

これらパラメータの全ての組合せについて和の正規分布を考え，その累積分布関数から閾値を超えた値となる確率を算出する．

6.2 性能評価結果

6.2.1 攻撃成功率

N_{atk} を 300 回として，N_{acc} を 60 回，120 回，240 回と変化させたときの攻撃成功確率のシミュレーション結果を**図 5** に示す．横軸に閾値，縦軸に攻撃成功確率をプロットしている．

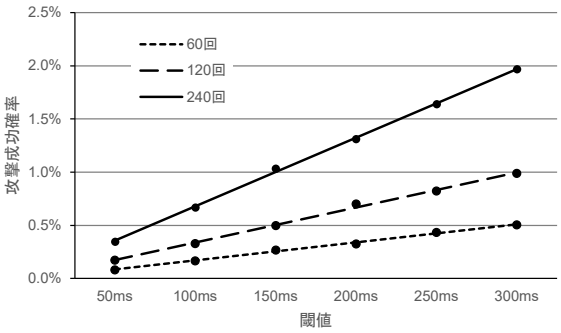

図 5 1 時間当たりの正当アクセス回数ごとの閾値による攻撃成功確率の変化

Fig. 5 Changes in the probability of attack success based on the threshold for the number of legitimate accesses per hour.

図 5 からは，閾値を大きくすることで，攻撃成功確率が上昇することがわかる．また，閾値が同じでも正当アクセス回数を増やすことで攻撃成功確率が上昇することがわかる．これは 1 時間当たりの正当アクセス回数を増やしたり，閾値を大きくすることで 1 時間当たりの攻撃可能時間の割合が増加するためだと考えられる．実際，T を 300 ミリ秒とし，N_{acc} が 60 回の時に，およそ 0.5%で攻撃が成功しているが，1 時間当たりの攻撃可能時間の割合を求めると式 (1) になる．

$$\frac{N_{atk} \times N_{acc}}{3600000(\text{ms})} \times 100 = \frac{300(\text{ms}) \times 60}{3600000(\text{ms})} \times 100 = 0.5 \quad (1)$$

また，N_{atk} を 600 回，1200 回として同様のシミュレーションを行ったが，N_{atk} を変化させても閾値ごとの攻撃成功確率はほとんど変化しなかった．そのため N_{atk} を 600 回，1200 回とした時のグラフは図 5 とほとんど同様のものとなる．これは，式 (1) で示したように，攻撃回数は攻撃可能時間の割合に関与しないためだと考えられる．

N_{acc} を 60 回として，N_{atk} を 300 回，600 回，1200 回と変化させたときの攻撃されたアクセス数の割合のシミュレーション結果を**図 6** に示す．横軸に閾値，縦軸に攻撃成功確率をプロットしている．

図からは，閾値を大きくすることでアクセスを許可する時間が伸び，攻撃されたアクセス数の割合が上昇することがわかる．また，閾値が同じでも回数を増やすことで攻撃成功確率が上昇することがわかる．N_{acc} を 120 回，240 回として同様のシミュレーションを行ったが，N_{acc} を変化させても閾値ごとの攻撃成功確率はほとんど変化しなかった．そのため N_{acc} を 120 回，240 回とした時のグラフは図 6 とほとんど同様のものとなる．

6.3 偽陽性率

μ_{DNS} と μ_{con} を 10 ミリ秒としたときの閾値ごとの偽陽性率の結果を**図 7** に，50 ミリ秒としたときの結果を**図 8**

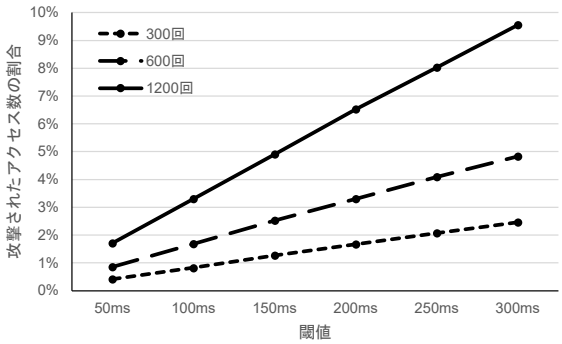

図 6 1 時間当たりの攻撃回数ごとの閾値による攻撃成功確率の変化
Fig. 6 Changes in the probability of attack success according to the threshold for the number of attacks per hour.

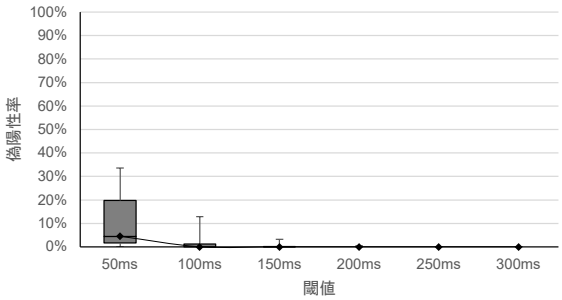

図 7 μ_{DNS} と μ_{con} を 10 ミリ秒としたときの閾値ごとの偽陽性率
Fig. 7 False positive rate for each threshold when μ_{DNS} and μ_{con} are 10 msec.

に，100 ミリ秒としたときの結果を**図 9** に示す．横軸に閾値，縦軸に偽陽性率をプロットしている．各図は σ_{DNS} と σ_{con} を変化させたときの偽陽性率の範囲を箱ひげ図によって表している．各処理時間が正規分布にしたがうとしてシミュレーションしているため，閾値が処理時間の平均値の合計と等しい時，偽陽性率が 50%となる．中央値はひし形のマーカで表しており常に $\sigma_{DNS} = \sigma_{con} = 12.5$ms とした時の値である．最大値は，偽陽性率が 50%を超えるまでは $\sigma_{DNS} = \sigma_{con} = 50$ms の時の値であり，50%を超えると $\sigma_{DNS} = \sigma_{con} = 5$ms の時の値となる．最大値は，偽陽性率が 50%を超えるまでは $\sigma_{DNS} = \sigma_{con} = 5$ms の時の値であり，50%を超えると $\sigma_{DNS} = \sigma_{con} = 50$ms の時の値となる．

各図からは，3 種類の処理時間の平均値の全てで，閾値を小さくすると偽陽性率が大きく上昇していることが読み取れる．

全ての処理時間の場合において，同じ処理時間であれば，偽陽性率が 50%を超えた点からは標準偏差が小さいほど偽陽性率が高くなっている状態となる．

7. 考察

提案手法中の閾値について適切な値を考察する．

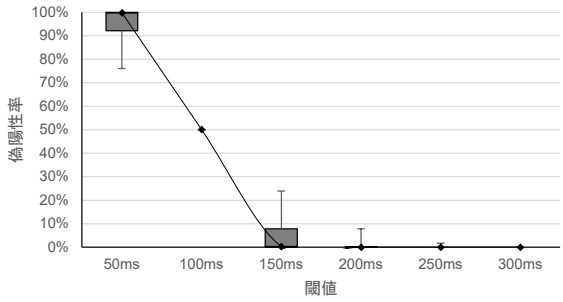

図 8 μ_{DNS} と μ_{con} を 50 ミリ秒としたときの閾値ごとの偽陽性率
Fig. 8 False positive rate for each threshold when μ_{DNS} and μ_{con} are 50 msec.

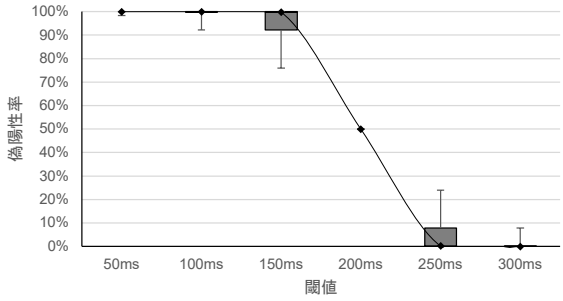

図 9 μ_{DNS} と μ_{con} を 100 ミリ秒としたときの閾値ごとの偽陽性率
Fig. 9 False positive rate for each threshold when μ_{DNS} and μ_{con} are 100 msec.

Slow HTTP DoS 攻撃の検証実験の結果より，攻撃レートは 10 秒から 15 秒に 1 回程度であることが推定できる．これは 1 時間当たりの攻撃回数に換算すると 240 回から 360 回ほどになる．また，コンテンツがキャッシュされていない時にのみオリジンサーバに正当なアクセスが発生するという特性から 1 時間当たりの正当なアクセスの回数は高々 60 回程度であると仮定できる．このとき，性能評価結果より T を 300 ミリ秒としても攻撃成功率は 0.5%程度となり一定の効果が発揮できると見込まれる．閾値が 300 ミリ秒であればキャッシュサーバの選択時間とコネクション確立時間がそれぞれ 100 ミリ秒あっても，十分に小さな偽陽性率 (0.23%) を達成できる．

ただし，攻撃されるアクセスの割合は 300 ミリ秒の時に 2%程度となると考えられる．1 時間当たりの正当なアクセスの回数が 60 回だとすると，常に攻撃が行われているという仮定の下では 1 時間から 2 時間に 1 回は攻撃を受ける可能性がある．また，偽陽性率も処理時間が大幅に遅延した場合には 8%程度になる可能性がある．そのため，より攻撃を防御しつつ偽陽性率を下げるために，各処理時間が安定して短くなることや処理時間を検知して動的に変化させる方法が求められる．

提案手法は，コンテンツ提供者が管理する DNS やサーバの設定を変更するだけでよく，新たに設備を導入する必要

がないという特徴がある．また，タイムスタンプをログとして残すだけで実装が可能なため，理論上どのようなサーバプログラムでも導入可能であると考えられる．

一方，提案手法では，アクセスが発生するたびにコンテンツ提供者の権威 DNS サーバのログの取得と，コネクション確立時のログの取得が必要なため，オリジンサーバへの負荷が大きくなる可能性がある．また，リソースレコードの TTL を短くするため DNS キャッシュポイズニング攻撃への耐性が下がることが考えられる．今後これらの影響について精査する必要がある．

8. むすび

本研究では CDN 上のキャッシュサーバを騙った端末によるオリジンサーバに対する Slow HTTP DoS 攻撃の検知・防御を目的として，コンテンツ提供者の権威 DNS サーバの問い合わせ時刻と，オリジンサーバへのコネクション確立時間の時刻の差を計算し，この時刻差と設定した閾値を比較してアクセス許可の判定を行うシステムを提案した．提案したシステムのシミュレーションを行った結果，攻撃成功率は閾値が大きいほど高く，正当なアクセス回数が多いほど高くなることが分かった．また攻撃されたアクセスの割合は閾値が大きいほど高く，攻撃回数が多いほど高くなることが分かった．偽陽性率については，閾値が小さいほど高くなり，標準偏差が大きいほど高くなる傾向が見られた．一般的な Slow HTTP DoS 攻撃の攻撃レートを考慮すると，提案したシステムは一定の防御効果があることが明らかになった．今後の課題としては，処理時間を検知して動的に閾値を変化させるシステムの開発等が挙げられる．

参考文献

[1] 宮崎椋平，上山憲昭：Cdn のキャッシュサーバを騙った ddos 攻撃の防御方式, 信学会 2021 年総合大会，B-11-10, 2021.

[2] 川橋裕，池田和樹，安江伴輔，藤本章宏：slow http dos 攻撃の手法と防御方法について学術情報処理研究，vol.20, no.1，pp.128–136，2016.

[3] 平川哲也，小倉加奈代，バハドゥール ビスタベッド，高田豊雄：Tcp コネクション数と継続時間に基づく slow http dos 攻撃に対する防御手法情報処理学会論文誌，vol.61, no.3，pp.581–590，03 2020.

[4] G. Yaltirakli “Slowloris,” 2015.

[5] W. Foundation: Wireshark, （オンライン）, 入手先〈https://www.wireshark.org/〉（参照 2024-1-31）.