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
NS2021-139(2022-03)

Bloom Filterによるキャッシュポリューション攻撃検知における正常ホストの誤検知低減法

芦原　大和†　　上山　憲昭††

† 福岡大学大学院 工学研究科 電子情報工学専攻　〒 814–0180 福岡市城南区七隈 8–19–1
†† 立命館大学 情報理工学部　〒 525-8577 滋賀県草津市野路東 1–1–1
E-mail: †td202001@cis.fukuoka-u.ac.jp, ††kamiaki@fc.ritsumei.ac.jp

あらまし　Web 閲覧や動画配信サービスを快適に行うため、ネットワークの様々な場所にキャッシュサーバを設置しホストの近くに存在するキャッシュサーバからデータを配信するキャッシュ配信が広く行われている．しかし悪意を持ったホストが意図的に低人気のコンテンツに多数の要求を行うことでキャッシュの効果を低下させるキャッシュポリューション攻撃 (CPA：cache pollution attack) の問題が指摘されている．CPA 発生時にはできるだけ迅速に攻撃ホストを特定し，キャッシュを保護することが重要である．そこで少ないメモリアクセス回数で高精度に攻撃ホストを特定するため，筆者らはホスト ID とコンテンツ ID の組をキーとする Bloom Filter (BF) を用いた検知方式を提案した．しかし正常ホストも既要求のコンテンツを複数回，視聴する可能性があり，その場合，正常ホストを CPA ホストとして誤検知する問題ある．そこで正常ホストの誤検知を避けるために，BF の検知回数に閾値を設け，閾値を超えた回数，BF で検知されたホストを CPA ホストとして検知することを提案した．また，継続的な CPA ホストの検知を可能とするため，BF のビットマップを 2 つ用意して交互に運用することを提案した．しかしこれまでに提案した方式では，検知回数が時間とともに増加し続けるため，いずれほとんどの正常ホストが誤検知される．そこで本稿では，正常ホストの検知を防ぐため，検知回数を周期的にデクリメントすることを提案する．また，これまでの評価では正常ユーザの要求発生分布に静的な Zipf 分布を想定しており，実際のユーザの同一コンテンツの反復視聴行動を正しく反映していない．そこで本稿では，実際のユーザの YouTube の視聴ログデータから動画コンテンツの要求発生モデルを構築し，本モデルを用いた数値評価を行う．提案方式を用いることで正常ホストの誤検知を防ぎつつ、CPA ホストを高精度に検知しキャッシュヒット率の低下を回避可能なことを示す．
キーワード　キャッシュ，キャッシュポリューション攻撃，Bloom Filter

Reducing False Identification of Normal Hosts in Detecting Cache Pollution Attack by Bloom Filter

Takakazu ASHIHARA† and Noriaki KAMIYAMA††

† Department of Electronic and Information Technology, Fukuoka University　8–19–1, Nanakuma, Jounan, Fukuoka 814–0180
†† College of Information Science and Engineering, Ritsumeikan University　1–1–1, Noji-higashi, Kusatsu,Shiga 525–8577
E-mail: †td202001@cis.fukuoka-u.ac.jp, ††kamiaki@fc.ritsumei.ac.jp

Abstract　To efficiently providing web browsing and video streaming services with reasonable quality，cache delivery in which digital data is delivered to users from cache servers located close to users has been widly used．However，the cache pollution attack (CPA) which degrades the effect of caches by intentionally sending many requests to non-popular content items will be a problem in the cache networks．Quickly detecting the CPA hosts and protecting the cache servers is important to effectively utilize the cache resources．Therefore，in this paper，we propose a method of accurately detecting the CPA hosts with using a limited amount of memory resources．The proposed method is based on a Bloom filter using the combination of host ID and content ID as keys．Normal hosts may also view the requested content multiple times，but in the proposed method，the number of detections increases over time，so most normal hosts will eventually be falsely detected．Proposed to introduce a threshold on the number of detections in the Bloom filter to identify the CPA hosts．We also proposed to use two Bloom filters in parallel to continuously operate the proposed detection method of CPA hosts．However，since the number of detections continues to increase over time，most normal hosts will eventually be falsely detected．Therefore，in this paper，in order to prevent false detection of normal hosts，we propose that when the number of detections exceeds the BF detection threshold，it is detected as CPA hosts and the number of detections is periodically decremented．In addition，the evaluations so far have assumed a static Zipf distribution for the request generation distribution of normal users, and do not correctly reflect the repeated viewing behavior of the same content of the user．Therefore，in this paper，we construct a request generation model for video content from the YouTube viewing log data of the actual user，and perform numerical evaluation using this model．We show that the proposed method reduce the degradation of the cache hit ratio caused by the CPA while avoiding the false identification of legitimate hosts.
Key words　cache, cache pollution attack, Bloom filter

1. はじめに

Web 閲覧や動画配信サービスを快適に行うため，多数のネットワーク上に多数のキャッシュサーバを配置し，ホストの近くに存在するキャッシュサーバからデータを配信する CDN (content delivery network）が広く行われている．また近年，スマホなどの移動端末においても動画コンテンツなどの大容量コンテンツを視聴するホストが増加し，基地局間を接続するバックホールネットワークのトラヒック量が増大し，その設備投資コストの増大や品質の低下が懸念される．そこでバックホールネットワークのトラヒック量の増加を回避するため，基地局にキャッシュサーバを配置し，ホストからの配信要求に対しキャッシュサーバからコンテンツを配信する Mobile Edge Computing (MEC) が注目され，精力的に研究が進められている [8] [12]．さらに次世代のネットワークとして期待を寄せられている情報指向ネットワーク (ICN：Information-Centric Networking) においても，ルータにキャッシュメモリを用意しルータからコンテンツを配信する [2]．

しかしキャッシュサーバの容量は有限であるため，存在する全てのコンテンツをキャッシュすることはできず，新たにコンテンツをキャッシュする際に容量が不足する場合にはキャッシュ置換アルゴリズムを用いて選択したキャッシュ済みコンテンツの一部を削除する．通常，コンテンツの人気には偏りがあるため，高人気のコンテンツを優先的にキャッシュに残すことでキャッシュヒット率の向上とトラヒック量削減効果の向上が期待できる．そのため最後に要求されてからの経過時間が最大のコンテンツから削除する LRU (least recently used) がキャッシュ置換アルゴリズムとして広く用いられている．

ところで悪意を持ったホストが意図的に低人気のコンテンツに多数の要求を行うことで，キャッシュの効果を低下させるキャッシュポリューション攻撃 (CPA：cache pollution attack) の問題が指摘されている [10] [14]．少ないメモリアクセス回数で迅速に CPA ホストを検知するため，ホスト ID とコンテンツ ID の組をキーとする Bloom Filter（BF）を用いて CPA ホストを検知する方式を提案した [1]．しかし正常ホストも既要求のコンテンツを複数回，視聴する可能性があり，その場合，正常ホストを CPA ホストとして誤検知する問題ある．そこで正常ホストの誤検知を避けるために，BF の検知回数に閾値を設け，閾値を超えた回数，BF で検知されたホストを CPA ホストとして検知することを提案した [1]．また，継続的な CPA ホストの検知を可能とするため，BF のビットマップを 2 つ用意して交互に運用することを提案した [1]．しかしこれまでに提案した方式では，検知回数が時間とともに増加し続けるため，いずれほとんどの正常ホストが誤検知される．そこで本稿では，正常ホストの検知を防ぐため，検知回数を周期的にデクリメントすることを提案する．また，これまでの評価では正常ユーザの要求発生分布に静的な Zipf 分布を想定しており，実際のユーザの同一コンテンツの反復視聴行動を正しく反映していない．提案方式は反復視聴に着目し検知を行うため，反復視聴行動は重要な要素である．そこで本稿では，実際のユーザの YouTube の視聴ログデータから動画コンテンツの要求発生モデルを構築し，本モデルを用いた数値評価を行う．提案方式を用いることで正常ホストの誤検知を防ぎつつ、CPA ホストを高精度に検知しキャッシュヒット率の低下を回避可能なことを示す．

以下，2 節では関連研究について，3 節では提案方式の詳細を述べる．そして 4 節では YouTube の視聴履歴の分析結果を示し，5 節で提案方式の数値評価結果を示す．最後に 6 節で全体をまとめる．

2. 関連研究

ICN を対象とした CPA の検知技術が数多く提案されている．例えば Yao らは，コンテンツを要求比率と要求の平均発生間隔に基づき，コンテンツを人気コンテンツと不人気コンテンツの 2 つにクラスタリングし，人気クラスタから不人気クラスタに移ったコンテンツ数の比率が閾値を超えた場合に CPA の発生を検出し，移ったコンテンツに対する Prefix を持つ配信要求 (Interest) に対してはコンテンツをキャッシュしないことを提案している [15]．また Guo らは攻撃者の Interest は特定のルータから生成されるため Interest の経路の多様性が低いことに着目し，要求が均一に発生する場合と比較した経路の多様性が閾値を下回る場合に検知する方式を提案している [7]．また Conti らは，ランダムにサンプルしたコンテンツセットに対し，各コンテンツの長期間における要求発生比率と，あるスナップショットにおける要求発生比率との差の総和が閾値を超えたときに CPA の発生を検知する方式を提案している [3]．

また Xu らは，CPA は同一の Prefix を用いて異なる多数のコンテンツが要求される点に着目し，複数のハッシュ関数とビットマップを用いて異なり数を計測する Flajolet-Martin sketch (FM Sketch) を用いて，異なり数が閾値を超えたときに CPA を検知する方式を提案している [13]．さらに Park らは，CPA 発生中はキャッシュされているコンテンツのランダム性が増加する点に着目し，コンテンツ名から得られるハッシュ値でキャッシュコンテンツを 2 値行列にマッピングし，高人気コンテンツを過去の 2 時点の行列との XOR をとって消去した行列のランクの統計的な量を閾値と比較することで CPA の発生を検知することを提案している [11]．しかしこれらの研究は ICN のルータのキャッシュに対する CPA に対象が限定されている．

一方，MEC のキャッシュに対する CPA を対象とした研究としては，Paul らの研究 [10] や Yang らの研究 [14] が見られる．Paul らは，HetNet の Femtocell に設置されたキャッシュを攻撃対象とする CPA の影響をシミュレーション評価している [10]．また Yang らは，CPA が Cooperative MEC の性能に与える影響を計算機シミュレーションにより分析し，攻撃者が全ノードをカバーする最少数のキャッシュに攻撃を行った場合の効果を明らかにしている [14]．しかしこれらの研究は CPA の影響の計算機シミュレーションによる評価結果を示すのに留まっている．

3. Bloom Filter を用いた CPA ホスト検知方式

本節では，まず 3.1 節で本稿で想定する CPA について述べる．さらに 3.2 節から 3.5 節で，筆者らがこれまでに [1] で提案した Bloom Filter を用いた CPA ホスト検知方式の概要を述べる．そして 3.6 節で，本稿で新たに提案する拡張方式について述べる．

3.1 キャッシュポリューション攻撃

CPA には，Locality-disruption 型と False-locality 型の 2 種類の攻撃法が存在する [5]．一般にキャッシュの性能はコンテンツの人気の偏りが強いほど大きくなるため，Locality-disruption 型の CPA では，攻撃ホストは低人気のコンテンツに対して一様 に要求を発生させ，コンテンツの人気の偏りを緩やかにすることでキャッシュの効果を低下させる．一方 False-locality 型の CPA では，攻撃ホストは低人気のコンテンツに対して多数の要求を発生させコンテンツの人気の順位を入れ替えることで，正常ホストの配信要求に対するキャッシュヒット率を低下させる．本稿の提案方式は，どちらの攻撃方法でも検知可能である．

本稿では CPA を一般化し，要求対象コンテンツを人気の最も低いものから C 個のコンテンツとする．ただしコンテンツ数を M とすると，$1 \leqq C \leqq M$ の範囲に C を設定する．CPA ホストが要求するコンテンツの選択方法として，文献 [1] において，筆者らが定義した 4 方式の内，キャッシュヒット率の低

下効果が大きな Smart divide 型を，Smart 型と定義する．

3.2 Bloom Filter を用いた CPA ホストの検知

正常なホストは短い時間内に既要求のコンテンツを複数回，視聴する可能性は低い．一方，CPA の攻撃ホストは短い時間内に既要求コンテンツに対し複数回の視聴要求を発生させる．そのため各配信要求に対し，ホスト ID とコンテンツ ID の組をテーブルで管理し，同一ホストからの既要求コンテンツに対する配信要求を検知すればよい．しかしホスト数とコンテンツ数の増加に伴い，管理テーブルに必要となるメモリアクセス回数が急増する．そこでメモリアクセス回数を抑えながら効率的に key の存在判定が可能である BF を採用した [1]．ホスト ID の例としては，例えば IP アドレスが考えられる．またコンテンツ ID の例としては，例えばコンテンツ事業者の ID を上位パートに，各コンテンツ事業者内で提供コンテンツを識別するために各コンテンツに付与された ID を下位パートに持つ，全世界でユニークなコンテンツ識別子が考えられる．

BF とは，空間効率のいい確率的データ構造である．k 個のハッシュ関数を用いて key に対する k 個のハッシュ値を生成する．生成されたハッシュ値に対応する Bitmap 上の k 個の bit がすべて 1 であれば 1 度呼ばれた key，1 つでも 0 があれば 1 度も呼ばれていない key であると判別される．その後生成された k 個のハッシュ値に対応する bit を 1 にセットする．この仕組みのため，異なる key に対し同一のハッシュ値が生成され，新規のものが既出現と判断される偽陽性による誤検知の可能性がある．ホスト ID とコンテンツ ID を key として BF を用いて，既出現性を判定することで，メモリサイズとメモリアクセス回数を抑えた CPA ホストの検知が可能となる．BF のサイズを M，偽陽性の目標エラー率を p とすると，BF への入力回数上限 N とハッシュ関数の個数 k は次式で与えられる．

$$N = -\frac{M(\log\ 2)^2}{\log\ p} \tag{1}$$

$$k = \frac{M}{N}\ \log\ 2 \tag{2}$$

3.3 2 個の Bloom Filter の並列運用

文献 [1] において，BF を永続的に使用するため，BF 用に 2 つの Bitmap BF1 と BF2 を用意し，これらを切り替えながら運用することを提案した．各 Bitmap に挿入するキーの総数を N 以下に抑えることから，Bitmap への挿入キー数が N に達した時点で，もう一方の Bitmap に運用を切り替える．ただし一方の BF から他方の BF に切り替えた時点で，新規に稼働を始める BF の Bitmap の更新を開始すると，Bitmap の全てのビットが 0 の状態から始めることになり，CPA ホストの検知が困難である．そこで運用開始の閾値パラメタ α を導入し，運用中の Bitmap の挿入キー数が αN に達した時点で，もう一方の Bitmap へのキーの挿入も並行して行う．そのため $\alpha < 0.5$ のとき，3 個以上の Bitmap が必要になるので，$0.5 \leqq \alpha \leqq 1.0$ の範囲で α を設定する．

3.4 BF 検知回数の閾値

正常ホストも既要求のコンテンツを複数回，視聴する可能性があり，その場合，正常ホストを CPA ホストとして誤検知する問題ある．正常ホストの検知を減らすため，BF での検知回数に BF 検知閾値 Y を設け，Y 回検知されたホストを CPA ホストとして検知した [1]．

BF 検知された各ホストの検知回数を記録するためのテーブル (BFDC: BF detection count) を用意し，ホスト u からの配信要求時には，まず BFDC の u に対する BF 検知回数 D_u をチェックし，$D_u < Y$ の場合はコンテンツ ID とホスト ID で BF 検索を行う．BF によって検知されなかった場合は，BF にコンテンツとホスト ID を記録し通常通りキャッシュを行う．BF によって検知された場合は，BFDC の D_u のカウンタをインクリメントする．$D_u = Y$ の場合は，BFDC の D_u のカウンタのインクリメントとキャッシュを行わない．

3.5 Bloom Filter の設計

Smart 型の CPA の場合，同一の CPA ホストが既要求のコンテンツを要求するまでに，この CPA ホストから C 回の要求発生が必要である．一方で，提案方式の BF は N 回のキー入力まで受け付けることができるため，提案方式は BF による CPA ホストの検知を行うには，N が見たすべき下限値 N_{min}(検知リミット) が存在する．文献 [1] において，N_{min} を導出した．全正常ホストの毎秒要求数を R_n，全 CPA ホストの毎秒要求数を R_c，CPA ホスト数を N_c とし，T を BF の 1 つの Bitmap にキーの挿入が開始されてからもう一方の Bitmap に切り替わるまでに要する時間とすると，

$$N_{min} = (R_c + R_n)T \tag{3}$$

となる．一方，1 つの CPA ホストから $C + Y$ 個の要求に対し，1 つの Bitmap にキーを挿入できれば CPA ホストの検知が可能なので，

$$\frac{R_c T}{N_c} = C + Y \tag{4}$$

が得られる．(4) から得られる T を (3) に代入すると，

$$N_{min} = \frac{N_c}{R_c}(R_n + R_c)(C + Y) \tag{5}$$

が得られる．(1) より，BF の 1 つの Bitmap に必要なメモリ量の下限値 M_{min} は，

$$M_{min} = -\frac{N_c(R_n + R_c)(C + Y)\ \log\ p}{R_c(\log\ 2)^2} \tag{6}$$

となる．

3.6 BFDC の周期的なデクリメント

正常ホストは少数のコンテンツに対し散発的に反復視聴するのに対し，CPA ホストは継続的にキャッシュヒット率を低下させるために大量のコンテンツに対し集中的に反復視聴する．そのため，筆者らが以前提案した方式 [1] では検知回数が時間とともに増加し続けるため，いずれほとんどの正常ホストが誤検知される．そこで本稿では，検知閾値のカウンタが増加し正常ホストが誤検知される問題を回避するため，検知回数を周期的にデクリメントすることを提案する．

BFDC の 1 以上のすべてのエントリを，一定周期 P(デクリメント周期) ごとに V(デクリメント値) だけデクリメントする．ただしカウンタ値が 0 となったエントリはそれ以上，減算しない．正常ホストの誤検出を抑えるためには，検出閾値とデクリメント値には，デクリメント周期ごとに起こりうる既要求コンテンツの要求の最大回数を超える数値を設定する．

4. YouTube の視聴履歴の分析

提案方式における正常ユーザの巻き添え率は正常ユーザの要求発生パタンに大きく依存する．しかし文献 [1] では正常ユーザの各要求に対し，独立に Zipf 分布に従う確率で要求コンテンツを選択したが，実際の要求発生パタンが考慮できていない可能性がある．そこで本稿では，正常ユーザの実際の要求発生パタンを考慮した性能評価を行う．本方式は同一ユーザが既要求のコンテンツを要求することで検知を行うため，正常ユーザが既要求コンテンツを繰り返し要求する時間間隔の分析が不可欠である．また現実の要求に近い形でシミュレーションを行うため，本節では実際のユーザの YouTube の視聴履歴を分析する．

YouTube は最も人気のある動画コンテンツの一つであり，日々多くの動画コンテンツが視聴および配信されており，モバイルデータ使用量の 38 パーセントを占めている [9]．そこで

YouTube の視聴履歴を分析する．筆者らの研究室のメンバー等の合計 21 人から Google アカウントに残る YouTube の視聴履歴を収集した．JSON ファイルで出力された YouTube の視聴履歴を個人情報保護の観点から Python プログラムにてコンテンツ名，チャンネル名をハッシュ値に変換し日時を取得した．YouTube の広告はデータから除外し，最長 3 年間の視聴履歴を分析した．

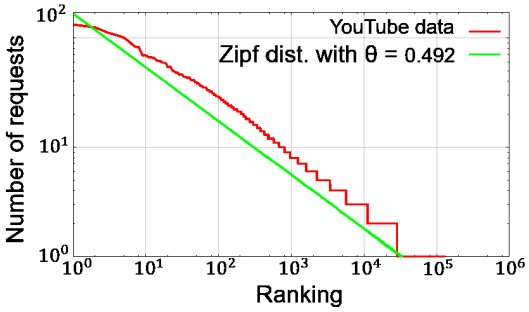

図 1　User request number distribution

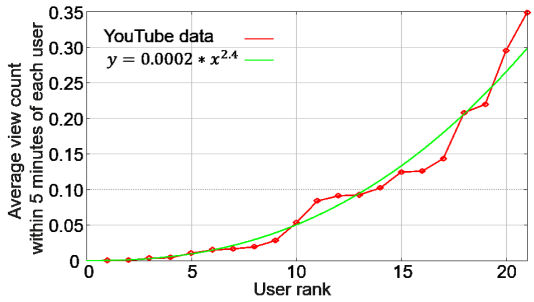

図 2　Average view count within 5 minutes of each user

全ユーザからの各コンテンツの総要求回数を，コンテンツごとに降順に並べた曲線を図 1 に示す．コンテンツの要求数はおおよそ，パラメタ $\theta = 0.492$ の Zipf 分布に従うことが確認できる．YouTube の要求数分布はパラメタ $\theta = 0.8$ の Zipf 分布に従うという報告がなされているが [4]，今回取得したデータすべての期間の要求回数の分布であるため要求の期間や要求コンテンツにばらつきがあることが要因と考えられる．

各ユーザが YouTube を利用していた期間 (最初にアクセスがあった時刻と，最後にアクセスのあった時刻との差分) 中の，5 分という単位時間における平均視聴回数を，その値の昇順にユーザごとにプロットした曲線とその近似式の曲線を図 2 に示す．各ユーザの YouTube のコンテンツの要求間隔は，$y = 0.0002 * x^{2.4}$ の式に近似できることを確認した．

各ユーザの既要求コンテンツ視聴割合を昇順に並べてた曲線とその近似式の曲線を図 3 に示す．各ユーザの YouTube の既要求コンテンツの要求割合は，$y = 0.0008 * x^2 + 0.009 * x - 0.003$ の式に近似できることを確認した．

図 4 に，各コンテンツに対し，反復要求 (同じ人の 2 回目以降の要求) の平均視聴間隔に対し，そのコンテンツの視聴回数を，散布図としてプロットする．既要求のコンテンツの平均視聴間隔は広く分布している．また視聴回数の多いコンテンツは，平均視聴間隔の短いものあれば，平均視聴間隔の長いものもある．すなわち反復視聴現象は，短期間に集中的に行われる場合も，長期間にわたって多数の視聴が行われる場合もある．

各コンテンツの反復視聴の平均視聴間隔の累積分布を図 5 に示す．また平均間隔を一致させた指数分布も合わせて示す．既要求コンテンツの同一ユーザによる視聴間隔の累積分布は指数分布と類似の形状を示すが，その分散は指数分布と比較して大きい．2 回以上視聴した各コンテンツの平均視聴間隔の平均値は 1441.677 時間，最小値は 0.033 時間，最大値は 64347.117 時間であった．

ユーザごとの 2 回以上視聴したコンテンツの要求時間間隔の最小値の累積分布を図 6 に示す．多くのユーザの既要求コンテンツの要求時間間隔の最小値は 24 時間以内である．

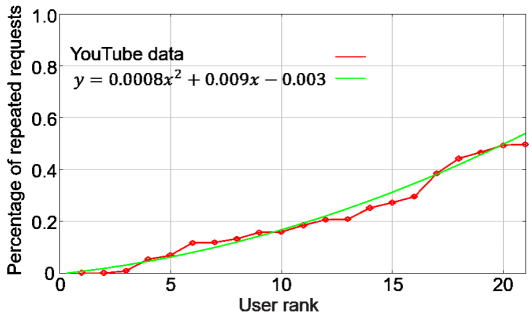

図 3　Percentage of repeated requests

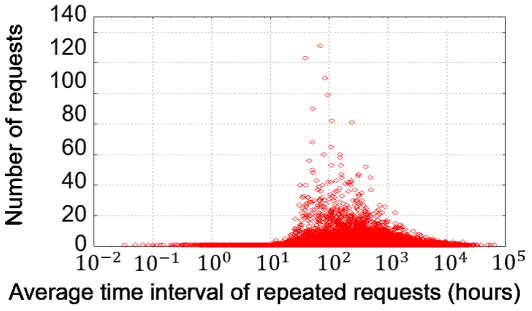

図 4　Number of requests of the content for the average interval of repeat requests

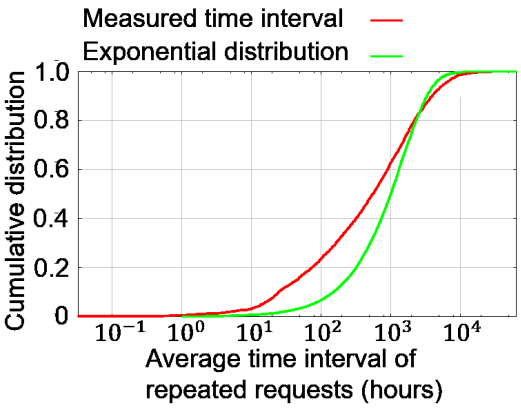

図 5　Cumulative distribution of average time intervals of repeated requests

5.　提案方式の数値評価

5.1　評 価 条 件

4 節で分析した，実際の YouTube コンテンツの視聴パタンに基づき，正常ユーザの要求を発生させた場合の計算機シミュレーション評価を行う．キャッシュ置換方式は LRU を想定する．M 個のコンテンツに対し，人気の高いものから順に $1, 2, \cdots, M$ の番号を付与し，また N 個のホストに対し，順に $1, 2, \cdots, N$ の番号を付与し，これらをコンテンツやホストの ID に用いる．コンテンツ数 $M = 20,000$ に対しキャッシュサイズは $1,000$ とする．正常ホスト数 $N = 1,000$ で $6,000$ 秒間シミュレーションを行う．$N_c = 10$ 個の CPA 攻撃ホストが存在し，各々はコンテンツの人気順位を把握していることを想定し，$1,000$ 秒から $5,000$ 秒の期間，攻撃ホストは CPA 対象コンテンツ $C = 1,000$ 個に対して毎秒 5 個の要求を行う．最も低人気の C 個のコンテンツに対し，攻撃対象コンテンツ数/CPA ホスト の間隔で等間隔に各 CPA ホストの要求開始

点を決め，人気の昇順にコンテンツを要求する Smart 型で要求を行う．BF の運用開始パラメタ を 0.5 に設定し，BF の偽陽性確率を $p = 0.01$ となるようサイズが 200KB のビットマップを BF 用に 2 個用いる．BF のハッシュ生成方法には，ハッシュの衝突を抑えるため murmur3 を利用する [6]．また BF 検知閾値を $Y = 10$ に，BFDC のデクリメント周期を $P = 100$ 秒に，デクリメント値を $V = 5$ に設定する．正常ホストは，研究室のメンバー 21 人の YouTube の視聴履歴をもとに作成した視聴間隔の分布の図 2 に基づき，ランダムに平均視聴間隔を 1,000 個の各正常ホストに設定する．そして各正常ホストに対し，図 3 に基づきランダムに既要求コンテンツの視聴割合を設定する．そして要求ごとに，初回要求コンテンツに対する要求か，既要求コンテンツに対する要求かを管理している各集合の選択確率からランダムに選択する．

正常ホストが未要求 (既要求) コンテンツを要求した場合は，未要求 (既要求) コンテンツの中からパラメタ 0.8 の Zipf 分布に従いランダムに選択したコンテンツを要求する．既要求コンテンツの要求割合の初期値は 0 とし，ホストがコンテンツを要求するごとに，個々のホストの既要求コンテンツの要求割合を総要求コンテンツの要求回数で除した値を，その正常ホストの既要求コンテンツの要求割合に加算していく．結果は全て 10 回の試行における平均値で評価する．

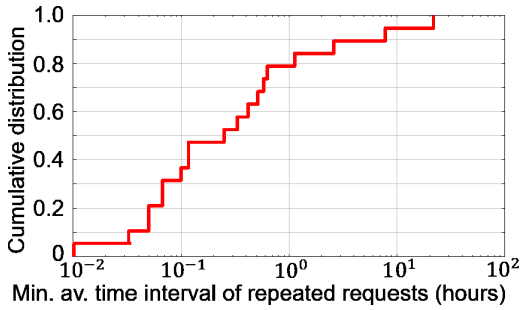

図 6　Cumulative distribution of minimum average time interval of repeated requests

5.2　巻き添え率

BF は偽陽性により，一度も要求をしていないにもかかわらず要求があったと判断される False positive が発生する可能性がある．一方，同じ要求を繰り返したにもかかわらず検知されない False negative は発生しない．正常なホストが提案方式によって誤検知される割合を巻き添え率と定義する．巻き添えの原因は，BF の False Positive に加え，同一ホストが偶然，既要求のコンテンツを要求した場合にも発生する．

図 7 は，BF 検知閾値を $Y = 10, 20, 30, 40$ に設定し，デクリメント値 V を変化させたときの正常ホストの巻き添え率をプロットする．BF 検知閾値 Y より大きなデクリメント値 V を設定した場合，$V = Y$ の場合と，巻き添え率は一致することから，$V \leqq Y$ の範囲でプロットする．検知閾値が小さい場合は，同じデクリメント値であってもデクリメントが発生するまでの期間に検知閾値回数を超えることが多くなり，巻き添え率が増加する．BFDC に対して周期的にデクリメントを行うことで巻き添え率が低下する．今回のシミュレーション条件において，デクリメント周期ごとに起こりうる既要求コンテンツの要求の最大回数は，23 回以上である．そのため Y や V が 23 を超える場合には，巻き添え率が 0.001 以下に抑えられることが確認できる．

BF 検知閾値 $Y = 20$，デクリメント値 $V = 10$ において，提案方式によって誤検知された各ホストと，誤検知されなかった各ホストの，平均要求発生間隔と，既要求コンテンツの要求割合の散布図を図 8 にプロットする．キャッシュヒットした要求の中で，要求間隔が短く，既要求コンテンツの要求割合が高いホストが生成した要求ほど，その割合が高い．図 8 より，このようなキャッシュヒットに占める割合が高いホストほど，提案方式によって検知されやすい傾向が確認できる．

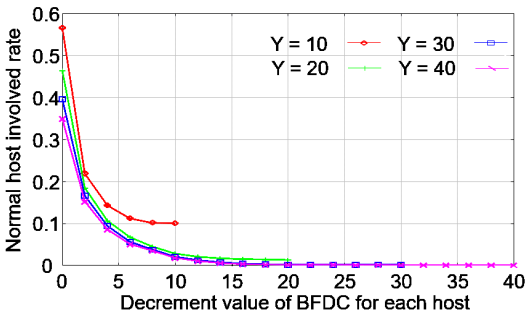

図 7　Normal host involved rate for decrement value of BFDC

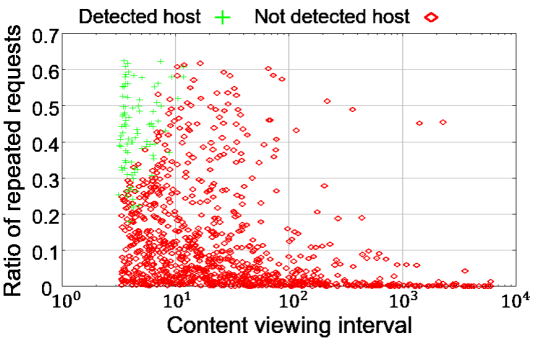

図 8　Distribution of content viewing interval and percentage of repeated requests of each host

5.3　攻撃強度

正常ホストのキャッシュヒット率の低下度合い (攻撃強度) を評価する．CPA の影響力は，キャッシュヒット率の低下量と，キャッシュヒット率の低下が継続した時間の長さの両方に依存し，各々が大きなほど攻撃強度も高くなる．そこで CPA の攻撃強度を，正常時のキャッシュヒット率と，CPA が行われた場合のキャッシュヒット率との差を，CPA が開始した時刻から，CPA ホストの検知に最も要した時間である 1,000 秒後の期間にわたり累積した値で定義する．

図 9 に，BF 検知閾値を $Y = 10, 20, 30, 40$ に，デクリメント値を $V = 10, 20, 30, 40$ に設定した場合の各々における，デクリメント周期を変化させた値の攻撃強度を示す．デクリメント周期が短い場合，デクリメントが頻繁に実施されるため CPA ホストを見逃すか，もしくは検知に要する時間が増加するため攻撃強度が低下する．一方，デクリメント周期が長い場合，正常ホストの巻き添えが増加するが，図 8 で見たように，主にキャッシュヒット率に対する貢献が大きなホストが主に誤検知されることから，正常ホストの平均的なヒット率が低下し，攻撃強度が増加する．

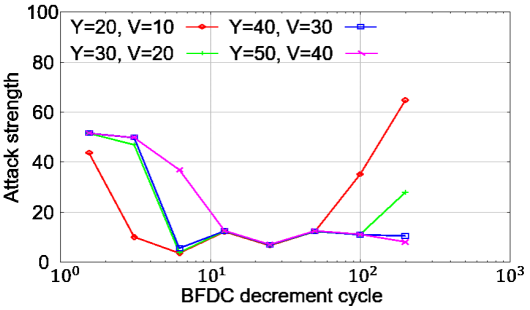

図 9　Attack strength against BFDC decrement cycle

5.4　検知時間の累積分布

図 10 に Y = 20，V = 10 に設定し，デクリメント周期 P を 5

つの値に設定したときの，CPA ホストの検知時間の累積分布を示す．デクリメント周期 $P = 1.5625$ の場合 CPA ホストの検知のカウントアップ中に BFDC のデクリメントが発生するため，CPA ホストの検知に遅れが発生する．$Y = 30, V = 20$ とした場合，$P = 1.5625$ では CPA ホストが検知できなかった．

図 11 に，BF 検知閾値を $Y = 10, 20, 30, 40$ に，デクリメント値を $V = 10, 20, 30, 40$ に設定した場合の各々における，CPA ホストの検知に要した時間の累積分布を示す．検知閾値とデクリメント値を増加させると検知時間は少し伸びる．しかし数秒単位であり，おおよそ 220 秒以内にすべての CPA ホストを検知することができる．

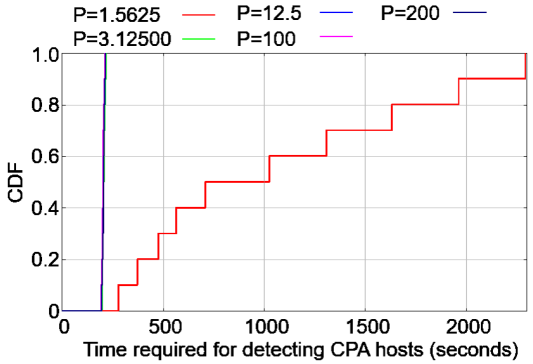

図 10　Cumulative distribution of detection time for each host with respect to the decrement cycle

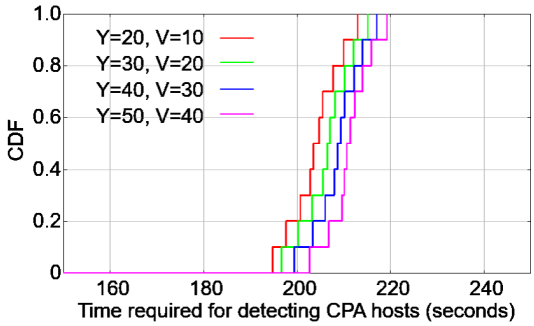

図 11　Cumulative distribution of detection time for each host to decrement value

6. ま と め

悪意を持ったホストが意図的に低人気のコンテンツに多数の要求を行うことでキャッシュの効果を低下させるキャッシュポリューション攻撃の問題が指摘されている．CPA 発生時にはできるだけ迅速に攻撃ホストを特定し，キャッシュを保護することが重要である．そこで少ないメモリアクセス回数で高精度に攻撃ホストを特定するため，筆者らはホスト ID とコンテンツ ID の組をキーとする Bloom Filter (BF) を用いた検知方式を提案した．しかし正常ホストも既要求のコンテンツを複数回，視聴する可能性があり，その場合，正常ホストを CPA ホストとして誤検知する問題ある．そこで正常ホストの誤検知を避けるために，BF の検知回数に閾値を設け，閾値を超えた回数，BF で検知されたホストを CPA ホストとして検知することを提案した．また，継続的な CPA ホストの検知を可能とするため，BF のビットマップを 2 つ用意して交互に運用することを提案した．しかしこれまでに提案した方式では，検知回数が時間とともに増加し続けるため，いずれほとんどの正常ホストが誤検知される．そこで本稿では，正常ホストの検知を防ぐため，検知回数を周期的にデクリメントすることを提案した．また，これまでの評価では正常ユーザの要求発生分布に静的な Zipf 分布を想定しており，実際のユーザの同一コンテンツの反復視聴行動を正しく反映していない．そこで本稿では，実際のユーザの YouTube の視聴ログデータから動画コンテンツの要求発生モデルを構築し，本モデルを用いた数値評価を行った．例えば取得したデータにおいて，BF の検知閾値とデクリメント値を 23 以上に設定することで，正常ホストの巻き添え率を 0.001 程度以下に抑えながら，220 秒程度以内に CPA ホストを検知できることを確認した．

謝辞　本研究成果は JSPS 科研費 18K11283 と 21H03437 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] T. Ashihara, and N. Kamiyama, Detecting Cache Pollution Attacks Using Bloom Filter, IEEE LANMAN 2021.
[2] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, “A Survey on Content-Oriented Networking for Efficient Content Delivery, IEEE Commun. Mag, vol.49, no.3, pp.121-127, Mar. 2011.
[3] M. Conti, P. Gasti, and M. Teoli, A lightweight mechanism for detection of cache pollution attacks in Named Data Networking, Elsevier Computer Networks, vol. 57, no. 16, pp. 3178-3191, Nov. 2013.
[4] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon, Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content Systems, IEEE/ACM ToN, Vol.17, NO.5, pp.1357-1370, Oct. 2009.
[5] L. Deng, Y. Gao, Y. Chen, and A. Kuzmanovic, Pollution attacks and defenses for Internet caching systems, Computer Networks, 52, pp.935-956, 2008.
[6] M. Dietrichstein.” Objective-C Bloom Filter implementation”. Github. 2017-5-18. https://github.com/mdietrichstein/objc-murmur3-bloomfilter
[7] H. Guo, X. Wang, K. Chang, and Y. Tian, Exploiting Path Diversity for Thwarting Pollution Attacks in Named Data Networking, IEEE Trans. Information Forensics and Security, vol. 11, no. 9, pp. 2077-2090, May 2016.
[8] J. Krolikowski, A. Giovanidis, and M. D. Renzo, Optimal Cache Leasing from a Mobile Network Operator to a Content Provider, IEEE INFOCOM 2018.
[9] S.Lall, and R.Sivakumar,A YouTube Dataset with User-level Usage Data:Baseline Characteristics and Key Insights, ICC 2020
[10] S. Paul, A. Seetharam, A. Mukherjee, and M. K. Naskar, Investigating the Impact of Cache Pollution Attacks in Heterogeneous Cellular Networks, IEEE ICNP 2017.
[11] H. Park, I. Widjaja, and H. Lee, Detection of Cache Pollution Attacks Using Randomness Checks, IEEE ICC 2012.
[12] X. Wang, M. Chen, T. Taleb, et al., Cache in the Air: Exploiting Content Caching and Delivery Techniques in 5G Systems, IEEE Communications Magazine, vol. 52, no. 2, pp. 131-139, Feb. 2014.
[13] Z. Xu, B. Chen, N. Wang, Y. Zhang, and Z. Li, ELDA: Towards Efficient and Lightweight Detection of Cache Pollution Attacks in NDN, IEEE LCN 2015.
[14] C. Yang, H. Li, L. Wang, and D. Tang, Exploring the Behaviors and Threats of Pollution Attack in Cooperative MEC Caching, IEEE WCNC 2018.
[15] L. Yao, Z. Fan, J. Deng, X. Fan, and G. Wu, Detection and Defense of Cache Pollution Attacks Using Clustering in Named Data Networks, IEEE Trans. Dependable and Secure Computing, early access, Oct. 2018.