Vol.2013-DPS-154 No.45
Vol.2013-CSEC-60 No.45
2013/3/15

ショートホップトレースバック方式の提案

古川　真之[1,a)]　双紙　正和[1]

概要：インターネットの普及に伴い，悪意のあるユーザからの攻撃が脅威となっている．特にサービス不能攻撃 (DoS 攻撃) サービス不可能攻撃 (DDoS 攻撃) が深刻な問題となっている．その対策の一つとして，IP トレースバックが研究されている．特に，パケットマーキング方式とロギング方式を組み合わせたハイブリッド IP トレースバックが有望視されている．本研究では，Yaar らが提案した RIHT よりも，効率の良い全く新しい方式を提案する．さらに，提案方式の評価をするために，シミュレーション実験を行い評価した．

キーワード：DoS 攻撃，DDoS 攻撃，IP トレースバック

A Short Hop IP Traceback Scheme

Masayuki Furukawa[1,a)]　Masakazu Soshi[1]

Abstract: In recent years DDoS (distributed denial of service) attacks have been serious threats to security of the Internet. To counter such attacks, in this paper we propose a short hop IP traceback scheme, which is a kind of hybrid traceback schemes and tries to trace back attackers by repeatedly recovering attack paths in short hops and going closer to attackers gradually. Furhermore in this paper we conduct simulation experiments in order to evaluate efficiency of our traceback protocol.

Keywords: DoS Attack, DDoS Attack, IP traceback

1. はじめに

近年，インターネットの急速な普及に伴い，悪意のあるユーザからの攻撃が脅威になっており，ネットワークセキュリティの必要性が高まっている．被害をもたらす攻撃の中で，サービス不能攻撃（Denial of Service Attack : DoS 攻撃），分散サービス不能攻撃（Distributed Denial of Service Attack : DDoS 攻撃）が深刻な問題となっている．それらの攻撃は，攻撃者（Attacker）が特定のサーバ (Victim) へ大量の不正なパケットを送信することで，サービスを停止させる攻撃である．

現状では，DoS 攻撃，DDoS 攻撃に対する有効な対策が確立されていない．なぜならば，DoS 攻撃，DDoS 攻撃では，攻撃者の数が数千から数万と大規模であり，それに加えて，攻撃者はパケットの IP ヘッダ中の “Source IP Address” を容易に偽造可能であるためである．

現在，発信元の IP アドレスが偽造されている場合においても，攻撃元を特定可能にする技術である IP トレースバックが研究されている．IP トレースバックとは，受信したパケットの情報をもとにパケットの発信元を特定する技術である．代表的なものとして，ルータがパケットに情報を書き込むマーキング方式 [1] やルータがパケットのログを取るロギング方式 [2]，および，それらを組み合わせたハイブリッド方式 [3] が挙げられる．

本研究では，2012 年に Yang らが提案したハイブリッド IP トレースバック方式の一つである RIHT に着目し，より効率の良い全く新しい手法を提案，評価を行う．

[1] 広島市立大学
Hiroshima City University
[a)] furukawa@sos.info.hiroshima-cu.ac.jp

Vol.2013-DPS-154 No.45
Vol.2013-CSEC-60 No.45
2013/3/15

R_i	ルータ
$D(R_i)$	R_i に隣接するルータの数
UI_i	ルータに接続するインターフェース番号

表 1 記号
Table 1 Notations

2. 関連研究

2.1 RIHT の概要

2012 年に Yang らによって RIHT[4] が提案された．これは，ルータがパケットに情報を書き込むマーキング方式と，パケットのログを取るロギング方式を組み合わせた方式であるハイブリッド方式のトレースバックプロトコルの一つである．この手法では，1 パケットで経路全体を正確にトレースバックを行うことを考えている．特徴として False Negative，False Positive が 0 であることが挙げられる．False Negative とは，攻撃経路であるにも関わらず攻撃経路でないと判定される割合で，False Positive とは攻撃経路でないにもかかわらず，攻撃経路であると判定される割合である．

RIHT を説明するにあたり使用する記号を表 1 に示す．

以下の 2.1.1 節でマーキング，ロギング方法，2.1.2 節でトレースバック方法，2.1.3 節で RIHT の問題点を示す．

2.1.1 マーキング，ロギング方法

本節では，RIHT のマーキング，ロギング方法を説明する．

ルータがパケットを受信する．そのとき，ルータが $mark_{new} = P.mark \times (D(R_i)+1) + UI_i + 1$ を計算する．もし $mark_{new} < 255$ なら，ルータは $mark_{new}$ を $P.mark$ に上書きし次のルータへ転送する．もし $mark_{new} > 255$ なら，ルータが $P.mark$ と UI_i のログを取る．ログの処理は，まず，2 次関数探査アルゴリズムを用いて $index$ を計算する．もし，ハッシュテーブルの $index$ に対応する $mark$, UI が空のとき，ハッシュテーブルの $index$ に対応する $mark$, UI に $P.mark$, UI_i をに挿入する．ハッシュテーブルの $index$ に対応する $mark$, UI が空でないときは，再度 $index$ 計算し，ハッシュテーブルの $index$ に対応する $mark$, UI が空のところを探す．$mark_{new} = index \times (D(R_i)+1)$ を計算する．$P.mark$ に $mark_{new}$ に上書きを行い，次のルータへ転送する．同様の処理を繰り返し行う．

図 1 を用いて RIHT のマーキング，ロギング方法を説明する．図 1 中の，矢印は Attaker が送信したパケットの流れ，R_1 から R_7 はルータを示す．R_1 は，$P.mark = 60$ のパケットを受信し，$mark_{new}$ を計算する．$mark_{new} = (60 \times 4 + 1 + 1) = 242(< 255)$ となるので，R_2 に転送し，$mark_{new}$ を計算する．$mark_{new} = (242 \times 4 + 1 + 1) = 970(> 255)$ となるので，R_2 にパケットのログを取る．ログの処理は，まず $index$ を計算する（今回は $index$ を 8 であると仮定する）．ルータ R_2 のハッシュテーブルの $index(= 8)$ に対応する $mark$，UI が空であるので，$P.mark$，UI_2 を挿入する．再び $mark_{new}$ を計算する．$mark_{new} = (8 \times 4) = 32(< 255)$ であるので，R_3 に転送し，$mark_{new}$ を計算する．$mark_{new} = (32 \times 5 + 2 + 1) = 163(< 255)$ であるので次のルータへ転送する．

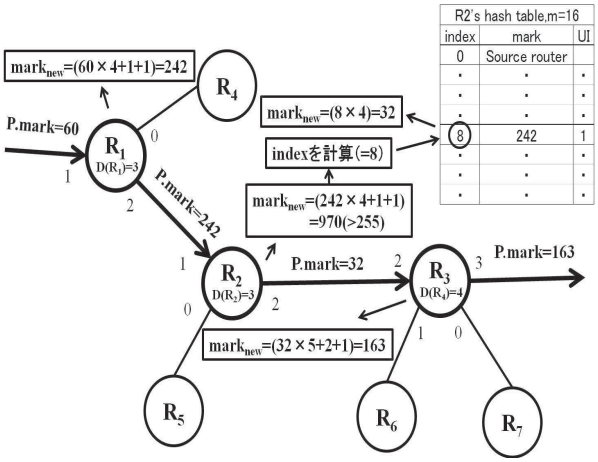

図 1 マーキング，ロギングの様子
Fig. 1 Example of RIHT's marking and logging

2.1.2 トレースバック方法

本節では，RIHT のトレースバック方法を説明する．$UI_i = mark_{req} \bmod (D(R_i)+1) - 1$ を計算する．$UI_i = -1$ の時，$index = mark_{req}/(D(R_i)+1)$ を計算する．$index = 0$ なら，そこが攻撃元である．$index \neq 0$ なら，ハッシュテーブルの $index$ に対応する $mark$ と UI それぞれを，$mark_{req}$, UI_i に更新し，トレースバックを行う．$UI_i \neq -1$ のとき，$mark_{old} = mark_{req} \bmod (D(R_i)+1)$ を計算し，トレースバックを行う．

図 2 を用いて RIHT のトレースバック方法を説明する．図 2 中の，矢印は Victim からのリクエスト，R_1 から R_7 は，ルータを示す．R_3 は，$mark_{req} = 163$ というリクエストを受信し，UI_3 を計算する．$UI_3 = 163 \bmod (4+1) - 1 = 2(\neq -1)$, $mark_{old} = 163/(4+1) = 32$ となるので，R_2 にトレースバックを行う．R_2 では，$UI_2 = 32 \bmod 4 - 1 = -1$ となる．したがって R_2 ではログを取っていることになるので，$index$ を計算する．$index = 32/4 = 8$ となり，R_2 のハッシュテーブルの $index$ に対応する $mark$ と UI を参照し，$mark_{req} = 242$, $UI_2 = 1$ とし，R_1 にトレースバックを行う．R_1 で UI_1，$mark_{old}$ を計算する．$UI_1 = 242 \bmod 4 - 1 = 1(\neq -1)$，$mark_{old} = 242/4 = 60$ となり，UI_1 にリクエストを送信しトレースバックを行う．

2.1.3 RIHT の問題点

RIHT の問題点として，3 点挙げる．1 点目は，Attacker によるパケット偽造に弱いことである．Attacker はパケットの IP ヘッダ中の”Source IP Address”などのパラメータを容易に偽造可能である．そのため，RIHT では確定的に

Vol.2013-DPS-154 No.45
Vol.2013-CSEC-60 No.45
2013/3/15

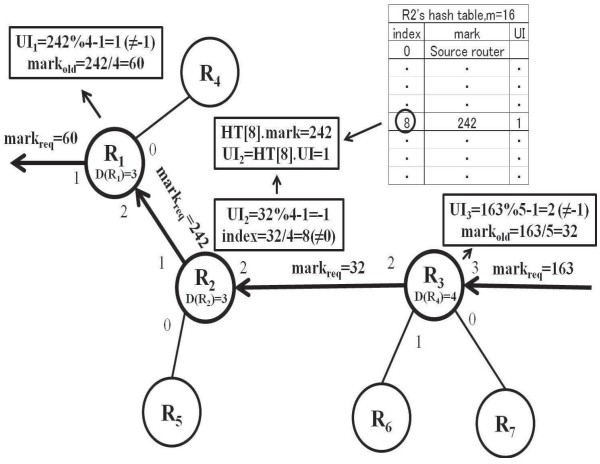

図 2 トレースバック例
Fig. 2 Example of RIHT's path reconstruction

マーキングやロギングを行っているので，Attacker はパケットのパラメータを偽造することで任意のルータに攻撃を仕掛けることが可能である．

2 点目は，動的な状況に対応できないことである．DoS 攻撃や DDoS 攻撃によって，ルータがマーキングやロギングに失敗，ルータが高負荷状態となることによってダウン，ルーティングが急に変わった時には対応できない．

3 点目は，ルータの負荷が高くなることである．RIHT では，パケットがルータを通過するたびにマーキングを行うことで，1 パケットでトレースバックを行うことを考えている．しかし，DoS 攻撃，DDoS 攻撃では数千，数万パケットが送信されるので，ルータへの負荷が非常に高くなってしまうことが考えられる．

3. 提案方式

3.1 概要

本研究では，ラフにトレースバックを行う効率の良い方式を提案する．RIHT では，パケットがルータを通過するたびに，マーキング，ロギングを行うことで，経路全体を正確にトレースバックを行っている．本方式では，短い経路を少しづつトレースバックすることを考える．パケットに全ての情報を詰め込むのではなく，出来るだけ有効利用し，効率よくトレースバックを行う．

以下の 3.1.1 節でマーキング，ロギング方法，3.1.2 節でトレースバック方法，3.1.3 節で Bloom filter を説明する．

3.1.1 マーキング，ロギング方法

本節では，提案方式のマーキング，ロギングアルゴリズム（Algorithm1）について説明をする．

あらかじめルータのマーキング確率 p，カウンタ c を設定しておく．パケットを受け取ったルータは，カウンタ c の値から 1 減らす．カウンタ c の値が 0 または，確率 p でマーキングするならば，Bloom filter に登録し，カウンタ c をセットし直して次のルータへ転送する．そうでなければ

Algorithm 1 markig and logging scheme

```
for each Packet P do
  r ←R [0,1)
  if P.counter == 0 or r < p then
    insert P into Bloom filter
    P.counter ⇐ counter
  else
    P.counter ⇐ P.counter − 1
  end if
  forword P to the next router
end for
```

次のルータへ転送する．**図 3** の基本的な例を用いて，マーキング，ロギング処理について説明する．図 3 中の S は Attacker，V は Victim，R_1，R_2，R_6，R_8，R_9 はロギングルータ，R_3，R_4，R_5，R_7 はマーキングルータ，実践の矢印は Attaker から Victim に向けて送信したパケットの流れを示す．

カウンタ c は 2 であると仮定する．S → R_1 → R_4 → R_8 → V の経路についての処理を説明する．最初に，カウンタ c を 2 にセットする．ルータ R_1 はロギングルータであるので，ルータ R_1 の Bloom filter にパケットのログを取り，再びカウンタ c に 2 をセットし，ルータ R_4 に転送する．ルータ R_4 はマーキングルータであるので，カウンタ c から 1 減らし，ルータ R_8 に転送する．ルータ R_8 はロギングルータであるので，ルータ R_8 の Bloom filter にパケットのログを取り，カウンタ c に 2 をセットし，V へ転送する．その他の経路も同様にマーキング，ロギング処理を行う．

提案方式では，カウンタ c，マーキング確率 p を任意にセットし，マーキング，ロギングを行う．例えば，カウンタ c の値を 3 にセットしたならば，3 ホップまたは，マーキング確率 p でパケットのログを取るので，2 ホップから 3 ホップで必ずログを取ることになる．カウンタ c とマーキング確率 p の最適値を求めることによって，ルータの負荷の軽減や効率良くトレースバックを行うことが可能であると考えられる．

3.1.2 トレースバック方法

本節では，提案方式のトレースバックアルゴリズム（Algorithm2）について説明をする．なお，Algorithm2 中の，L_v はトレースバックに使うパケット集合を示し，また，Algorithm2 は各ルータ R で実行しているものとする．

Victim はカウンタ c をセットし，受信した全てのパケットを上流ルータにフラッディングを行う．受け取ったルータは自分の Bloom filter にパケットのログが見つかれば，自分自身が攻撃経路上にあるルータであると認識し，カウンタ c をセットしなおす．自分の Bloom filter にパケットのログが見つからなければ，カウンタ c から 1 減らす．カウンタ c が 0 になった時，パケットを破棄し終了する．**図 4** の基本的な例を用いて，トレースバック処理を説明する．. 図 4 中の，S は Attacker，V は Victim，R_1，R_2，

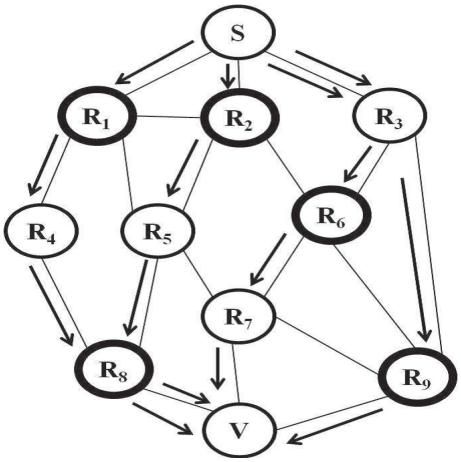

図 3　マーキング，ロギングの様子
Fig. 3　Example of a short hop IP traceback scheme's marking and logging

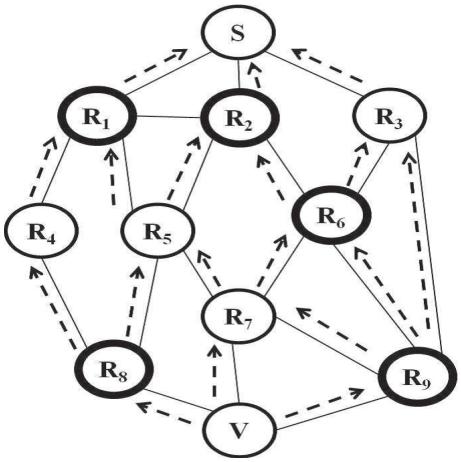

図 4　トレースバックの様子
Fig. 4　Example of a short hop IP traceback scheme's path reconstruction

Algorithm 2 path reconstruction scheme

```
when router R receives traceback packet P with L_v from R'
if R received P before then
  drop P
else
  for each packet Q in L_v do
    if Q is found in the Bloom filter then
      mark R itself as convicted
      break
    end if
  end for
  if R is convicted then
    P.counter ⇐ counter
  else
    P.counter ⇐ P.counter − 1
  end if
  if P.counter == 0 then
    drop P
  else
    broadcast P to adjacent routers other than R'
  end if
end if
```

R_6，R_8，R_9 はロギングルータ，R_3，R_4，R_5，R_7 はマーキングルータ，点線の矢印はフラッディングを示す．

Victim はカウンタ c をセットする．V が上流ルータ（V に隣接するすべてのルータ）にフラッディングを行う．例えば，ルータ R_8 は，自分の Bloom filter にパケットのログが存在するかどうかを調べる．ルータ R_8 はロギングルータであるので，カウンタ c に 2 をセットし，上流ルータにフラッディングを行う．ルータ R_4，ルータ R_5 も同様にして，Bloom filter にパケットのログが存在するかどうかを調べる．ルータ R_4，ルータ R_5 はマーキングルータであるので，カウンタ c から 1 減らし，上流ルータにフラッディングを行う．

提案方式では，Attacker までの経路を，ログを取ったルータまで少しずつたどることでトレースバックを行っている．例えば，カウンタ c を 2 にセットしたならば，必ずそこから 2 ホップ以内にログを取ったルータが存在する．そのログを取ったルータを少しずつたどることでトレースバックを行っている．パケットに書き情報量が少なくし，短いビット長を有効利用することで，効率よくトレースバックすることが可能であると考えられる．

3.1.3　Bloom filter

本節では，Bloom filter について説明する．

Bloom filter[5] とは，効率的に空間を利用できるデータ構造であり，それは，提案方式のパケットのログを取るときに利用する．Bloom filter は，長さ m のビット配列と k このハッシュ関数を用いる．ハッシュ関数はビット配列の *index* を出力する．空の Bloom filter はすべてのビットが 0 である長さ m ビットの配列である．ある要素 x を Bloom filter に追加するには，x を k 個のハッシュ関数に入力し，k 個の結果を得る．この結果が示す *index* のビットを 1 にする．ある要素 y が Bloom filter に存在するかどうかを調べるには，y を k 個のハッシュ関数に入力して得られた k 個の *index* のビットのうち一つでも 0 であれば，その要素は Bloom filter の中に存在するか，あるいは実際に存在しないが他の要素を追加したときに，偶然全部 1 になったために誤判定したかのいずれかである．後者のように False Positive による誤検出はあり得るが，False Negative はない．**図 5** に Bloom filter の動作例を示す．

4.　シミュレーションによる評価

4.1　攻撃のモデル

本章では，提案方式をシミュレーションによって評価する．攻撃パケット数 Np=100000，Attacker から Victim までの距離 d=16，攻撃者数 100 から 2000 でシミュレーションを行った．

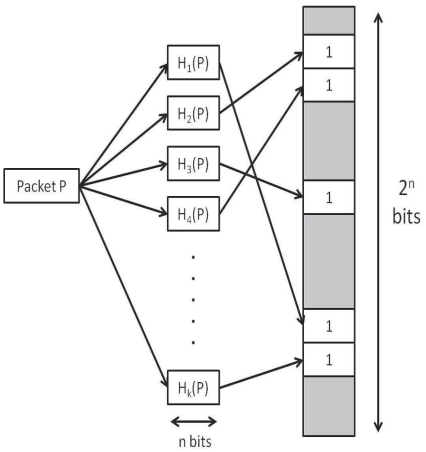

図 5 Bloom filter の動作例
Fig. 5 The process of a Bloom filter with k hash functions

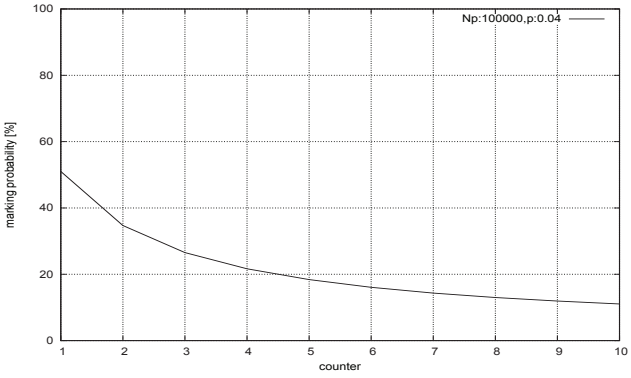

図 6 カウンタ c とマーキング確率の変動
Fig. 6 Counter and marking probability oscillation

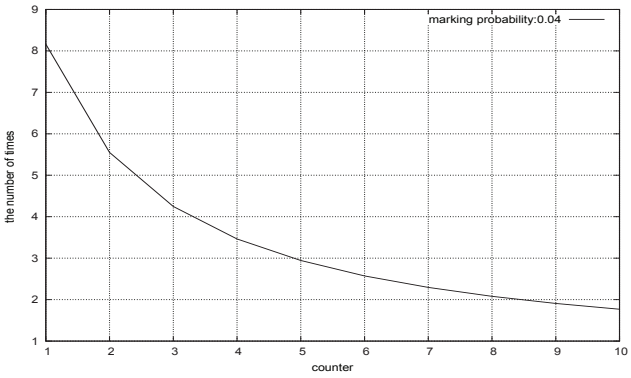

図 7 Attacker から Victim までに 1 パケットがマーキングされる回数
Fig. 7 The number of times of marking per packet from attacker to victim

4.2 カウンタ c の評価

本節では，提案方式におけるカウンタとルータのマーキング確率によるルータ処理の変動，Attacker がパケットを Victim に送信したときにパケットが Victim に届くまでにルータによってマーキングされる回数についての考察をする．

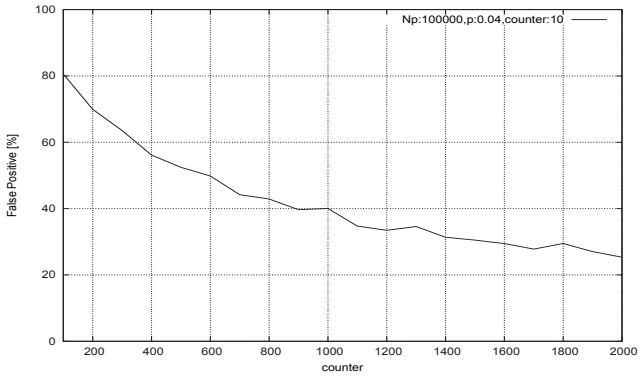

図 8 提案方式における False Positive の評価
Fig. 8 Evaluation of False Positive

図 6 に，ルータのマーキング確率 $p = 0.04$，各カウンタに対応するマーキング確率の変動を示す．縦軸は，Victim にパケットが届くまでにログを取った割合，横軸は，カウンタを示す．図 6 より，カウンタが大きくなるほど，ログを取った割合は少なくなっていることがわかる．これは，カウンタを高くすることで，ログを取る回数が少なくなるためである．

図 7 に，Attacker から送信されたパケットが Victim に到着するまでまでに，ルータが 1 パケットにマーキングした回数を示す．縦軸はルータがパケットにマーキングした回数，横軸は，カウンタを示す．図 7 より，例えば，カウンタが 1 のとき約 8 回，カウンタが 10 のとき約 2 回マーキングしていることが分かる．カウンタとマーキング確率の最適値を求めることにより，ルータへの負荷の軽減しトレースバックを行うことができると考えられる．

4.3 False Positive，False Negative の評価

本節では，提案方式の False Positive，False Negative の評価を行う．False Positive とは，攻撃経路ではないにも関わらず攻撃経路であると判断する割合，False Negative とは，攻撃経路にも関わらず攻撃経路でないと判断する割合である．

図 8 は，提案方式における False Positive を示す．縦軸は False Positive，横軸は，カウンタを示す．図 8 より，攻撃者数が増えるにしたがって，False Positive が減少していることがわかる．これは，Victim が受信した全てのパケットを用いてトレースバックを行うため，トレースバックを行わなくてもよい経路も余分にトレースバックを行ったためであると考えられる．

提案方式における，False Negative の値はいずれの場合も 0 となった．これは，Bloom filter の False Negative が 0 であるため，ルータがログを取っていないパケットも，ログを取ったと判断したことが原因であると考えられる．

5. まとめ

本研究では，ラフにトレースバックを行う効率の良い手法を提案した．経路全体を正確にトレースバックを行うのではなく，パケットのログをとったルータまでトレースバックを行い，それを繰り返す．さらに，提案方式を評価するために False Positive，False Negative の評価を行った．

6. 今後の課題

今後の課題は，提案方式の解析的評価，トレースバックに必要となるパケット数の評価，カウンタとマーキング確率の最適値の評価を行うことで提案方式の性能評価を行うことである．

参考文献

[1] Yaar, A., Perrig, A. and Song, D.: FIT: fast internet traceback, *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, Vol. 2, IEEE, pp. 1395–1406 (2005).
[2] Snoeren, A. C., Partridge, C., Sanchez, L. A., Jones, C. E., Tchakountio, F., Schwartz, B., Kent, S. T. and Strayer, W. T.: Single-packet IP traceback, *IEEE/ACM Transactions on Networking (ToN)*, Vol. 10, No. 6, pp. 721–734 (2002).
[3] Gong, C. and Sarac, K.: A more practical approach for single-packet IP traceback using packet logging and marking, *Parallel and Distributed Systems, IEEE Transactions on*, Vol. 19, No. 10, pp. 1310–1324 (2008).
[4] Yang, M. and Yang, M.: RIHT: A Novel Hybrid IP Traceback Scheme, *Information Forensics and Security, IEEE Transactions on*, Vol. 7, No. 2, pp. 789–797 (2012).
[5] Bloom, B. H.: Space/time trade-offs in hash coding with allowable errors, *Communications of the ACM*, Vol. 13, No. 7, pp. 422–426 (1970).