オペレーションの現場で使う no-Sampling ～DoSアタックの対応について

さくらインターネット(株)
技術部　大久保修一
ohkubo at sakura.ad.jp

前回の発表のおさらい

- JANOG19

  「トラフィックデータ取れるんです - 高精度データ収集編 」

  にて「Non Sampledトラフィック解析の応用例　ー　iDC編」

- no-Samplingの応用例として、
  - DoSアタックリアルタイム検出
  - トラフィック課金システム

  を紹介しました。

「Sampling」vs「no-Sampling」

- Sampling
  - NetFlowやsFlowなど
  - 間引いてトラフィックデータを取得する
- no-Sampling
  - ポートミラーやTAPなどを用いる
  - 全てのデータを取得する

	sampling	no-Sampling
DoSアタック検出	時間が必要	リアルタイムに検出可能
課金システム	誤差が発生	正確な算出が可能

今回の発表では、、、

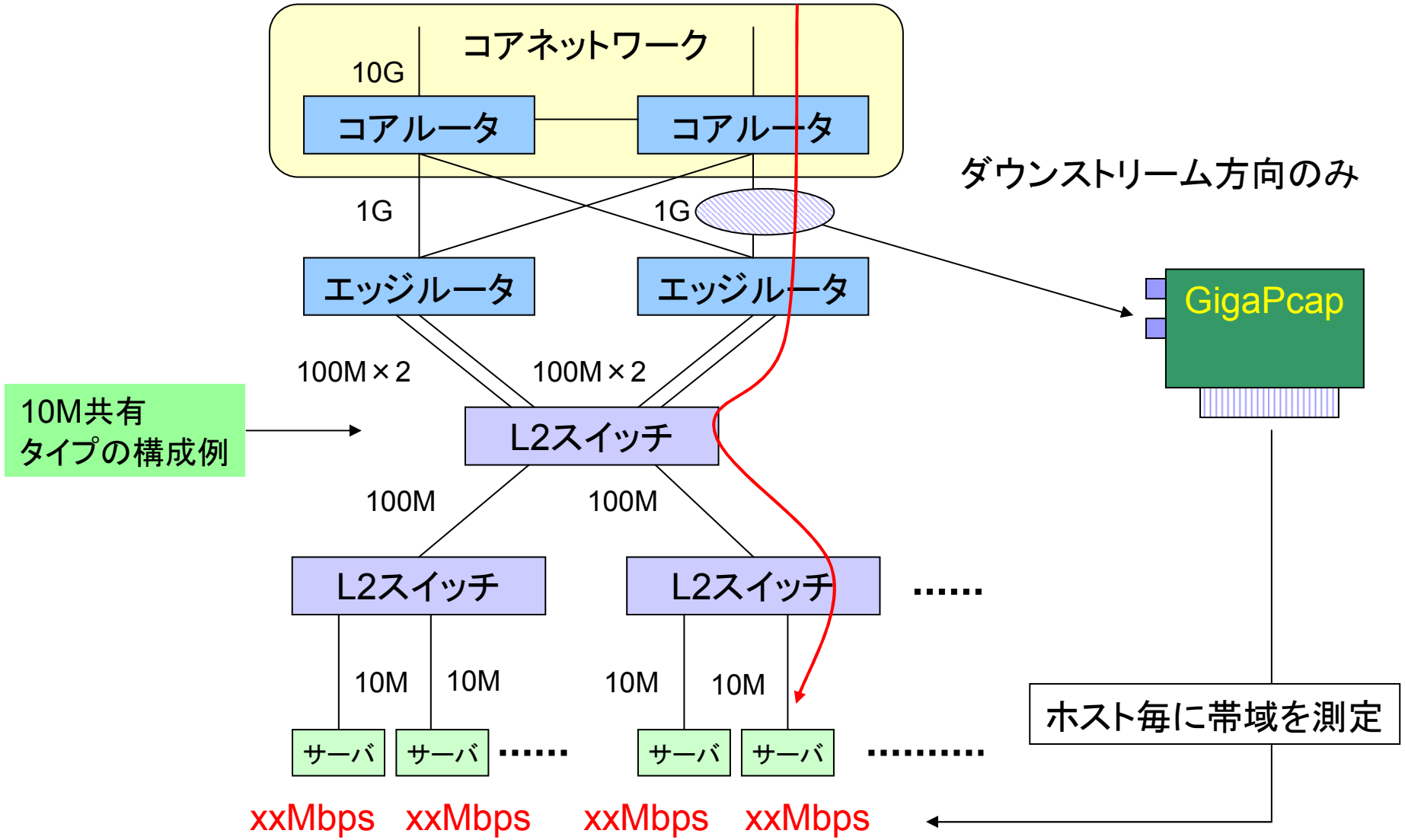
- 今回、JANOG20では、、
  DoSアタックリアルタイム検出の続編として、
  - 検出システムの紹介
  - 検出後の対応について紹介

  をいたします。

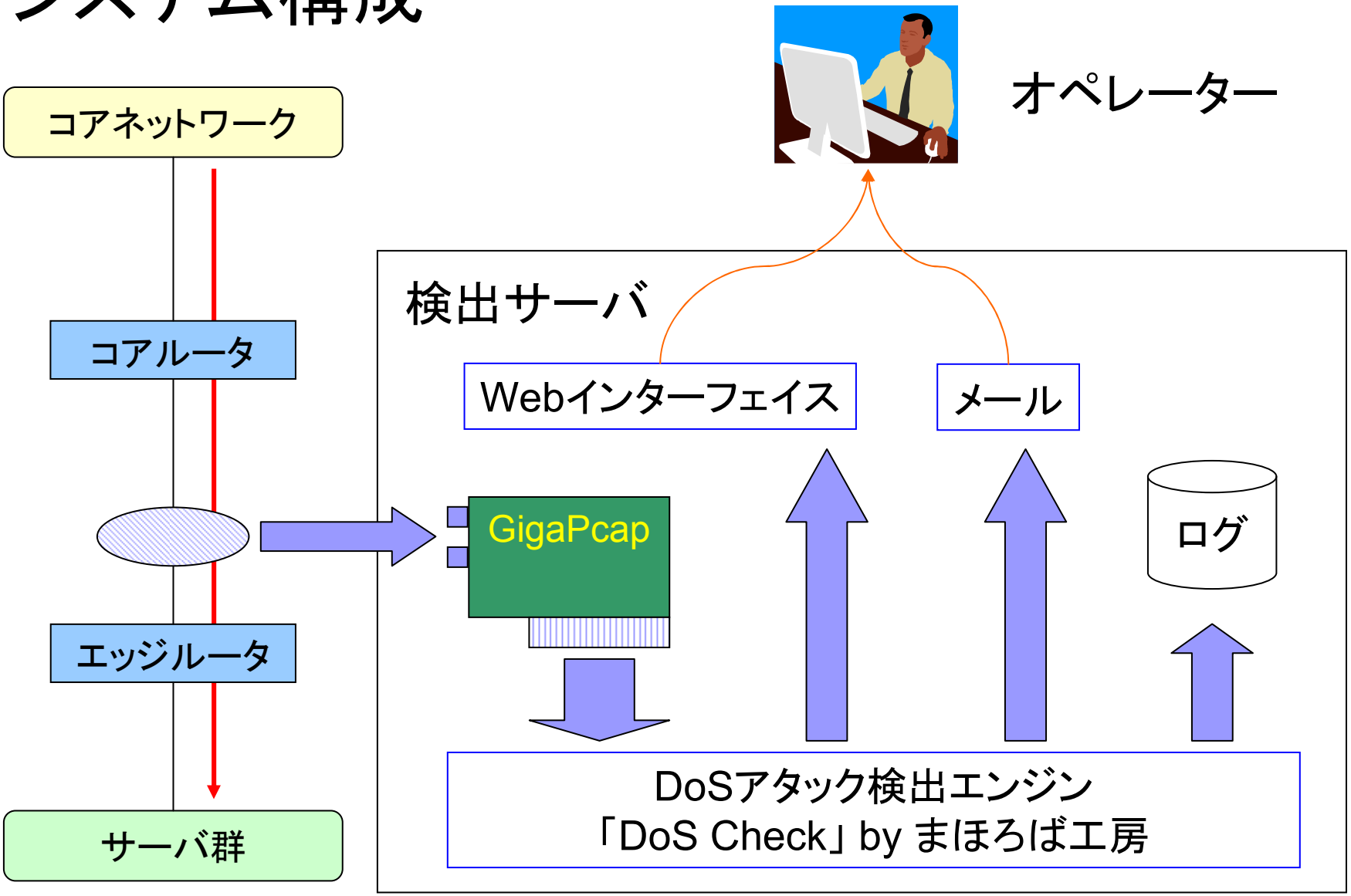
DoSアタックの検出方法

- インターネットから各サーバへ向かうトラフィック量を計測
- 10秒間隔で帯域を算出
- 契約帯域より決められた閾値を超えた場合、
  DoSアタックとみなす

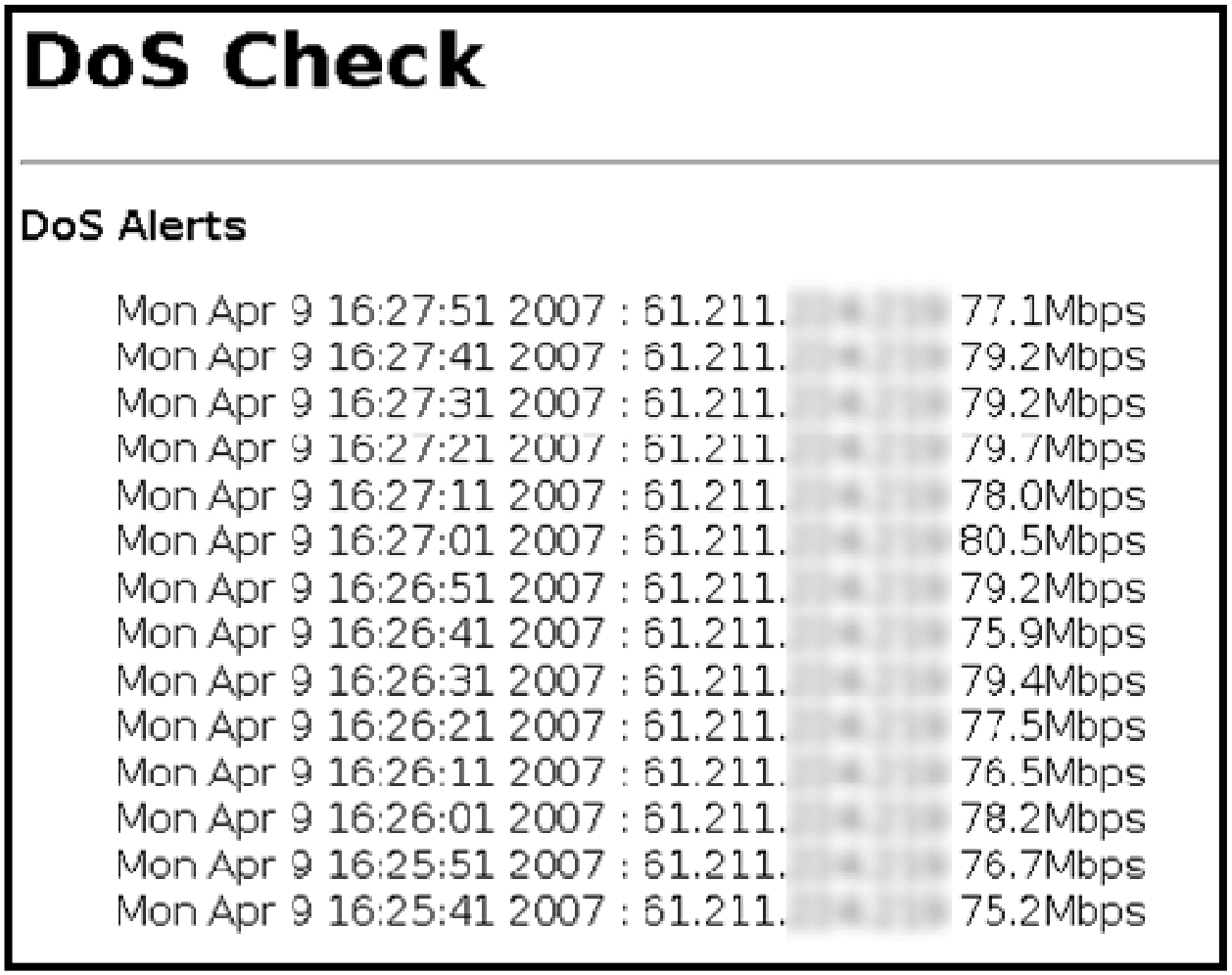
DoSアタックの検出方法



システム構成


コアネットワーク
コアルータ
エッジルータ
サーバ群
オペレーター
検出サーバ
Webインターフェイス
メール
GigaPcap
ログ
DoSアタック検出エンジン
「DoS Check」by まほろば工房

検出例 (Webインターフェイス)



検出例 (発生時メール)

差出人: [redacted]@sakura.ad.jp> 宛先: [redacted]@sakura.ad.jp
件名: DoSアタック検出 (61.211.[redacted])

From: [redacted]@sakura.ad.jp>
To: [redacted]@sakura.ad.jp
Subject: DoSアタック検出 (61.211.[redacted]

DoSアタックを検出しました。

● DoSアタックの内容

検出日時 :　　2007/04/09 16:25:41
宛先IPアドレス : 61.211.[redacted]
帯域 :　　77.1Mbps (閾値 20.0Mbps)

● null routingによる対応方法

-- [redacted].bb.sakura.ad.jpにて --

下記config投入
config term
  ip route 61.211.[redacted]/32 null0
  exit

sho ip route 61.211.[redacted]

検出例 (収束時メール)

差出人: [redacted]@sakura.ad.jp> 宛先: [redacted]@sakura.ad.jp
件名: DoSアタック収束 (61.211.[redacted])

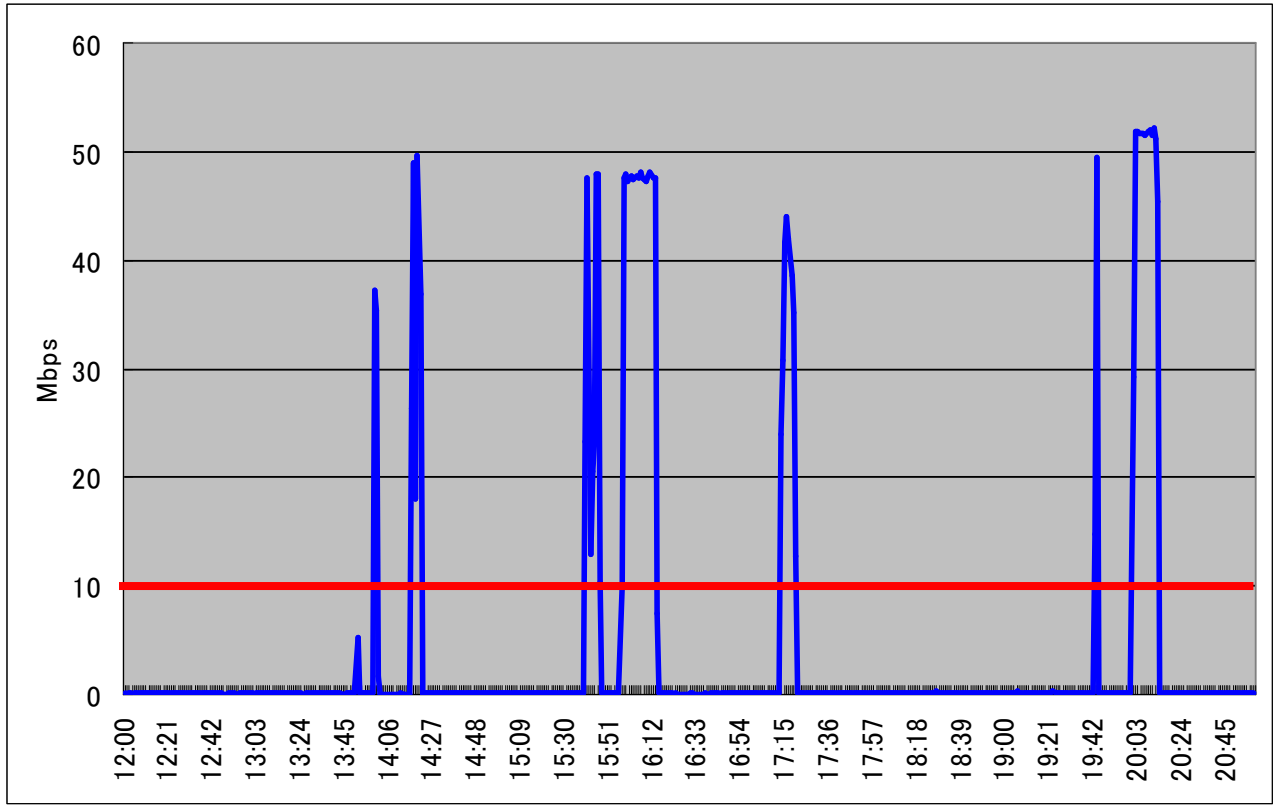
```
From: [redacted]@sakura.ad.jp>
To: [redacted]@sakura.ad.jp
Subject: DoSアタック収束 (61.211.[redacted])

DoSアタックが収束しました。

● DoSアタックの内容

収束日時 :       2007/04/09 16:30:51
宛先IPアドレス : 61.211.[redacted]

● null routingの解除方法

-- [redacted]bb.sakura.ad.jpにて --

下記config投入
config term
  no ip route 61.211.[redacted]/32 null0
  exit

sho ip route 61.211.[redacted]
```

DoSアタック検出後の対応

- 影響範囲が該当ホストのみの場合
  - 基本的にはそのまま様子見
  - 顧客の依頼に基づき、フィルタ設定
- 同セグメントの他のサーバにも影響がある場合
  - 該当ホスト宛の通信をフィルタリング
    - ip route xx.xx.xx.xx/32 null0
  - 該当ホストの契約顧客に連絡
  - 顧客の依頼に基づき、フィルタ設定

実際に発生したDoSアタックの一例

- インターネットからお客様向け
  - 10M共有サービスのお客様です。



10Mbps

トラフィックの中身

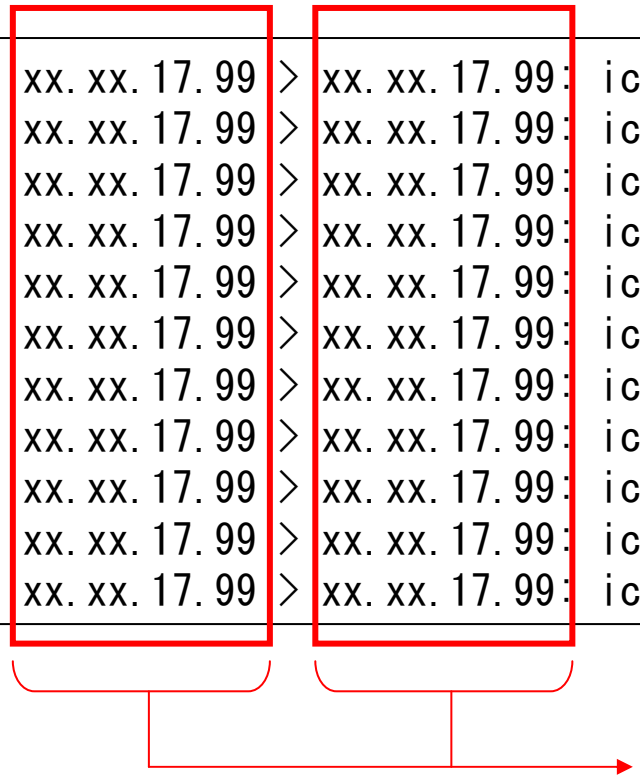
tcpdumpのログより

```
17:18:00.000436 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@12888+)
17:18:00.000489 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@14320+)
17:18:00.000502 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@15752+)
17:18:00.001046 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@17184+)
17:18:00.001064 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@18616+)
17:18:00.001087 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@20048+)
17:18:00.001552 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@21480+)
17:18:00.001630 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@22912+)
17:18:00.001645 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@24344+)
17:18:00.002079 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@25776+)
17:18:00.002090 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@27208+)
17:18:00.002188 xx.xx.162.247 > xx.xx.17.99: icmp (frag 28:1432@28640+)
```

フラグメントアタックのようです。

さらに、、

```
17:18:00.082956 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@2864+)
17:18:00.082969 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@4296+)
17:18:00.083525 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@5728+)
17:18:00.083575 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@7160+)
17:18:00.083589 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@8592+)
17:18:00.083608 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@10024+)
17:18:00.084023 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@11456+)
17:18:00.084052 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@12888+)
17:18:00.084078 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@14320+)
17:18:00.084668 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@15752+)
17:18:00.084680 xx.xx.17.99 > xx.xx.17.99: icmp (frag 39496:1432@17184+)
```

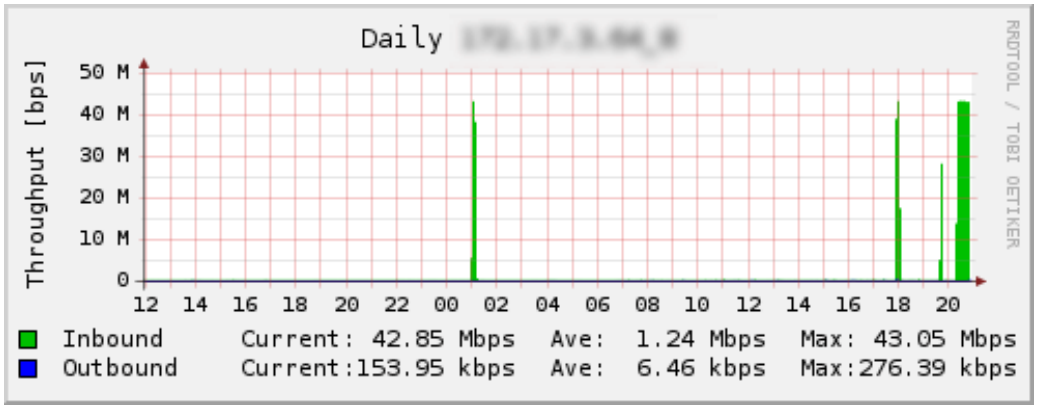


Srcアドレスがspoofされています。。。

他にも。。

- UDPショートパケット
  - これが一番多い
  - サーバのパフォーマンス低下が大きい
  - ネットワーク輻輳時、パケットロス率が高い

サーバのMRTGグラフ




```
20:52:38 IP xx.xx.204.93.3707 > xx.xx.183.111.35120: UDP, length: 1
20:52:38 IP xx.xx.204.93.3707 > xx.xx.183.111.36181: UDP, length: 1
20:52:38 IP xx.xx.204.93.3707 > xx.xx.183.111.50509: UDP, length: 1
20:52:38 IP xx.xx.204.93.3707 > xx.xx.183.111.57473: UDP, length: 1
20:52:39 IP xx.xx.204.93.3707 > xx.xx.183.111.22874: UDP, length: 1
20:52:39 IP xx.xx.204.93.3707 > xx.xx.183.111.14203: UDP, length: 1
```

lengthが1

ネットワーク全体への適用を考える

- 現在のエッジルータの台数：90台
  →アップリンク回線180本
- GigaPcap 180台必要
  →エッジルータのアップリンクの監視はスケールしない
- 1台〇百万円として、180台だと〇億円。
  →コスト的に厳しい。。

- もっと上位ネットワークで監視できるといいかも？
- 10Gコアネットワークで。
- 10G対応のGigaPcap??

まとめ

- 障害発生（DoSアタック発生）から障害復旧（フィルタ対応）までの時間短縮が可能
- 見逃しがちな少量のDoSアタックについても検出可能
- 極短時間(数十秒レベル)のDoSアタックも検出可能に
- フィルタ対応については注意が必要
  - DoSアタック発生後、すぐに通信をとめてしまうのはちょっと。。。
  - アタックの規模と影響範囲を判断しながらフィルタ設定を行う必要あり。
- no-Samplingは便利
  でも、、、ネットワーク全体に適用するのは大変。。