社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－CSEC－39 (2)
2007／12／14

トラフィックフロー分析に耐性があるトンネリング手法の分析 ～パケット長と送信間隔の検討～

三村 守† 中村 康弘†

†防衛大学校 情報工学科
239-8686 神奈川県横須賀市走水 1-10-20

g45042@nda.ac.jp yas@nda.ac.jp

あらまし ネットワークの利用状況を把握するために，パケットを傍受してトラフィックを分析する様々な技術が開発されている．特に最近では分析すべきトラフィックの伝送速度および伝送量は増加傾向にあるため，それに対応できる低負荷なトラフィック分析手法が提案されている．これらの手法ではペイロードを走査する必要がないため，暗号化された通信の分析にも利用することが可能である．このような暗号通信の分析技術は，安全な通信経路を構築するための脅威となりえる．そこで，著者らは暗号通信に対するトラフィックフロー分析の対策として，パケット長と送信間隔を隠ぺいすることにより分析に耐性があるトンネリング手法を提案した．本稿では，設定したパケット長と送信間隔に対するスループットおよびパケット数を測定し，その特徴を分析するとともに，最適なパケット長と送信間隔を検討する．また，TCP を利用した実装についても検討する．

Analysis of Tunneling Against Traffic Flow Analysis ～An Examination of the Packet's Length and the Transmission Intervals～

Mamoru Mimura† Yasuhiro Nakamura†

†Department of Computer Science, National Defense Academy
1-10-20, Hashirimizu, Yokosuka-shi, Kanagawa 239-8686, Japan

g45042@nda.ac.jp yas@nda.ac.jp

Abstract Currently, some traffic analysis technologies with packet capturing are being developed to observe the usage of the network, because of the rapid increase in the speed and amount of recent traffic. For this reason, some low load methods are proposed. These methods can analyze encrypted traffic without scanning the payload. At the same time, these methods can be a threat to construct secure communication route by encryption. We proposed a method and design of a tunneling application that hides packet's length and transmission intervals against the traffic analysis. This report measures throughput and packet's number at several packet's length and intervals, analyzes the characteristic, and investigates the optimum packet's length and transmission intervals.

1 はじめに

ネットワークトラフィックを分析することは，LAN管理のための異常検知や利用者のアプリケーション使用状況を把握する上で重要な課題であり，様々な技術が開発されている．従来のトラフィック分析によるプロトコルやアプリケーションの識別技術は，ペイロードに含まれる特定の文字列の検出やポート番号等のヘッダ情報を利用する手法が主流であった．しかし近年，トラフィック量やアプリケーションの種

類は増加し，NAT やトンネリングによるポート番号の変換や暗号化された通信は広く一般に普及している．このためポート番号とプロトコルやアプリケーションを関連付けることは困難となり，ペイロードを走査することも難しくなってきた．そこで，ペイロードを走査せず，パケット長やパケット送受信のタイミングなどのトラフィックフローを分析することによる，低負荷で高速なプロトコルやアプリケーションの識別技術が研究されるようになった．これらの技術は反対に，悪意ある第三者が盗聴した暗号通信を分析し，プロトコルやアプリケーションを識別する目的でも利用可能である．よってこれらの技術は，プロトコルやアプリケーションを第三者に対し秘匿化することが求められるような，機密性の高い暗号通信にとっては脅威となりえる．トラフィック分析に対する対策としては，カプセル化の際にランダムな長さのパディングを挿入するなどの手法が提案されている [1]．著者らはさらにパケットの送信間隔に着目し，パケット長と送信間隔を隠ぺいすることにより分析に耐性があるトンネリング手法を提案した [2]．また，提案手法に基づくトンネリングアプリケーションを実装した．本稿では，実装したトンネリングアプリケーションを使用し，設定したパケット長と送信間隔に対するスループットおよびパケット数を測定し，その特徴を分析するとともに，最適なパケット長と送信間隔を検討する．また，ファイアウォールや NAT ルータを経由した場合にも，柔軟にトンネリングアプリケーションを動作させるため，TCP を利用した実装についても検討する．以下，第 2 章ではトラフィックフロー分析に耐性があるトンネリング手法の設計と実装について述べる．第 3 章ではパケット長を変更した場合と送信間隔を変更した場合のスループットおよびパケット長を測定し，パラメータの最適値を検討する．また，TCP を利用した実装についても検討する．最後にまとめと今後の課題を述べる．

2 トラフィックフロー分析に耐性があるトンネリング

本章ではトラフィックフロー分析に耐性があるトンネリングアプリケーションの設計と実装について説明する．

2.1 設計

トラフィックフロー分析に耐性があるトンネリングアプリケーションの動作概要を図 1 に示す．左側の送信側トンネリングゲートウェイは到達したパケットを固定長に分割し，送信バッファに格納する．送信バッファに格納した固定長の分割パケットを，一定時間ごとに右側の受信側トンネリングゲートウェイに送信する．受信側トンネリングゲートウェイは受信した分割パケットを受信バッファに格納し，格納した分割パケットから元のパケットを復元し，宛先ホストへ中継する．トンネリングゲートウェイにはパラメータとして，分割パケット長と送信間隔を設定する．カプセル化は，IP ヘッダ以降を固定長に分割してペイロードとし，新たに受信側トンネリングゲートウェイへ到達するためのヘッダを付加して行う．パケットの復元はその逆の手順で実施する．受信したパケットが設定パケット長より短い場合や，固定長への分割の最後に生じる端数長のパケットにはパディングを行い，固定長にカプセル化する．

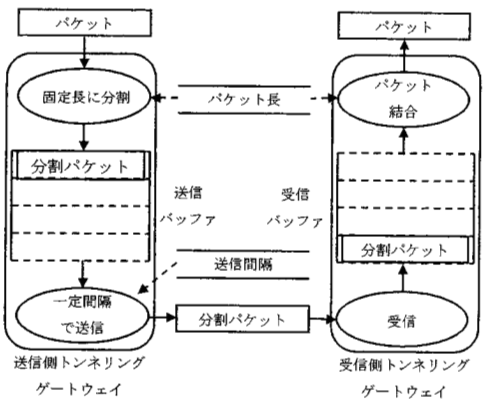

図 1: トラフィックフロー分析に耐性があるトンネリング

2.2 実装

設計したトラフィックフロー分析に耐性があるトンネリングアプリケーションを，C 言語を用いて Linux-2.6 システムを搭載した PC に実装した．さらに，トンネリングゲートウェイ間の通信に TCP を利用する機能を追加した．図 2 を用いて実装について説明する．実装したトンネリングゲートウェイはク

ライアントおよびサーバから構成され，トンネリングゲートウェイの外側と内側に2つのインタフェースを備えている．外側のインタフェースからはストリームもしくはデータグラムソケットを利用し，内側のインタフェースからはLinux固有のパケットソケットを利用してパケットの送受信を行う．クライアントおよびサーバ間の通信はTCPまたはUDPを使用し，内側のインタフェースから読み込んだパケットをカプセル化してサーバとクライアント間で送受信する．クライアントとサーバは同一の構造であり，それぞれmain，send queue，receive queueの3つのプロセスが内部で動作する．

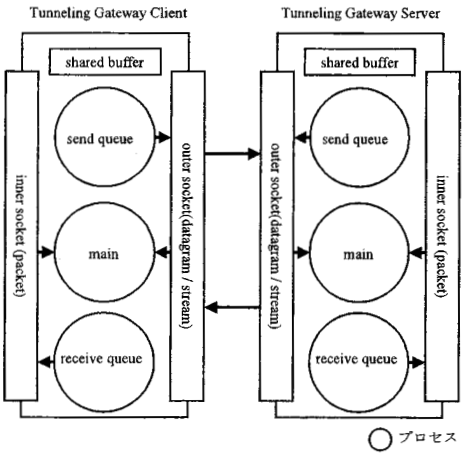

図 2: トラフィックフロー分析に耐性があるトンネリングの実装

2.3 プロセスの動作概要

mainプロセスは内側および外側のソケットを監視し，受信したデータを共有メモリに格納する．send queueプロセスは設定時間が経過するごとに共有メモリを監視し，送信すべきデータがあれば外側のソケットに書き込む．receive queueプロセスは共有メモリを監視し，受信した先頭ペイロードに含まれるIPヘッダからパケットを復元し，内側のソケットに書き込む．

2.4 動作モード

本稿では次の2つの動作モードを利用して実験を行う．fragmentモードではパケットを固定長に分割する．fragment + delayモードではパケットを固定長へ分割し，さらに送信間隔を均一にする．

3 実験

本章では実装したトンネリングアプリケーションを利用し，パケット長および送信間隔を変更した場合のスループットおよびパケット長を観測し，パラメータの最適値を検討する．

3.1 実験環境

実験環境ネットワーク構成を図3に示す．各ホストおよびトンネリングゲートウェイはLinux-2.6システムを搭載したPCで構成されており，100BASE/Tイーサネットで接続されている．各インタフェースにおけるMTU (Maximum Transmission Unit) はいずれも1500である．2台のトンネリングゲートウェイのカーネルのパケット転送機能は無効に設定されており，実装したトンネリングゲートウェイを動作させることによってホストAとホストBの間で通信が可能となる．

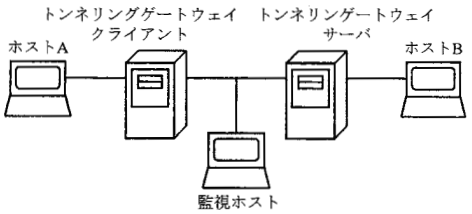

図 3: 実験環境ネットワーク構成

3.2 実験内容

ホストAとホストB間でトンネリングゲートウェイを介して通信を行い，そのパケット数を監視ホストで観測する．通信はsshプロトコルを使用し，ホストAからホストBにscpコマンドで10MBのファイルを転送し，スループットを計測する．

3.3 パケット長の変更

設定パケット長の変更に対し，スループットおよびパケット数がどのように変化するか測定した．ここでは送信間隔の均一化処理は実施しない．図4および図5にパケット長ごとのスループットの変化を示す．図の横軸は設定したパケット長であり，縦軸はスループットを示す．

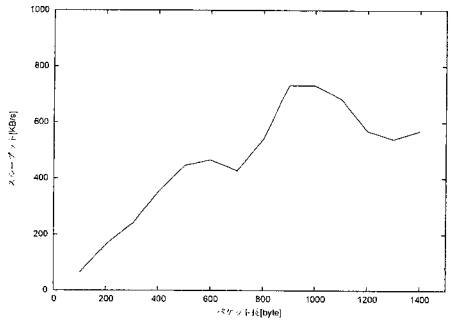

図 4: スループット (UDP fragment)

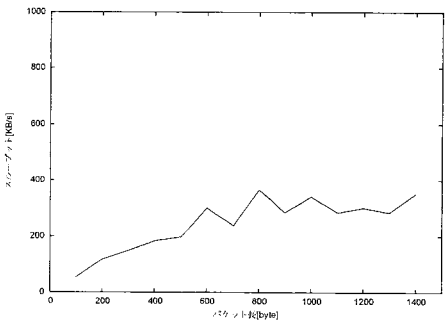

図 5: スループット (TCP fragment)

図6および図7にパケット長ごとのパケット数の変化を示す．図の横軸は設定したパケット長であり，縦軸は観測したパケット数を示す．

スループットについてはやや不安定ではあるが，パケット長を長くするほど向上していることが観測できる．UDP のパケット数についてはパケット長を長くするほど減少しており，意図したとおりにパケットの分割が行われているものと考えられる．しかし，TCP のパケット長についてはパケット長の変化に対し，UDP のような変化を示さなかった．このことから TCP については意図したとおりにパケットの分割が行われていないことがわかる．実装では Nagle アルゴリズム [3] を無効にしているが，ソケットに書き込まれたデータがバッファに蓄積され，直ちに送信されていないものと考えられる．

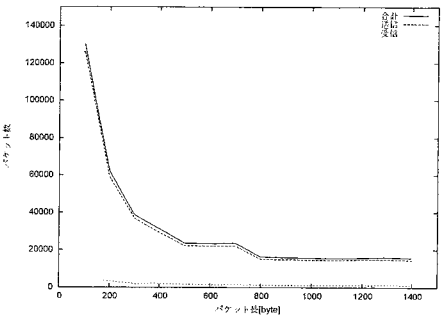

図 6: パケット数 (UDP fragment)

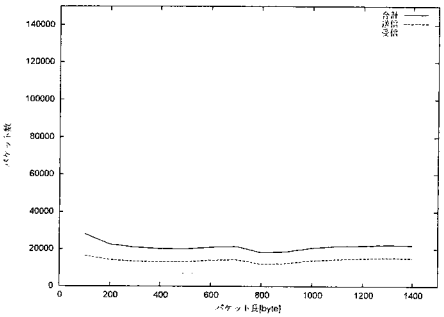

図 7: パケット数 (TCP fragment)

3.4 送信間隔の変更

設定パケット長を 1000byte に固定し，送信間隔の変更に対し，スループットおよびパケット数がどのように変化するか測定した．図8および図9に送信間隔ごとのスループットの変化を示す．図の横軸は設定した送信間隔であり，縦軸はスループットを示す．

図10および図11に送信間隔ごとのパケット数の変化を示す．図の横軸は設定した送信間隔であり，縦軸は観測したパケット数を示す．

スループットについては送信間隔を長くすると，急激に低下することが観測できる．UDP のパケット数は送信間隔に対し，ほぼ一定で変化していない．

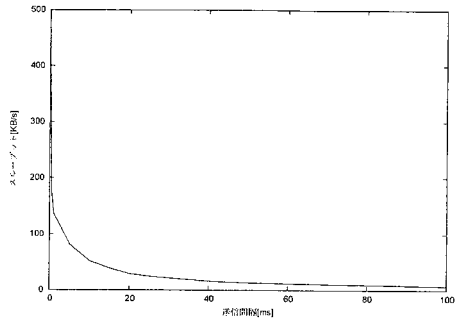

図 8: スループット (UDP fragment + delay)

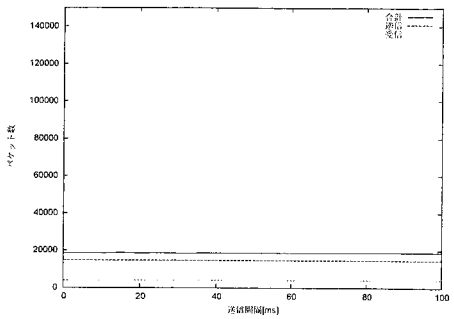

図 10: パケット数 (UDP fragment + delay)

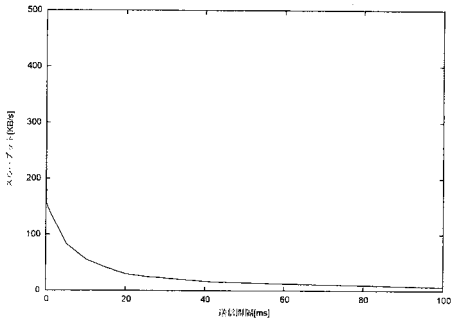

図 9: スループット (TCP fragment + delay)

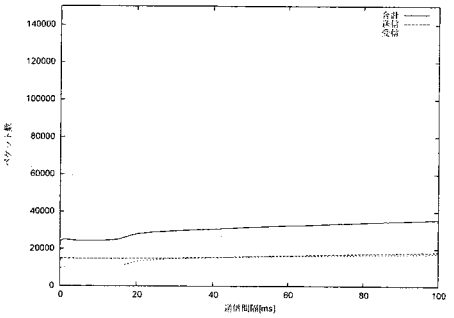

図 11: パケット数 (TCP fragment + delay)

しかし，TCP のパケット数は送信間隔を長くするごとに，緩やかに増加している．さらに，10ms から 20ms の間で受信パケット数がやや急に増加する現象が観測された．この現象は，遅延確認応答のパケット数が増加しているために生じているものと推定される．そこで，この現象による影響を緩和するため，送信間隔を 50ms に固定して設定パケット長を変更し，スループットおよびパケット数を測定する．

3.5 TCP におけるパケット長の制御

TCP において送信間隔を 50ms に固定し，設定パケット長の変更に対し，スループットおよびパケット数がどのように変化するか測定した．図 12 にパケット長ごとのスループットの変化を示す．図の横軸は設定したパケット長であり，縦軸はスループットを示す．

図 13 にパケット長ごとのパケット数の変化を示す．図の横軸は設定したパケット長であり，縦軸は観測したパケット数を示す．

スループットについては，パケット長が 700byte から 800byte に変化したときに急激に増加している．パケット数に関しては，図 6 に近い波形となり，意図したとおりに分割されているものと考えられる．スループットの急激な変化の原因としては，分割数の変化によるパケット数の増加が考えられる．実験環境の MTU は 1500 であり，大容量のペイロードを送信する場合には MTU を最大限に利用したパケットが多数送信される．このため，パケット長が 700byte から 800byte に変化したときに，そのパケットの分割数が 3 から 2 に変化しているものと考えられる．

3.6 考察

以上の実験結果から，極力スループットを低下させないためのパケット長の最適値は，MTU と関係していることがわかる．MTU と設定したパケット長の整数倍の値が近いほど，パディング容量は少な

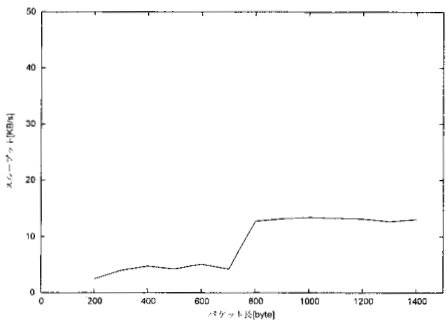

図 12: スループット (TCP fragment + delay)

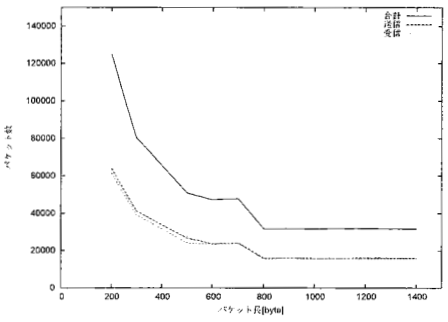

図 13: パケット数 (TCP fragment + delay)

くなり，データを効率的に処理することができる．送信間隔については当然のことながら，短いほどスループットは低下しない．トンネル構築に UDP などのデータグラム指向のプロトコルを使用する場合には，簡単にパケット長を均一に制御することができた．しかし，TCP などのストリーム指向のプロトコルを使用する場合には，Nagle アルゴリズムを無効にしても，パケット長を制御することはできなかった．TCP においてパケット長を制御するためには，ソケットにデータを書き込んだ後，Linux の実装では約 50ms ほど待つ必要がある．しかし，送信間隔を長くするほど，スループットは急速に低下する．このため，TCP ではパケット長を制御しつつ，スループットを向上させることは難しい．

4 おわりに

本稿では，実装したトラフィックフロー分析に耐性があるトンネリングアプリケーションを利用し，検証実験を行ってパケット長を変更した場合と送信間隔を変更した場合のスループットおよびパケット長を測定し，結果を考察した．その結果，UDP においてはパケット長を簡単に制御できることを確認し，パケット長の最適値を検討した．また，TCP においてパケット長を制御するためには，ソケットにデータを書き込んだ後，一定時間待つ必要があることがわかった．

今後の課題は，実装したトンネリングアプリケーションの分析に対する耐性の評価である．パケット長を均一にすることは，多くの分析手法でパケット長を重要な要素として扱っていることから，分析に対して効果があるものと考えられる．しかし，送信間隔については耐性との関係を定量的に評価するのは難しい．実装したトンネリングアプリケーションでは，ミリ秒単位のタイミングは消去できるが，秒単位あるいは分単位のタイミングは消去できるとは言い切れない．提案手法ではこのような長期間のタイミングを消去することは考慮しておらず，ここに提案手法の限界がある．現状では，本稿で想定したパケット長と時間によるトラフィックフロー分析のような，通信実装に対する攻撃の実例は少ない．代表的な通信実装である TCP/IP スタックは通信効率を最大にするように設計されており，通信実装に対する攻撃に耐性を持つように設計されていない．しかし，通信流量と暗号通信は確実に増加しており，今後はこのような通信実装に対する攻撃も考えられ，対策を講じる必要が出てくる可能性がある．

参考文献

[1] Marc Liberatore and Brian N.Levine：Inferring the Source of Encrypted HTTP Connections, Proceedings of the 13th ACM Conference on Computer and Communications Security, pp.255–263 (2006).

[2] 三村 守，中村 康弘：トラフィック分析に耐性があるトンネリング手法の検討，コンピュータセキュリティシンポジウム 2007 (CSS2007) 論文集，4C-3, pp.325–330 (2007).

[3] John Nagle：Congestion Control in IP/TCP Internetworks RFC896 (1984).