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

2007－ＤＳＭ－45（15）
2007／5／11

ペイロード長の遷移パタンを用いたネットワークアプリケーション弁別手法

八木清之介† 和泉 勇治† 角田 裕† 根元 義章†

† 東北大学大学院 情報科学研究科 〒980-8579 宮城県仙台市青葉区荒巻字青葉 6-6-05
E-mail: †{yagi,wai,tsuno,nemoto}@nemoto.ecei.tohoku.ac.jp

あらまし 不正なアプリケーションによるネットワーク利用により，情報流出などの問題が生じている．その対策として，問題のあるアプリケーションの通信を弁別し，遮断することが必要である．しかし，それらのアプリケーションはポート番号を詐称することが可能であるため，ポート番号を用いないフローの統計情報を利用したアプリケーションの弁別手法が提案されている．しかし，これらの手法は統計情報が有意な値を取るまで弁別が行えないため，弁別までに時間を要し，弁別をした時点で既に情報が流出している危険性がある．そこで本稿では通信開始直後におけるペイロード長の遷移パタンに着目し，より早い段階でアプリケーションを弁別する手法を提案する．本稿で提案する弁別手法は，アプリケーションが通信開始直後に定型的な通信を行うことでペイロード長の遷移パタンがアプリケーション毎に類似することを仮定し，その類似性に着目して弁別を行う．また，弁別精度改善のためにペイロード長の逆数を用いた手法を提案する．実際に P2P アプリケーションを含むトラヒックデータを用いた評価実験における，提案手法の弁別性能を報告する．

キーワード アプリケーション弁別, P2P

Classifying Network Applications Using Transition Pattern of Payload-length

Shinnosuke YAGI†, Yuji WAIZUMI†, Hiroshi TSUNODA†, and Yoshiaki NEMOTO†

† Guraduate School of Information Sciences, Tohoku University
E-mail: †{yagi,wai,tsuno,nemoto}@nemoto.ecei.tohoku.ac.jp

Abstract Recently, information leakage caused by illegal use of network applications (e.g. P2P application) has become a new social issue. To prevent information leakage, early detection and blocking of the traffic exchanged by such applications is strongly required. In this paper, we propose a method for discriminating application of monitored traffic based on the transition pattern of payload-length during start-up phase of the communication. The proposed method does not need port numbers, which are easily spoofed, and can quickly discriminate applications compared with the conventional methods using traffic statistics. Through experiments using real network traffic, we show that the proposed method can quickly and accurately discriminate applications including the P2P application.

Key words Application discrimination, P2P

1. まえがき

近年，インターネットを経由した個人情報や機密情報の漏洩事故が相次ぎ，深刻な社会問題となっている．2005 年に報道されたインターネット経由の情報漏洩事故は 130 件にものぼる [1]．中でも特に P2P によるファイル交換ソフトウェアの設定ミスやワームへの感染により，本来公開すべきでない情報が P2P ネットワーク上で共有されてしまう事故が増加している．[2,3] また，P2P ネットワーク経由で感染するトロイの木馬の中には，感染したホスト上で http サーバを起動し，ホスト内の情報をインターネット上に公開するものも存在する．[4] このような情報漏洩の対策として，P2P ファイル交換ソフトウェアやトロイの木馬により起動した http サーバなどの問題の原因となるアプリケーションによるフローをいち早く識別し，情報漏洩が発生する前に遮断する方策が必要である．ここでフローとは，送受信 IP アドレス，送受信ポート番号，プロトコ

ルの 5-tuple の組合せが一致するパケットの集合である．

フローを送受信しているアプリケーションの弁別には，通常，そのフローのポート番号が用いられる．これは一般に，通信で使用するポート番号がアプリケーション毎に固定されているためである．しかし，P2P ファイル交換ソフトウェアをはじめとする近年問題となっているアプリケーションは，通信に利用するポート番号が一定していない．また．Well-known ポートとして一般的に他のアプリケーションが利用するポート番号を利用し，他のアプリケーションになりすますケースもある．従って，ポート番号を用いた弁別手法では，前述の P2P アプリケーションを弁別することは困難といえる．また，そもそもポート番号は通信の当事者が任意に変更可能であるため，ポート番号のみによる弁別は信頼性に欠ける可能性がある．

ポート番号を用いない弁別手法として，ペイロードに含まれる特徴的な文字列より生成したシグネチャを用いる手法 [5] やフロー中のパケットサイズやパケットの到着間隔の平均値や分散などの統計情報に基づいた弁別手法が提案されている [6, 7]．しかし，これらも実用面で問題を抱えている．

そこで本稿では，フローの初期段階におけるアプリケーション弁別を目的として，フロー開始直後のペイロード長の遷移パタンに着目した弁別手法を提案する．本稿で提案する弁別手法は，フロー開始直後の定型的な通信によるペイロード長の遷移パタンがアプリケーションごとに異なる特徴を示すと仮定し，その特徴によりアプリケーションを弁別する．

以下，**2.** ではポート番号を用いないアプリケーション弁別手法とその課題について述べる．**3.** では **2.** で示した課題に対して，通信開始直後におけるペイロード長の遷移パタンの法則性を検証し，その過程でペイロード長の遷移パタンを用いたアプリケーション弁別における問題点を示す．**4.** では **3.** で示した問題点に対する解決策としてペイロード長の逆数を用いた弁別手法を提案し，実際にアプリケーションの弁別を行い提案手法の弁別性能を評価する．**5.** は本稿のまとめである．

2. 関連研究

本章では，ポート番号を用いないアプリケーション弁別の関連研究について述べる．

ポート番号を用いない弁別手法として，文献 [5] ではペイロードに含まれる特徴的な文字列から生成したシグネチャを用いる手法を提案している．しかし，P2P アプリケーションのように通信内容を暗号化するアプリケーションに対しては適用が困難であるため，有効な対策とは言い難い．

文献 [6] では，フローを構成するパケットのサイズおよび到着間隔について平均値，中間値，度数分布，分散といった統計情報を取得し，その情報を用いてアプリケーションを弁別する．また，文献 [7] は統計情報に基づいてフローをクラスタリングするとアプリケーション別のクラスタが形成されることを確認し，いくつかのクラスタリングアルゴリズムを用いて弁別性能の比較検討を行っている．これらの手法はポート番号やシグネチャに依存せず高精度なアプリケーション弁別を実現している．しかし，利用する統計情報が有意な値になるまでに多数のパケットを観測する必要があり，結果として通信開始から弁別までに時間を要するという問題がある．そのため，フローがどのアプリケーションによるものかを早期に識別し，必要に応じてフローを遮断するといった用途への適用は困難である．従って，いち早いアプリケーション弁別のためには．アプリケーション毎の特徴をフローの初期段階でとらえることのできる指標が必要である．

フローの初期段階でアプリケーションの特徴を捉えるために，文献 [8] では，アプリケーションが送受信するパケットのサイズの遷移パタンに着目した分析を行っている．その結果，通信開始直後におけるパケットサイズの遷移パタンがアプリケーション毎に異なることを明らかとした．しかし，この研究においてはアプリケーション毎に異なる遷移パタンが存在することを示すのみで，実際のアプリケーション弁別を行う手法は確立されていない．

そこで本稿では，アプリケーション層で解釈される情報のみを抽出するために，パケットサイズから IP ヘッダ長と TCP ヘッダ長を取り除いたペイロード長を指標として利用し，フロー開始直後におけるペイロード長の遷移パタンに着目したアプリケーション弁別手法を提案する．実際にアプリケーション弁別を行い，その弁別精度を評価する．また，ペイロード長の遷移パタンを用いてアプリケーション弁別を行う際の問題点を指摘し，その解決策としてペイロード長の逆数を用いる手法を提案する．

3. アプリケーション弁別のための ペイロード長の遷移パタンの分析

本章ではアプリケーション毎のペイロード長の遷移パタンを分析し，アプリケーション弁別におけるペイロード長の遷移パタンの有効性を検証する．

3.1 ペイロード長の遷移パタンのベクトル化

多くのアプリケーションでは，通信開始時にソフトウェアのバージョンや利用可能なオプションの告知，ユーザ認証などといった定型的な通信が行われる．これらの通信の手順や通信される内容の書式などはプロトコルによって定められているため，同一アプリケーションのフローであれば同種の内容を含むパケットが同一の順序で送信されると考えられる．そのため，定型的な通信によるペイロード長の遷移パタンはアプリケーションごとに類似すると考えられる．そこで本稿では，その類似性を定量的に評価するためにペイロード長を到着順に並べたベクトル $\vec{p}$ を式 (1) のように定義し，そのベクトル間距離を利用する．同一アプリケーションのフローより取得した $\vec{p}$ は距離が近くなることを仮定し，アプリケーションの弁別を試みる．予め各アプリケーションを代表する参照ベクトルを作成し，観測されたフローから取得した $\vec{p}$ と参照ベクトルとのベクトル間距離に基づきアプリケーションを弁別する．

$$\vec{p} = [p_1, p_2, \ldots p_n]^T \quad (1)$$

$p_i : i$ 番目のパケットのペイロード長

$\vec{p}$ の次元数 n は弁別時に考慮するパケット数と一致する．パ

ケットの送信方向を区別するため，最初に syn パケットを送信したホストが送信するパケットのペイロード長を正，受信するパケットのペイロード長を負とし，$\vec{p}$ の各要素を定義する．また，ペイロードを持たない ack パケットなどのトランスポート層における制御情報の影響を無視するため，ペイロード長が 0 のパケットは $\vec{p}$ の要素としないこととする．

3.2 ペイロード長の遷移パタンの類似性に関する検証

同一アプリケーションのフローより取得した $\vec{p}$ 同士のベクトル間距離は近く，異なるアプリケーションのフローより取得した $\vec{p}$ 同士のベクトル間距離は遠くなると考えられる．

そこでこの仮定の妥当性を検証するために Self-Organizing Map(SOM) [9] により各アプリケーションの $\vec{p}$ の位置関係を可視化し，類似性の評価を行う．

SOM は教師なし学習によってベクトルのクラスタリングを行うアルゴリズムである．SOM により生成される参照ベクトルは 2 次元における隣接関係が定義されており，多次元空間での隣接関係をある程度保持したまま 2 次元に写像することが可能となる．そのため，多次元空間におけるクラスタの分布を 2 次元平面に可視化することが可能である．

そこで本実験では，生成された参照ベクトルを SOM により 16x16 の平面に配置し，$\vec{p}$ の位置関係を可視化することにより，実際に同一アプリケーションのフローから取得した $\vec{p}$ が平面上で隣接するかどうかを確認する．

本実験で使用するデータは，ホスト数 50 程度のネットワークの出入口において観測されたトラヒックデータより抽出した http，smtp，ssh，pop3，imap，pop over ssl(pops)，http over ssl(https)，rtsp の 8 種類のアプリケーションによるフローと，ホスト数 3 の実験ネットワークのトラヒックデータより抽出した winny，share によるフローである．各アプリケーションについて 100 個の $\vec{p}$ を取得した．今回 winny，share のフローを抽出した実験ネットワークにおいては，インターネットに接続された 3 台のホストのうち 1 台で winny，share を起動し，60 分ずつ通信を行った．ただし，このときファイル検索などの操作は行わず，起動したのみの状態でトラヒックを観測した．

$\vec{p}$ の次元数 n を 5 としたときの結果を図 1 に示す．色が濃くなっている部分はベクトル間距離が大きいことを示す．また，アプリケーションのラベルが上下に 2 つ並んでいる場所は，その参照ベクトルが代表する領域に 2 種類以上のアプリケーションの $\vec{p}$ が存在していることを示す．

図 1 より，部分的に複数のアプリケーションが重複して配置されている箇所があるものの，全体としては参照ベクトルがアプリケーション毎にまとまって分布していることが分かる．また，http，https，rtsp が分布する範囲とそれ以外のアプリケーションが分布する範囲の境界の色が濃くなっていることから，これらのアプリケーションの参照ベクトルが他と離れて位置していることが確認できる．

次に $\vec{p}$ の n=3，n=10 の場合を，それぞれ図 2 と図 3 に示す．

図 2 より，pop3，winny などのアプリケーションについては図 1 の n=5 の時よりラベル数が減少しており，同一アプリケーションのフローに共通した特徴がより顕著に出ていることが分

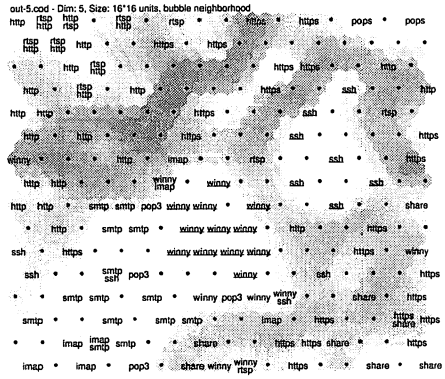

図 1 $\vec{p}$ の分布 (次元数 5)

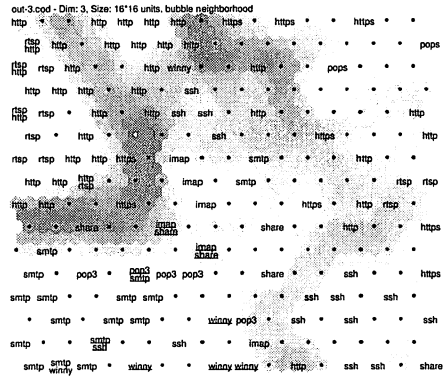

図 2 $\vec{p}$ の分布 (次元数 3)

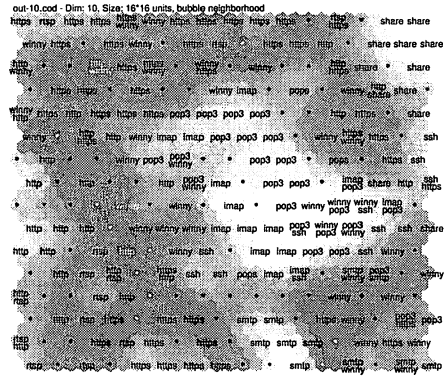

図 3 $\vec{p}$ の分布 (次元数 10)

かる．一方で imap と share など複数のアプリケーションが重複しており，異なるアプリケーション間における特徴の差が十分にあらわれていないことが考えられ，正しい弁別ができない可能性がある．

図 3 では，多くのアプリケーションのベクトルが広い範囲に重複して分布し，アプリケーション毎のクラスタが形成できていないことが分かる．これは，各プロトコルの定型的な通信が終了し，コンテンツの送受信によるパケットが $\vec{p}$ の要素として含まれてしまい，同一アプリケーションのフローであってもコ

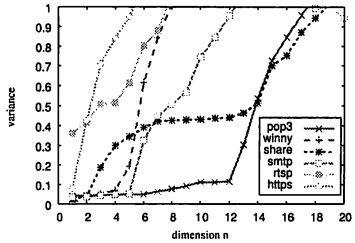

図 4 $\vec{p}$ 同士のベクトル間距離の分散

ンテンツの違いにより類似しない部分が生じてきたためと考えられる．n=10 以上の場合でもこの傾向は変わらず，次元数が増え過ぎると正しい弁別が行えなくなる可能性が考えられる．

続いて，同一アプリケーションの $\vec{p}$ 同士の類似性が次元数によりどう変化するかを確認するため，各アプリケーションの $\vec{p}$ の分布の分散と次元数の関係を調査した．図 4 に同一アプリケーションの $\vec{p}$ の分布の分散と次元数の関係を示す．図 4 の横軸は $\vec{p}$ の次元数，縦軸はアプリケーション毎の分散である．なお，トラヒックを観測したリンクの MTU は 1500 バイトであるため，$\vec{p}$ の各要素はペイロード長の最大値である 1460 で割り，各要素とも最大値が 1 となるように正規化した．

図 4 を見ると，各アプリケーションとも次元数が増えるにつれ分散が増加しており，次元数を増加させると同一アプリケーションのフローであってもペイロード長の遷移パタンに差が生じることが分かる．

これらの結果から，ペイロード長の遷移パタンを用いたアプリケーション弁別において，考慮するパケットの数を増加させると弁別精度が低下する可能性が考えられる．

4. ペイロード長の遷移パタンを用いたアプリケーション弁別手法

前章では，ペイロード長の遷移パタンがアプリケーション毎の類似性を調査し，考慮するパケット数を増加させるとその類似性が低下するという問題点を指摘した．本章ではこの問題点に対する解決策としてペイロード長の逆数を要素とした遷移パタンを用いることを提案する．そして，実際に提案した遷移パタンによるアプリケーションの弁別実験を行い弁別性能を評価する．

4.1 ペイロード長の逆数を用いたペイロード長の遷移パタンの表現

考慮するパケット数の増加とともにペイロード長の遷移パタンの類似性が低下する原因は，定型的な通信が終了し，コンテンツの送受信が始まるためと考えられる．これは，コンテンツの送受信においては，同じサービスのフローであってもコンテンツの内容が異なるために，ペイロード長の遷移パタンが大きく変動するためである．

この問題の対策として，コンテンツの送受信が始まる前に弁別を行えるよう，$\vec{p}$ の次元数を小さくすることが考えられるが，$\vec{p}$ の次元数が少なすぎる場合は異なるアプリケーション間の差が十分に表れず，弁別性能が低下する可能性がある．また，コンテンツの送受信が開始されるタイミングはアプリケーション毎に異なるため，すべてのアプリケーションにおいてコンテンツの送受信が始まる前に弁別を行うことは困難といえる．そのため，コンテンツの送受信による影響を軽減することが必要となる．

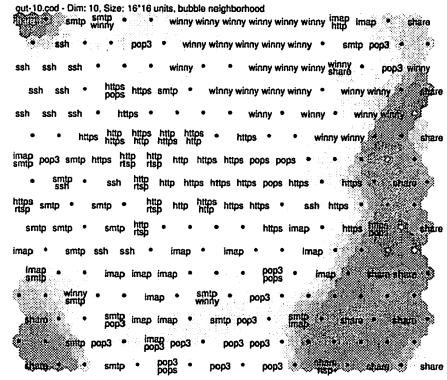

図 5 $\vec{p}_{inv}$ の分布 (次元数 10)

そこで本章では，定型的な部分とコンテンツが送受信されている部分におけるペイロード長のサイズの違いに着目し，$\vec{p}$ の改良を行う．定型的な通信によるパケットはペイロード長が小さい傾向にあり，コンテンツの送受信によるパケットはペイロード長が大きくなる傾向にある．ゆえに，ペイロード長が小さいパケットを重視し，ペイロード長が大きなパケットの影響を軽減することが対策として考えられる．そこで，ペイロード長の逆数を要素として用いたベクトル $\vec{p}_{inv}$ を式 (2) と定義して $\vec{p}$ の代わりに用いることを提案する．

$$\vec{p}_{inv} = [\frac{1}{p_1}, \frac{1}{p_2}, ..., \frac{1}{p_n}]^T \quad (2)$$

逆数を用いることにより，定型的な通信によるパケットのペイロード長の違いが強調され，コンテンツの送受信によるパケットのペイロード長の差は軽減されることとなり，コンテンツの送受信によりペイロード長の遷移パタンに変化が生じた場合にも正しい弁別が行えると考えられる．

3. 同様の方法で $\vec{p}_{inv}$ の類似性を評価する．$\vec{p}_{inv}$ の次元数を 10 とした場合の SOM の出力結果を図 5 に示す．図 5 を見ると，次元数 10 においても各アプリケーションのクラスタが維持されていることが分かる．また、図 6 に，同一アプリケーションの $\vec{p}_{inv}$ の分布の分散を示す．図 6 を見ると，share 以外のアプリケーションにおいて，次元数の増加によるベクトル間距離の増加が図 4 に比べ軽減できていることが分かる．ここで share のベクトル間距離が次元数に伴い大きくなっている原因は，share のペイロード長の遷移パタンが大きく 2 種類に分かれているためと考えられる．図 5 においても share の参照ベクトルが分布する範囲はグレーの領域により上下に区切られており，同一アプリケーションであっても複数の異なる遷移パタンが存在することが分かる．

4.2 アプリケーション弁別実験による弁別性能の評価

以下の手順でアプリケーションの弁別を行い，提案手法の弁

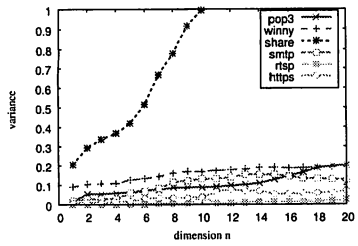

図 6 $\vec{p}_{inv}$ 同士のベクトル間距離の分散

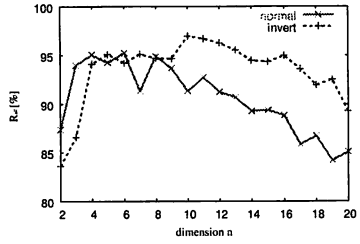

図 7 $\vec{p}$,$\vec{p}_{inv}$ の次元数と弁別精度の関係

別精度を検証する．まず，予め収集した各アプリケーションのフローから $\vec{p}$ および $\vec{p}_{inv}$ を取得し，学習用データとする．学習データの各ベクトルにはアプリケーションのラベルを付加し，Optimized LVQ1 による学習を行い参照ベクトルを作成する．新たに観測されたトラヒックより取得した $\vec{p}$, $\vec{p}_{inv}$ と参照ベクトル間のユークリッド距離を評価し，最近傍にある参照ベクトルと同じアプリケーションを弁別結果とする．

4.3 実験環境

トラヒックデータは **3.** と同様のものを用いた．それぞれのアプリケーションから 300 フローずつを学習データとして用い，その学習データより作成した参照ベクトルにより，学習データとは別に用意した各アプリケーション 200 フローずつをテストデータとして弁別し，その精度を調査した．また，$\vec{p}$, $\vec{p}_{inv}$ の次元数の最大値はフローに含まれるパケット数であることから，フローに含まれるパケット数がテスト時に考慮する次元数に満たないフローはテストデータから除外する．そのため，一部アプリケーションは次元数を増やすにつれテストに用いるフロー数が減少している．

4.4 弁別精度の評価

$\vec{p}$ の次元数 n と式 (3) に定義する弁別精度 R_{all} の関係を図 7 に示す．

$$R_{all} = \frac{\text{正しく弁別できたフロー数}}{\text{全テストフロー数}} \times 100 \qquad (3)$$

図 7 から，$\vec{p}$ を用いた場合に最も精度が高かった $\vec{p}$ の n=6 のときを見ると，全テストフローのうち 95% を正しく弁別できていることがわかり，$\vec{p}$ のベクトル間距離が弁別の指標として有効であることを確認できる．さらに，$\vec{p}$ を用いた場合と $\vec{p}_{inv}$ を用いた場合の弁別精度の変化を比較すると，ベクトルの次元数 n が増加するにつれ $\vec{p}_{inv}$ を用いた方が高い精度で弁別可能なことが分かる．

表 1 アプリケーションの弁別結果

	弁別精度 (6 次元)[%]		弁別精度 (10 次元)[%]	
アプリケーション	$\vec{p}$	$\vec{p}_{inv}$	$\vec{p}$	$\vec{p}_{inv}$
http	88.00	95.00	87.00	95.00
pop	99.00	99.00	98.50	99.00
smtp	98.00	96.50	86.50	96.00
ssh	97.50	98.00	97.50	97.50
imap	99.00	97.50	97.50	97.50
pops	99.50	100.0	99.50	100.0
https	92.00	98.00	85.50	100.0
share	97.00	94.00	97.00	98.50
winny	89.94	89.39	53.66	89.02
rtsp	83.64	23.64	81.82	80.00
total	95.26	94.27	91.36	96.95

また，n=6 と n=10 の場合における各アプリケーションの弁別精度 R_{app} を表 1 に示す．ここで各アプリケーションの弁別精度を式 (4) で定義する．

$$R_{app} = \frac{\text{正しく弁別できたフロー数}}{\text{そのアプリケーションによるフローの総数}} \times 100 \qquad (4)$$

表 1 より，どのアプリケーションに関しても高精度の弁別が可能であることが分かる．また，$\vec{p}$ を用いた場合，次元数 n が増加すると一部のアプリケーションは弁別精度が低下しているが，$\vec{p}_{inv}$ を用いることで精度を改善できることが分かる．

4.4.1 P2P アプリケーションに対する弁別精度の評価

近年特に問題となっている P2P アプリケーションである winny と share に関する弁別精度 R_{P2P} と $\vec{p}$ の次元数 n の関係を図 8 に示す．ここで，図 5 の縦軸は式 (5) で定義される．

$$R_{P2P} = \frac{\text{正しく弁別出来た } winny,\ share \text{ のフロー数}}{winny,\ share \text{ の全フロー数}} \times 100 \qquad (5)$$

図 8 より，winny，share ともに $\vec{p}_{inv}$ が 6 次元以下で 90% 以上の弁別精度を示しており，P2P アプリケーションをフローの初期段階で弁別可能であることが分かる．また，winny に対する弁別精度に着目すると，$\vec{p}$ を用いた場合には $\vec{p}$ の次元数が増加するにつれ精度が急激に低下しているのに対し，$\vec{p}_{inv}$ を用いた場合にはその低下を軽減できている．なお，winny に比べて share の弁別精度が高いのは，share は定型的な通信の段階で送受信されるパケットの数が winny に比べて多いため，弁別に有効な情報が多くなり，コンテンツを含むパケットの影響が小さくなったためと考えられる．

さらに，他のアプリケーションのフローを winny，share に誤認識した割合 R_{FP} と $\vec{p}$,$\vec{p}_{inv}$ の次元数との関係を図 9 に示す．図 9 の縦軸は式 (6) と定義する．

$$R_{FP} = \frac{winny,\ share \text{ に誤認識されたフローの数}}{winny,\ share \text{ 以外によるフローの数}} \times 100 \qquad (6)$$

図 9 の結果から，$\vec{p}$ が 6 次元以下であれば他のアプリケーションに対する誤認識を 10% 以下に抑えられることが分かる．また，ここでも $\vec{p}$ を用いた場合は誤認識が急激に増加するのに対し，$\vec{p}_{inv}$ を用いることで誤認識の増加を抑えられている．ここからコンテンツを含むパケットの影響で他のアプリケーショ

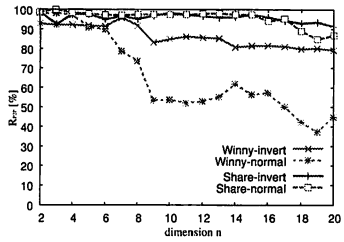

図 8　winny と share に関する弁別精度

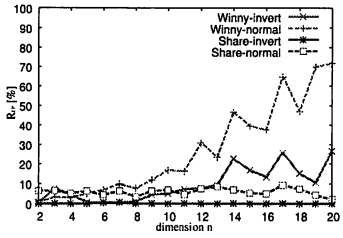

図 9　他アプリケーションに対する誤認識の割合

ンのフローを winny と誤認識してしまうことを逆数を用いることにより軽減出来たといえる．これらの点から，$\vec{p}_{inv}$ を用いることで弁別精度の次元数に対する依存性を低減でき，次元数の決定が容易になると考えられる．

4.4.2　誤認識したフローに関する考察

逆数を用いたことにより誤認識となる例も存在した．$\vec{p}$ を用いた場合には正しく弁別できた rtsp のフローの一部は，$\vec{p}_{inv}$ を用いた場合に http と誤認識された．これは rtsp が http に似せて設計されており [10]，フローの初期段階における rtsp と http の挙動が類似しているためと考えられる．一方で，これらのフローが $\vec{p}$ を用いた場合に正しく認識されたのは，コンテンツを含むパケットのペイロード長の最大値が http と rtsp で異なっていたためであると考えられる．$\vec{p}$ を用いた場合はこの最大値の差により http と rtsp が正しく弁別されたが，$\vec{p}_{inv}$ を用いるとこの違いがあまり生じないため，誤認識が生じたものと考えられる．このようなアプリケーションに対しては，$\vec{p}_{inv}$ を用いた弁別は必ずしも有効ではないといえ，他の指標との組合せが必要になると考えられる．

5.　まとめと今後の課題

本稿では，ポート番号やシグネチャによらず通信の初期段階においてアプリケーションを弁別することを目的とし，ペイロード長の遷移パタンに基づいた弁別手法を提案した．提案手法は，アプリケーションが通信開始直後に行う定型的な通信によるペイロード長の遷移パタンがアプリケーション毎に類似することを仮定し，アプリケーション毎の類似性を利用して弁別を行う．本手法はアプリケーションが通信を開始した直後の数パケットを観測するのみで弁別が可能となる点が最大の特徴である．また，実際にアプリケーションの弁別を行ううえで障害となるコンテンツの送受信の影響を軽減する方法として，ペイロード長の逆数を利用する手法を提案した．P2P アプリケーションを含むトラヒックデータを用いたアプリケーション弁別実験においては，次元数を 6 としたときに 90%以上の弁別精度を示した．さらに，ペイロードの逆数を用いることで，次元数の増加による弁別精度の低下を軽減できることを確認した．ここから，通信開始直後における数パケットのペイロード長の遷移パタンを用いることにより高精度のアプリケーション弁別が可能であると分かる．

しかし，今回提案した手法はあらかじめ弁別の対象となるアプリケーションのフローから参照ベクトルを作成する必要があるため，参照ベクトルが存在しないアプリケーションを弁別することは不可能である．未知のアプリケーションへの対策としては，参照ベクトルと観測した通信から取得したベクトルのベクトル間距離が一定以上である場合に未知アプリケーションと判断する方法が考えられるが，そのためには未知と判断するための閾値を適切に設定することが重要となるため，クラスタリング結果をより詳細に分析する必要がある．

文　献

[1] NPO 日本ネットワークセキュリティ協会：“2005 年度情報セキュリティインシデントに関する調査報告書 - 情報漏洩による被害想定と考察” (2006).

[2] 内閣官房情報セキュリティセンター：“Winny を介して感染するコンピュータウイルスによる情報流出対策について”. URL: <"http://www.nisc.go.jp/press/inf_msrk.html">.

[3] JPCERT, JapanComputerEmergencyRespoceTeam：“”p2p ファイル共有ソフトウェアによる情報漏えい等の脅威について””. URL: <"http://www.jpcert.or.jp/ed/2006/ed060001.txt">.

[4] IPA，情報処理推進機構：“情報セキュリティ白書 2007 年版”. URL: <"http://www.ipa.go.jp/security/vuln/20070309_ISwhitepaper.html">.

[5] S. Sen, O. Spatscheck and D. Wang: “Accurate, scalable in-network identification of p2p traffic using application signatures”, WWW '04: Proceedings of the 13th international conference on World Wide Web, pp. 512–521 (2004).

[6] 北村, 静野, 岡部：“フロー挙動分析に基づくアプリケーション識別手法”, 電気情報通信学会技術研究報告,NS2005-136 (2005).

[7] J.Erman, M.Arlitt and A.Mahaniti: “Traffic classification using clustering algorithms”, MineNet '06: Proceedings of the 2006 ACM SIGCOMM workshop on Mining network data, pp. 281–286 (2006).

[8] 北村, 静野, 岡部：“パケットタイプ遷移パタン分析を用いたトラヒック識別手法”, 電気情報通信学会技術研究報告,NS2006-27 (2006).

[9] T. Kohonen：“自己組織化マップ”, シュプリンガー フェアラーク東京 (1996).

[10] H. Schulzrinne, A. Rao and R. Lanphier: “Real time streaming protocol (rtsp)” (1998).