Vol.2017-ARC-227 No.31
2017/7/28

動画トラフィックに着目した NIDS における文字列探索処理負荷削減手法の提案

高徳真晴[†1, a)]　八巻隼人[†1]　三輪忍[†1]　本多弘樹[†1]

概要：近年，サイバー攻撃は巧妙化，悪質化するだけでなく，被害件数は増加し続けており，より高機能なセキリュティ技術の導入が期待されている．この課題の解決手法の一つとして，ネットワーク侵入検知システム（Network Intrusion Detection System: NIDS）が近年注目されている．NIDS はネットワーク経路上に設置されることから，ネットワークセグメント内の全ユーザに対してセキュアな通信を提供できるセキュリティシステムである．しかしながら，NIDS は通過するパケットのヘッダだけでなくペイロードまで解析，検査するため処理負荷が重く，スループットの獲得が課題となっている．そこで本稿では，動画トラフィックに着目し，動画トラフィックを NIDS の検査対象から外すことにより，NIDS の文字列探索処理負荷を削減する手法を提案する．提案手法では同一フローのトラフィック量がスレッショルドを超えた時に，そのフローの以降のトラフィックを動画トラフィックと判断する．シミュレーションの結果，スレッショルドを 15MB 程度に設定することでシミュレーションプログラムが動画トラフィックのみを抽出できることがわかった．

キーワード：サイバーセキュリティ，NIDS，文字列探索

1. はじめに

近年，サイバー攻撃は巧妙化，悪質化するだけでなく，被害件数は増加の一途をたどっている．また，近年の多様なインターネットサービスは，個人情報や金銭といった重要な情報を扱うことが多く，サイバー攻撃による被害はより深刻な問題となっている．このため，サイバーセキュリティの重要性はますます高まり，より高機能なセキュリティシステムの導入が期待されている．そこで，ネットワーク経路上で専用機器を用いて通信の解析を行うことで，悪意のあるトラフィックを検知するネットワーク侵入検知システム（Network Intrusion Detection System：NIDS）が注目されている．

NIDS は，外部からの不正なアクセスや攻撃及び悪意のあるプログラムのダウンロードがないか，Deep Packet Inspection（DPI）と呼ばれる技術を用いてネットワーク上のパケットの中身を精査することで検知するシステムである．NIDS はネットワーク経路上で設置されるためネットワークセグメント内の全ユーザが NIDS によるセキュリティ機能の恩恵を受けることができる．これにより，センサデバイスのような，セキュリティ対策機能を搭載するのに十分な計算能力を持たないデバイスであっても，外部からのアタックや不正アクセスを防ぐことができる．

一方で，NIDS はパケットのペイロードまで精査するため，処理負荷が高く，スループットの獲得が困難であることが指摘されている[1][2]．今後，インターネットトラフィックが急増することを考慮すると，NIDS 処理スループットの向上は重要な課題である．この課題解決に向けて，文字列探索処理の高速化手法と，文字列探索処理負荷の削減手法が提案されてきた．しかしながら，これらのアプローチによる NIDS 処理スループットの向上効果は，将来的なネットワークでの NIDS の実用に対して十分であるとは言えない．従って，よりスループットを向上可能な新たな手法を模索する必要がある．

そこで，本報告では，動画トラフィックを NIDS の検査対象から外すことで，NIDS の処理負荷を削減する手法を提案する．現在インターネットトラフィックに占める動画トラフィックの割合は大きく，シスコシステムズの報告[3]によると，全世界のインターネットトラフィックに占める IP ビデオトラフィックの割合は 2015 年から 2020 年にかけ，更に 70%から 82%に増加する見込みである．このことから，NIDS における検査対象の大部分はビデオトラフィックであることがわかる．そして，動画パケット内にウィルスなどの悪質なデータが含まれている報告がないことが明らかになっており，動画トラフィックにウィルスなどの悪質なデータが含まれていることはない．

そこで本報告では，HTTP 通信の開始時にやりとりのされる DNS（Domain Name System）に着目することでインターネットトラフィック中から動画トラフィックを識別し，識別された動画トラフィックを NIDS の検査対象から外すビデオバイパス手法を提案する．そして，実際のネットワークトラフィックを用いたシミュレーションにより NIDS 負荷削減率の評価を行う．

2. NIDS

NIDS は，外部からの不審なアクセスや不正なプログラムのダウンロードがないか，ネットワーク上を流れるトラフィックを精査することで検知するシステムである．NIDS は，DPI により構成されている[4]．DPI は，ネットワーク上におけるパケット検査技術の一つで，パケットのペイロ

†1　電気通信大学
The University of Electro-Communications
a)　takatoku@hpc.is.uec.ac.jp

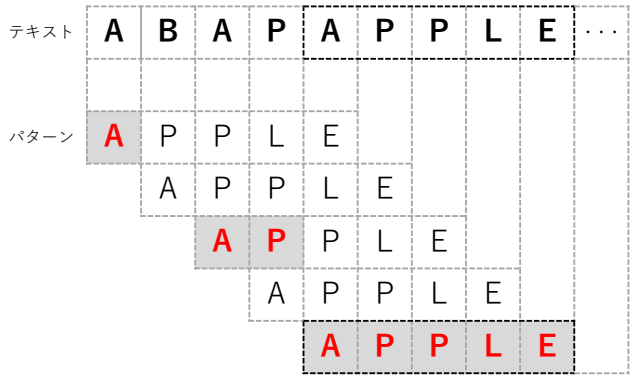

図 1．Brute-force 法における文字列探索処理の模式図

ードの解析結果を基にフィルタリングなどの処理を決定する技術である．DPI により，あらかじめ用意されたブラックリスト内の攻撃シグネチャ(パターン)とパケットペイロードに対し，文字列マッチングを行うことで，悪意ある通信を検知する．

文字列探索手法のうちより単純な単一文字列探索法について，単一文字列探索法の代表的な手法である，Brute-force 法を例として説明する[4]．図 1 に Brute-force 法の文字列探索処理の模式図を示した．Brute-force 法では，検索対象となるテキストに対し，パターンの 1 文字目との一致をテキストの先頭から 1 文字ずつ照合していき，パターンの 1 文字目がテキスト中で発見された場合，テキストとパターンそれぞれの次の文字との一致が照合され，照合が失敗するまでこの処理は繰り返される．そして，最終的にパターンの最後の文字まで一致した場合に，このテキストからパターンが検出されたこととなる．パターンの文字の照合が失敗した場合は，パターンの文字列をテキストに対して後方に一文字ずらし，再びテキストとパターンの 1 文字目を照合する．

このように，文字列探索処理はパケットのペイロードに対して一文字ずつパターンの照合を繰り返すことから，パケットサイズに比例して照合回数が増大し，NIDS におけるスループット向上のボトルネックとなることが知られている[5]．今後のインターネットトラフィック量の増大に対し，NIDS のスループットを向上させるためには，文字列探索処理を改善する必要がある．

3. 関連研究

従来の NIDS における処理スループット向上に関する研究は大きく分けて 2 つに分類される．1 つはオートマトンの改善や処理並列度の向上により文字列探索処理を高速化する手法，もう 1 つはパターンや探索文字列を圧縮，削減することで処理負荷を削減する方法である．本章では，これら 2 つのアプローチについて既存研究を紹介し，その利点と欠点を述べる．

NIDS の処理スループット向上を目的とした研究の多くは，文字列探索の高速化に焦点を当てている．Rathod らは，NIDS の文字列探索処理の高速化のために，これまでいくつかの有限オートマトンが提案されていることを述べている．オートマトンは 2 種類存在し，高速だが要求メモリ量の大きい決定性有限オートマトン（deterministic finite automata : DFA）と，低速だが要求メモリ量の小さい非決定性有限オートマトン（non-deterministic finite automata : NFA）が存在する．論文[4]では，それぞれの問題点であるメモリ量，処理速度を改善するため，SM-DFA，TM-DFA，BCAM-NFA といった，様々なオートマトンが提案されている．

その他に，GPU (Graphics Processing Unit) を使った NIDS の文字列探索処理の高速化手法がある[1][6][7][8]．Vasiliadis らは，Snort の文字列探索を GPU 上で処理できるよう Snort を改良した侵入検知システムである Gnort を開発し，合成したネットワークトレース用いたシミュレーションにより最大 2.3Gbps のトラフィック処理スループットが得られたと述べている[1]．また，イーサネットインターフェースを使用した実ネットワークトラフィックに対し，非改良の Snort と比較して 2 倍の処理性能を示した．

また，文字列探索処理負荷の削減に着目した研究もなされている．Wang らは，一つの有限オートマトンである StriFA (Stride Finite Automata) を提案し，文字列探索の際に，タグ文字と呼ばれる指定したアルファベット文字を利用することで，すべての文字列に対して文字列探索を行うことなく，有限オートマトンの速度，容量ともに改善できると述べている[9]．StriFA では，タグ文字間の文字数によりオートマトンを構成することで，オートマトンのサイズを削減し，また少ないマッチング回数によりパターン照合を可能とする．更に，タグ文字を複数用意することで，より正確に指定したパターンを検出できると述べている．しかしながら，StriFA は用意したタグ文字の種類が少ない場合，タグ文字間の文字数の同じ文字列を誤検出してしまう可能性がある．これに対し，タグ文字の種類を増やすことで誤検出を削減することは可能だが，そうした場合，StriFA の長所である高速性が損なわれてしまう課題がある．

NIDS の文字列探索処理の高速化の研究では，従来の NIDS と比較して 2 倍ほどのスループット向上を実現し，NIDS の文字列探索処理の負荷削減手法では，従来の NIDS よりも高速で低メモリ量での文字列探索処理を可能としたが，今後インターネットトラフィックが急増することを考えると，従来研究で実現できたスループットよりも大幅な NIDS のスループット向上が必要である．

4. NIDS におけるビデオバイパス手法の提案

本報告では，動画トラフィックに着目した NIDS における文字列探索処理負荷の削減手法を提案する．ここでは，動画トラフィックを動画サーバが送る全トラフィック，動画データを動画本体のデータと定義する．ネットワークトラフィックから動画トラフィックを識別するためには，以

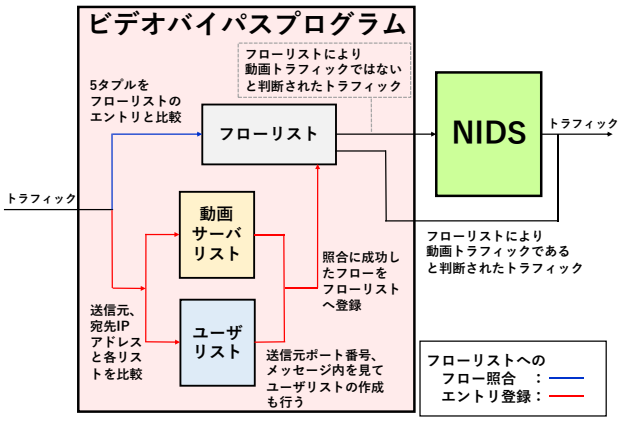

図 2. 本提案手法のアーキテクチャ図

下の 3 つのプロセスを要する.

・動画サーバの IP アドレスの特定

・動画サイトで動画を視聴するユーザの IP アドレスの検出

・これらのユーザと動画サーバ間の通信からの動画トラフィックの検出

が必要である. 本報告では, 図 2 に示したように, これら 3 つの機能をそれぞれ

・動画サーバリスト

・ユーザリスト

・フローリスト

を用いて実現し, NIDS の文字列探索対象から動画データをバイパスするビデオバイパス手法を提案する.

まず, ユーザが視聴する動画データを配信する動画サーバの IP アドレスの特定方法を検討する. 本報告では, NIDS サーバ自身が動画サーバと通信することで動画サーバの IP アドレスを取得する手法を考える. まず, NIDS サーバが動画サイトへ定期的にアクセスし, 各動画サイト内の動画を一定数ランダムに視聴する. NIDS サーバは, この時の動画視聴トレースを解析することで動画サーバの IP アドレスを取得し, 動画サーバの IP アドレスを動画サーバリストへ記録する. 動画サーバの IP アドレスは頻繁には変わらないため, 動画サーバリストの更新は一週間に一度ほどのペースで行えば十分だと考えられる. 本報告では, 動画サーバリストはあらかじめ作成されているものとし, 動画サーバリストの更新については考えない.

次に, 動画サイトから動画を視聴するユーザを検出する手法を検討する. ここで, 我々は DNS (Domain Name System) に注目した. DNS は, インターネット上で Web ページの URL などに含まれるドメイン名と, 実際のやり取りに用いられる IP アドレスを変換するためのシステムである. DNS における通信は, まず, ユーザがドメインを記載した DNS リクエストパケットを, IP アドレスとドメインの変換表を持つ DNS サーバに対して送り, DNS サーバが回答として, ドメインに対する IP アドレスをドメインと共に記載した

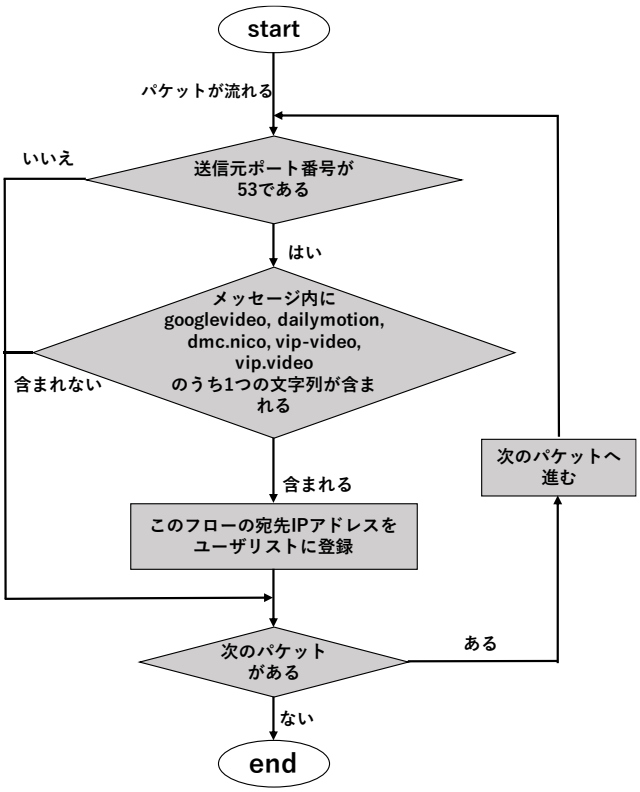

図 3. ユーザリストの作成のフローチャート

DNS レスポンスパケットをユーザへ変身することで完了する. ユーザは DNS レスポンスパケットメッセージに記載された IP アドレスへアクセスすることで, Web ページへアクセスすることができる. このことから, メッセージ内に動画サーバのドメインを含んだ DNS レスポンスパケットを検出することで, 直後に動画を視聴しようとするユーザが特定できる. 図 3 に動画視聴ユーザの検出フローを示した. 一般的に, DNS の通信はポート番号 53 を用いることから, パケットの送信元ポート番号が 53 のパケットのメッセージ部に対し, 動画サーバのドメインを文字列探索する. 動画サーバのドメインはそれぞれ,

・Youtube[10]…"googlevideo"

・Dailymotion[11]…"dailymotion"

・ニコニコ動画[12]…"dmc.nico"

・FC2[13]…"vip.video"または"vip-video"

であった. ここで, 動画サーバのドメインを検知した場合には, 当該パケットの宛先 IP アドレスを動画視聴ユーザと判別し, ユーザリストへと IP アドレスを登録する.

最後に, ユーザと動画サーバ間の通信から動画データを識別する方法を検討する. これまで述べた手法で得られたユーザリストと動画サーバリストを参照することで, 動画サーバから動画視聴ユーザに向けた通信を検出することは可能だが, 動画サーバからの通信には動画データ以外のデータも含まれる可能性がある. そこで, より細かい通信粒度としてポート番号まで含めた 5 タプル (送信元 IP アドレス, 宛先 IP アドレス, 送信元ポート番号, 宛先ポート番号,

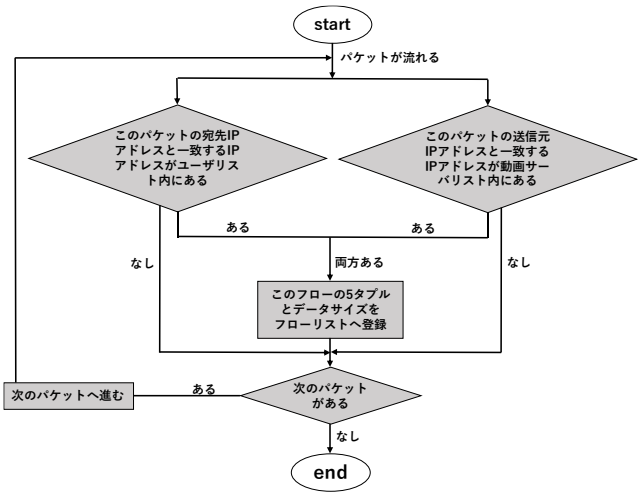

図 4．フローリストの作成のフローチャート

プロトコル番号）により定義されるフロー単位で，ユーザと動画サーバ間の通信を管理するフローリストを新たに作成する．

フローリストの作成フローを図 5 に示した．動画サーバからユーザに向けた通信フローを抽出するため，動画サーバリストに登録された IP アドレスを送信元に持ち，ユーザリストに登録された IP アドレスを宛先に持つパケットを検出し，そのフロー情報とそのフローで転送された総データ量をフローリストに登録する．一般的に動画データは HTML や画像といった他のデータと比較してデータサイズが桁違いに大きいことから，フローリストに登録されたフローの中で転送された総データ量がある閾値を超えたフローを動画データの転送フローと判断し，当該フローの以降のパケットを NIDS 検査対象外とする．本報告では，動画データの通信が終了したフローに対しては，フローリストから理想的に削除するものとして考える．

5. ビデオバイパス手法の評価

本章では，提案手法の有効性を評価するため，ソフトウェア実装したビデオバイパス手法のシミュレータを用いてバイパス率を測定し評価する．シミュレーションには，含有する動画サイズの分布を変化させた 3 つのトラフィックを使用した．以下では，まずワークロードとなる 3 つのトラフィックの作成方法について述べ，次にビデオバイパスプログラムの概要について述べる．

評価トラフィックは，動画トラフィックの含まれないベーストラフィックと，動画トラフィックの含まれるトラフィックをあらかじめ準備し，これらのトラフィックの動画含有率が前述したように 7 割程度となるよう適切に複製およびマージすることで作成した．ベーストラフィックは，研究室内において 15 分程度，動画サイト以外の Web ページにランダムにアクセスしたトラフィックを 1 ユーザのトラフィックとして複数キャプチャし，20 トラフィック分をマージすることで 20 ユーザが存在するネットワーク内を想定したトラフィックを作成した．ここで，トラフィックのキャプチャには Wireshark(ver2．2．2)[14]を，トラフィックのマージには，tcpreplay を用いた．なお，トラフィックをマージする際に，各トラフィックのユーザ IP アドレスが重複しないよう，tcpreplay により IP アドレス書き換えてマージした．動画トラフィックを含んだトラフィックは，無作為に選んだ動画サイトの動画のページリンクをクリックする直前にキャプチャを開始し，動画データ転送が終わる十分な時間の後にキャプチャを終了した．ここで，動画データサイズの計測のために，Youtube では Clipconverter[15]，Dailymotion，ニコニコ動画，FC2 動画の 3 サイトでは動画ゲッター[16]を使用し，動画のデータサイズのみを計測した．取得した動画トラフィックは動画データの範囲が 10~160MB 程度の動画トラフィックを，Youtube，Dailymotion，ニコニコ動画，FC2 動画の 4 サイトでそれぞれ複数本キャプチャした．

フローリストの作成方法についてフローチャートを使って説明する．図 4 にはフローリストのフローチャートを示した．以下にフローリストのフローチャートの説明をしていく．まずは，ビデオバイパスプログラムへパケットが入力される．まずこの入力パケットの宛先 IP アドレスがユーザリスト内にある IP アドレスと一致するかどうかと，送信元 IP アドレスが動画サーバリスト内にある IP アドレスと一致するかどうかを検査する．ここで，一方もしくは両方で IP アドレスの一致が確認できなかった場合次のパケットがあるかの条件分岐へ進み，両方のリストでの IP アドレスの一致が確認できた場合，次の条件分岐へ進み，入力パケットの 5 タプルの情報とデータサイズをフローリストへ登録し，次のパケットがあるかどうかの条件分岐へ進む．次のパケットがあるかどうかの条件分岐では，次のパケットがあった場合は最初に戻り，パケットの宛先 IP アドレスとユーザリストの照合の処理をする．もし次のパケットがなかった場合，ビデオバイパスプログラムの処理を終了する．以上のようにフローリストは作成される．

本報告では，ビデオバイパスプログラムと上述した評価用トラフィックを用いて，本来の動画含有率と，ビデオバイパスプログラムにより検出された動画率を比較評価する．2 章で述べたように，NIDS における文字列探索処理はパケットサイズに比例してパターン照合回数が増えることから，ビデオバイパスプログラムによる動画検出率と NIDS の負荷削減率はほぼ等しいと考えられる．また，フローリストにおいて動画データ転送フローを判別するために要するデータ転送量の閾値についても評価を行い，最適なスレッショルドを検討する．

図 5 は，作成した 3 種類の評価トラフィックが内包する動画データサイズの累積分布を示したグラフである．図 5 の横軸は評価トラフィック内マージされた動画トラフィックそれぞれの動画データサイズを表し，縦軸はそれぞれの

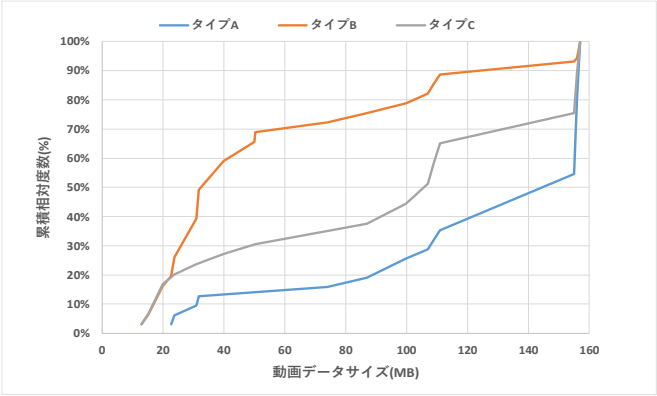

図 5. 各評価トラフィックにおける動画データの累積分布

動画データサイズの値以下の動画トラフィックが含まれる累積割合を示している．今回作成した評価トラフィックを以降では，それぞれタイプ A，B，C の評価トラフィックと呼ぶ．タイプ A の評価トラフィックは，動画データサイズが 100MB を超える，データサイズの大きい動画トラフィックが全体の半数以上を占めるように作成した．タイプ B の評価トラフィックは，動画データサイズが 50MB 以下の動画トラフィックが全体の半数以上を占めるように作成した．タイプ C の評価トラフィックは，全動画トラフィックの本数が均一になるように作成した．これらの評価トラフィックに含まれる動画の本数は，タイプ A の評価トラフィックが 31 本，タイプ B の評価トラフィックが 94 本，タイプ C の評価トラフィックが 30 本である．

次に，バイパス率について説明する．バイパス率 B は，ビデオバイパスプログラムへ入力される全トラフィックを T_{all}，ビデオバイパスプログラムにより NIDS の検査対象から外されるトラフィックのデータ量をT_{out}として，以下の式で定義する．

$$B[\%] = \frac{T_{out}[\text{byte}]}{T_{all}[\text{byte}]}$$

上式より，ビデオバイパスプログラムが理想的に動画データのみを抽出できた場合，バイパス率は動画含有率と等しくなる．本報告ではフローリストにおける動画データ転送フロー判別のためのスレッショルド値を 10KB，100KB，1MB，10MB，100MB とし，バイパス率の差を評価する．

タイプ A，B，C の評価トラフィックにおいて異なるスレッショルド値を設定した場合のトラフィック削減率を図 6 から図 8 に示した．スレッショルドとして小さいデータサイズを設定した場合，トラフィック削減率が動画含有率を超えることが確認されたが，これは動画データ以外のフロー（画像や HTML データ等）を動画データであると誤検知したためだと考えられる．また，スレッショルドとして 10MB より大きいデータサイズを設定した場合は，バイパス率が動画含有率を下回った．これは動画データであっても検知するまでに 10MB 以上のデータを要すことから，検知可能な動画データサイズが減ってしまう，またはデータサイズの小さい動画を検知できていないためである．

次に，評価トラフィックの違いによる誤検知率及び非検知率の差を評価する．誤検知率は検知したデータにおいて動画データ以外のデータを検知した割合を示し，非検知率は動画データ全体に対し検知できなかった動画データの割合を示す．誤検知率 F は，動画含有率を V として，以下の式により算出する．

$$F[\%] = \frac{B[\%] - V[\%]}{B[\%]}$$

同様に，非検知率 N は以下の式により算出する．

$$N[\%] = \frac{V[\%] - B[\%]}{V[\%]}$$

上式を用いて，各評価用トラフィックにおける誤検知率および非検知率を算出した結果を表 1 に示した．

表 1．各評価トラフィックのスレッショルドによる誤検知率 F 及び非検知率 N

		各評価トラフィックの F または N[%]		
		タイプ A	タイプ B	タイプ C
スレッショルド	10	11.89(F)	16.37(F)	16.86(F)
	100	11.79(F)	16.25(F)	16.71(F)
	1,000	10.78(F)	15.30(F)	15.31(F)
	10,000	3.39(F)	8.76(F)	6.18(F)
	100,000	34.44(N)	43.39(N)	54.24(N)

表 1 から，データサイズの大きい動画データの比率が高いタイプ A の評価トラフィックの誤検知率は，他の 2 種類の評価トラフィックにおける低い割合となることが分った．これは，動画データのデータサイズが大きいことから，画像データや HTML データのデータサイズが相対的に小さくなるためだと考えられる．従って，誤検知率はトラフィックに含まれる動画数ではなく，トラフィックに含まれる動画データサイズの組成により異なると考えられる．また，非検知率に関しては，タイプ A と比較して他の 2 つの評価トラフィックにおける非検知率が高くなった．これは，タイプ A と比較して，他の 2 つの評価トラフィックには 100MB 以下の動画データが多く含まれることから，このような動画データの大部分を検知できていないためだと考えられる．

最後に，以上の結果からスレッショルドの最適な値を求

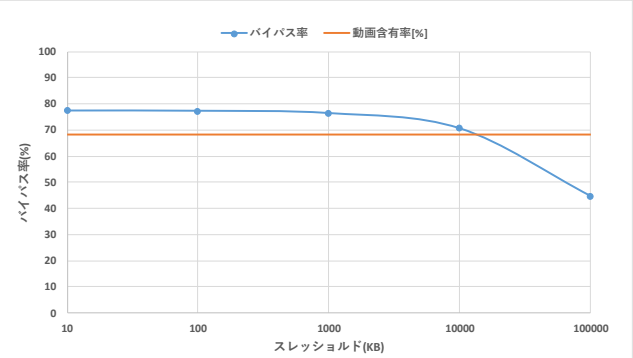

図 6.　タイプ A の評価トラフィックにおける
バイパス率とスレッショルド値の相関

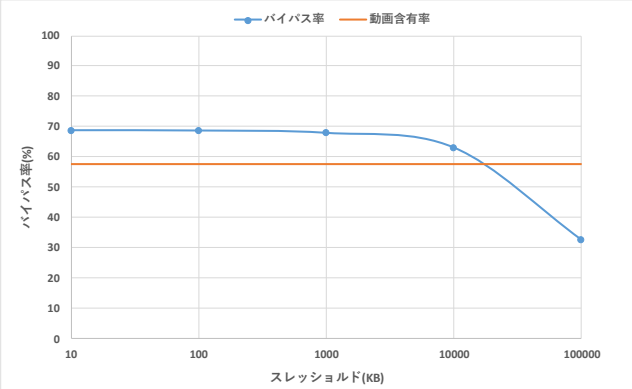

図 7.　タイプ B の評価トラフィックにおける
バイパス率とスレッショルド値の相関

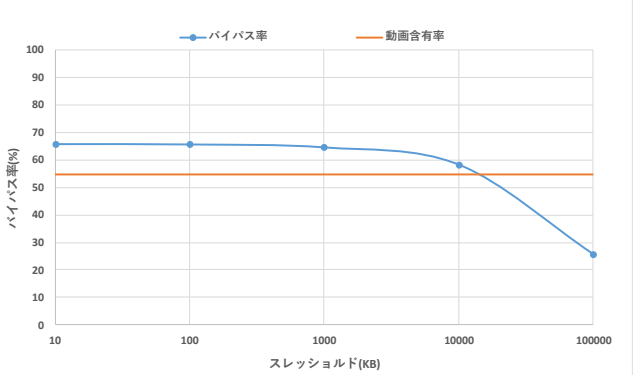

図 8.　タイプ C の評価トラフィックにおける
バイパス率とスレッショルド値の相関

める．図 6 から図 8 を見ると，誤検知率はスレッショルドを 10KB から 1MB に設定した場合は大きく変化せず，スレッショルドを 10MB 以上に設定した場合に低下していく．一方で，スレッショルドを高データサイズとするほど非検知率が増加していくことから，動画検知精度とトラフィック削減性能の間にはトレードオフの関係があることがわかる．以上を踏まえると，ビデオバイパスプログラム手法における動画データ識別のためのスレッショルドは，誤検知率および非検知率が 0 となる 15MB 程度が適切であると考えられる．

6.　おわりに

今後ネットワーク技術の進化と共にサイバー攻撃は巧妙化，悪質化するだけでなく，被害件数は増加の一途をたどり，より高機能なサイバーセキュリティの重要性がますます高まってきている．上記の課題解決に期待されるセキュリティシステムの一つとして，NIDS が注目されている．

NIDS はネットワーク経路上に設置されることから，ネットワークセグメント内の全ユーザに対して NIDS によるセキュアな通信を提供できる.しかし，NIDS は通信の中身まで解析するため処理負荷が重く，スループットの獲得が困難である.この課題に対し, GPU による文字列探索処理の高速化や文字列探索手法の改善に関する研究など, NIDS のスループット向上のための研究が数多くなされてきたが，今後ネットワークトラフィックが急増していくことを考えると，より大幅な NIDS のスループット向上が必須であると考えられる.

そこで，動画トラフィックを NIDS の検査対象から外すことで，NIDS の文字列探索処理負荷の削減を目指す.本報告では，その初期検討として，動画トラフィックに着目した NIDS の文字列探索処理負荷の削減手法を提案し，動画トラフィックと他のトラフィックの判別をするスレッショルドの適正値をシミュレーションプログラムによって評価した.シミュレーションの結果，スレッショルドを 15MB 程度に設定することでビデオバイパスプログラムが動画トラフィックのみを抽出することができることがわかった.

また，適切なスレッショルドを設定しなかった場合，動画ではないトラフィックを動画トラフィックと誤検知，または動画トラフィックの検出漏れを引き起こすことがわかった.これにより，本提案手法に加えて，トラフィックのペイロード解析を行うことによりこれまで以上に正確な動画の抽出を可能とし，より大きな NIDS の文字列探索処理負荷の削減を望めることが分かった.

参考文献

[1] Vasiliadis G.,Antonatos S.,Polychronakis M.,Markatos E.P.,Ioannidis S.,“Gnort: High Performance Network Intrusion Detection Using Graphics Processors”,RAID 2008.Lecture Notes in Computer Science,vol 5230.Springer,Berlin,　Heidelberg

[2] Mukherjee, Biswanath, L. Todd Heberlein, and Karl N. Levitt. "Network intrusion detection." *IEEE network* 8.3 (1994): 26-41.

[3] The Zettabyte Era — Trends and Analysis, Cisco http://www.cisco.com/c/ja_jp/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

[4] Prashantkumar M. Rathod; Nilesh Marathe; Amarsinh V. Vidhate,“A survey on Finite Automata Based Pattern Matching Techniques for Network Intrusion Detection System (NIDS)”, Computers and Communications (ICAECC), 2014 International Conference on Advances in Electronics

[5] Yi-Ching Liao, “A Survey of Software-Based String Matching Algorithms for Forensic Analysis”, 2015 ADFSL 77-86, 2015 CDFSL Proceeding

[6] Desai, Neil. "Increasing performance in high speed NIDS." *A look at Snort's Internals* (2002).
[7] Ambekari T; A.N.Mulla, "Network Instusion detection on CPU GOU System", International Jounal of Sientific and Research Publications Volume4 Issue 12 December 2014, ISSN 2250-3153
[8] Huang, Nen-Fu, et al. "A gpu-based multiple-pattern matching algorithm for network intrusion detection systems." *Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on*. IEEE, 2008.
[9] Xiaofei Wang and Yang Xu, "StriFA: Stride Finite Automata for High-Speed Regular Expression Matching in Network Intrusion Detection Systems", in IEEE SYSTEMS JOURNAL,VOL.7, NO.3, SEPTEMBER 2013.
[10] Youtube 公式サイト https://www.youtube.com/?hl=ja&gl=JP
[11] Dailymotion 公式サイト http://www.dailymotion.com/jp
[12] ニコニコ動画公式サイト http://www.nicovideo.jp/?wapr=5940e2fa
[13] FC2 動画公式サイト http://video.fc2.com/ja/
[14] Wireshark https://www.wireshark.org/
[15] ClipConverter http://www.clipconverter.cc/
[16] 動画ゲッター http://www.douga-getter.com/index.html