サービス指向型ルータのためのパケット単位の文字列探索

原島　真悟†　　石田　慎一‡　　西　宏章†

慶應義塾大学理工学部システムデザイン工学科†

慶應義塾大学大学院理工学研究科‡

1. はじめに

現在，インターネット上の情報検索を含むあらゆるサービスは，エンドホストにより提供されている．例えば、検索エンジンはクローラによりインターネット上の情報を収集するに留まり，エンドホストから得られる情報に頼って検索結果の表示順位を決定している．しかし，利用者があるサイトに訪れた後どのサイトに訪れたか等の行動履歴やサイトの滞在時間に基づく検索順位情報の決定などは，順位決定に必要な情報が欠如するため困難である．一方，トラフィック情報を把握するという点では，バックボーン等に存在するルータは優れた位置に存在する．これは，ルータがパケット中継の中枢であるためである．

我々は，ルータが単なるパケット転送に留まらず，積極的にトラフィック情報を収集，解析するサービス指向型ルータ(SOR)を提案している[1][2]．またサービス指向型ルータにおける動作検証と有効性を調べ，上位アプリケーション開発環境を提供するために Linux で動作するシミュレータソフトウェア SRIM (Service oriented Router sIMulator) を開発している．

SOR の情報収集では，高速に流れる混ざり合った複数の TCP コネクションから必要なコネクションの必要な文字列を抽出することが求められる．本稿では，SRIM における文字列を探索する String Extract Filter 提案し，評価する．

2. SRIMにおけるペイロード解析

TCP を用いた通信のペイロード解析には，分割されたパケットを束ねることで TCP ストリームを再構築する必要がある．特に SRIM の正規表現解析は perl 互換の正規表現ライブラリ boost_regex を用いて実装しており，解析にはストリーム再構築後のデータが必要となる．

この解析においては，すべてのコネクションを一律に再構築し，正規表現処理する必要はない．例えば，先頭に位置する数パケットのペイロード情報で抽出する必要があるかどうかがわかる場合は，ペイロードの保持や再構築，解析するのは計算資源の無駄である．従って，再構築の完了を待つのではなく，到着したパケットのコネクション情報をもとに当該ストリームの最新の処理状態を呼び出し，到着パケットの処理が終わった時点で再びその処理結果を保存する，いわゆるコンテキストスイッチ可能な文字列探索方式が必要である．

SRIM は libpcap により eth デバイスを通過するパケットを取得し，解析を行う．SRIM ではパケットが到着すると TCP Reconstructer によりパケットが属するストリームを調べ，パケットにストリーム情報を付加，パケットのポインタを Stream Reconstruct Information に登録する．その後，IP とポート情報を確認し，HTTP/1.1 のデコードを行う．また，String Extract Filter において文字列探索を行い，正規表現処理を行う．そして，抽出情報を PostgreSQL Save Engine によりデータベースに蓄える．この流れを図 1 に示す．

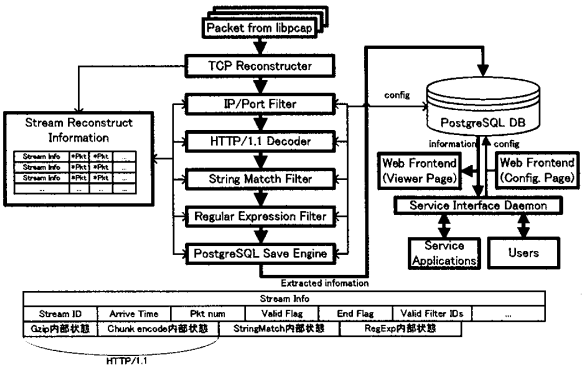

図 1 SRIM の処理の流れ

3. String Extract Filter

String Extract Filter は後に続く Regular Expression Filter の負荷を軽減するためのフィルタである．コンテキストスイッチを行いながらパケットの到着毎に検索対象文字列がペイロードに含まれるかを調べることで，検索対象に該当しないことが確実になった時点で正規表現処理を省略する．また，"abc[d-z]+123(¥d*)def" などのように文字列探索のみでは処理できない場合も中に含まれる固定文字列を探索すること

Packet Base Pattern Matching for Service Oriented Router

† Shingo Harashima, Hiroaki Nishi
Department of System Design Engineering, Faculty of Science and Technology, Keio University
‡ Shin-ichi Ishida
Graduate School of Science and Technology, Keio University

で正規表現処理の軽減をはかる．例えば，各ページのタイトルを収集することを想定し，ソースポートが 80 の 519 個のコネクション(全データサイズ 59,975,885 バイト)を用いて解析したところ，<title>，</title>の 2 つのタグは 519 個のコネクションのうち 484 個で検出され，検出されたタグの位置はデータサイズに関わらず先頭 650 から 900 バイトの間に分布していた．この場合，全体の 1.5%のデータを探索することでタイトルの位置を特定することができるため，検索コストを削減することが可能となる．

また，さらに検索コストを削減するため，文字列検索を行う String Extract Filter に利用する，新しい文字列検索アルゴリズム，Sunday-Harashima 法(SH 法)を提案する．

図 2 のように英文によるアルファベット各文字の出現頻度には偏りがある[3]．SH 法は Sunday 法を拡張し，単純にパターン末尾から照合せず，パターン内の存在確率の低い文字から照合を行うアルゴリズムである．存在確率はトラフィックの文字をカウントし定期的に再計算する．

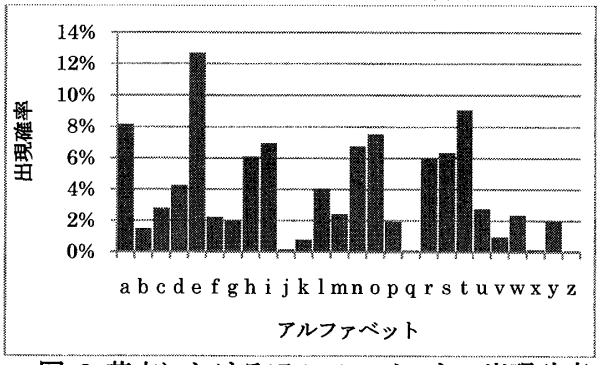

図 2 英文におけるアルファベットの出現確率

4. SH 法の評価

WIKIPEDIA の英語サイトにアクセスしたトレースを用いて評価を行った[4]．評価には Knuth-Morris-Pratt 法，Boyer-Moore 法，Horspool 法，Sunday 法，および SH 法の 5 種類の文字列探索アルゴリズムを用いた．2009 年 12 月 27 日，本研究室のゲートウェイにおいて取得したトラフィックデータを利用し，ソースポート，ディスティネーションポートのうちいずれかが 80 である 1038 個のコネクション(全データサイズ 60,094,034 バイト)を用いて実験を行った．

検索対象文字列として snortrules-snapshot-2.8 のうち web クライアントに関わる 66 個の正規表現を抜き出し，正規表現の固定文字列をパケットの到着毎に探索した[5]．表 1 に結果を示す．String Extract Filter を使わない場合，各コネクションに 66 個の正規表現をかける必要があるが，このフィルタにより平均 3.72 個まで減らすことができた．またバイトに換算すると処理対象は全データサイズの 66 倍の 3,966,206,244 バイトから 11.38%の 451,412,777 バイトまで削減できることができた．

表 1 正規表現削減量

	ルール数	対象サイズ(バイト)
使用前	66	3,966,206,244
使用後	3.72	451,412,777
割合	5.64%	11.38%

4.1. アルゴリズムの比較

各アルゴリズムの 1 文字照合回数，メモリ参照回数を調べ，上記の 66 ルールの平均を算出したものを図 3 に示す．ただし Boyer-Moore 法は 2 つのテーブルを参照するため参照回数を 2 倍にしている．SH 法のメモリ参照回数は Sunday 法と同等であるが，照合回数は Sunday 法より 2.65%減少している．

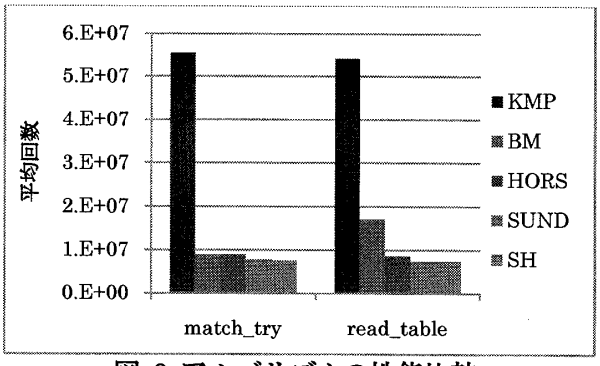

図 3 アルゴリズムの性能比較

5. 結論と今後の課題

SRIM において，コンテキストスイッチを用いて文字列探索を行うことによって正規表現処理をスキップ，または正規表現対象を 12%程度まで削減できることがわかった．また各アルゴリズムでは SH 法が優れていることがわかった．今後の課題として SH 法のハードウェア実装が挙げられる．

参 考 文 献

[1] Koichi Inoue, Dai Akashi, Michihiro Koibuchi, Hideyuki Kawashima, and "Semantic router using data stream to Hiroaki Nishi enrich services", 3rd International Conference on Future Internet CFI 2008 Seoul, Korea, pp. 20-23, June, 2008.
[2] 永冨泰次，石田慎一，三野峻徳，川島英之，鯉渕道紘，西宏章：リッチなユーザサービスを提供するセマンティックルータにおける正規表現プロセッサの提案，電子情報通信学会，ネットワークシステム研究会(NS) (2008).
[3] Lewand, Robert Edward, Cryptological mathematics 36pp. November, 2000
[4] http://en.wikipedia.org/
[5] http://www.snort.org/