計測自動制御学会東北支部 第 316 回研究集会 (2018.6.22)
資料番号 316-1

Zabbix を利用した異常トラフィックパターンの高速検出法

Fast Detection Method of Anomaly Traffic Patterns using Zabbix

○大坂侑平，佐藤健，細川靖

○ Yuhei Osaka, Ken Sato, Yasushi Hosokawa

八戸工業高等専門学校

National Institute of Technology, Hachinohe College

キーワード： ネットワーク監視 (Network Monitoring)，異常トラフィック (Anomaly Traffic)，並列処理 (Parallel Processing)，Zabbix

連絡先： 〒 039-1192 青森県八戸市大字田面木字上野平 16-1 八戸工業高等専門学校 産業システム工学専攻 電気情報システム工学コース

大坂侑平，E-mail: h29ae02@hachinohe.kosen-ac.jp

1. はじめに

ネットワークの運用において異常を迅速に発見するために監視は重要である．現在ネットワーク監視ではネットワーク機器からトラフィックデータを取得し，それをグラフ化する監視ツールが広く利用されている．しかし不具合の発見や迅速な解決は管理者の経験と勘に頼る場合が多い．またネットワークの規模が大きくなると異常の特定も複雑化し管理者の負担も大きくなる．

近年，リアルタイムにトラフィックの可視化や通知，データの蓄積機能を有する Zabbix が注目されている [1)]．Zabbix は API を備えており PHP により蓄積したデータを柔軟に再利用することが可能である．そこで Zabbix で取得したトラフィックデータをもとに異常トラフィックの発生源を自動的に発見するシステムの作成を行っている．しかし，これまでのシステムではデータ量が多く計算時間が長くなってしまったため，本研究ではネットワークトポロジーに基づいた探索と並列処理による計算時間短縮の検討を行った．

2. 実験方法

実験環境は八戸高専のネットワークとした．対象機器数は 64 個で対象ポート数は約 4000 個である．Fig.1 は本校のネットワークトポロジーを簡略化したもので，一般的なネットワーク構成となっている．Zabbix サーバの環境は以下のとおりである．

- Zabbix:Zabbix Appliance 3.4.4
  - OS:Ubuntu 16.04
  - Web サーバ:Apache 2.4.18
  - データベース:MySQL 5.7.22

先行実験として Zabbix API を利用して Zabbix で取得したトラフィックデータから異常トラフィックの発生源を自動検出するシステムを作成

した (Fig.2)．異常トラフィックの発生源を特定する処理の流れを説明する．Zabbix のフロントエンドで，ある監視対象機器に異常トラフィックを確認した場合，

1) 異常トラフィック発生源自動検出システムの Web ページ上で Zabbix に設定している監視対象機器グループ，そのグループに属する監視対象機器を取得して異常トラフィックが確認されたインターフェイスを選択する．

2) 該当インターフェイスがもつ監視項目から生成したグラフデータを取得し，異常トラフィックのグラフを選択する．

3) 異常トラフィックが確認された日時を選択する．

4) Web ページから Zabbix API を利用して Zabbix のデータベースにアクセスして条件に該当するトラフィックデータを取得する．

5) 他の監視対象機器の監視項目のトラフィックデータを取得し，異常トラフィックパターンとパターンマッチングする．

6) 検出された類似パターンをグラフとして Web 上に表示する．

Fig.2 は以上の処理の流れを図に示したものである．

類似パターンの検出方法は相関係数を用いた．相関係数は比較するそれぞれ n 個のデータをもつ X，Y に対して以下の式で与えられる．

$$r = \frac{\frac{1}{n}\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i-\bar{X})^2}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i-\bar{Y})^2}} = \frac{s_{XY}}{s_X s_Y}$$

$\bar{X}$ と $\bar{Y}$ はそれぞれ平均，s_{XY} は共分散，s_X と s_Y はそれぞれ標準偏差である．相関係数は $-1 \leqq r \leqq 1$ の値を取り，$|r|$ が 1 に近いほど相関が高い．相関係数の算出結果から高い相関を示すトラフィックパターンを検出し該当通信の経路上のインターフェイスを推定する．類似条件は相関係数 $r \geqq 0.8$ とした．

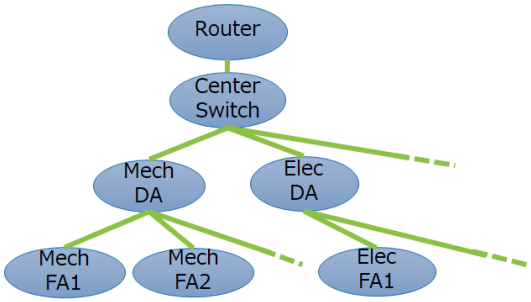

Fig. 1　実験環境のネットワークトポロジー概略図

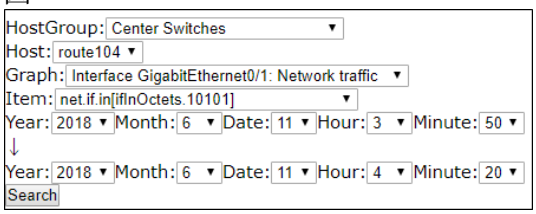

Fig. 2　異常トラフィック発生源自動検出システムの Web ページ

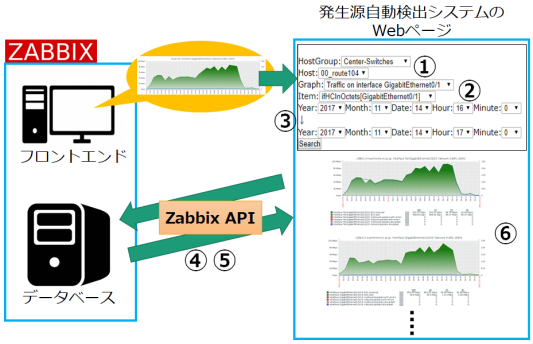

Fig. 3　異常トラフィック発生源検出の処理の流れ

2.1　ネットワークトポロジーのツリー構造

検索効率を向上させるため，ネットワークトポロジーをもとにツリー構造を構成し機器間の接続形態を考慮した探索を行う [2]．ツリー構造クラスでツリーを構成し，Zabbix API は監視対象機器の ID を指定することで該当インターフェイスからデータを取得するため各ノードに

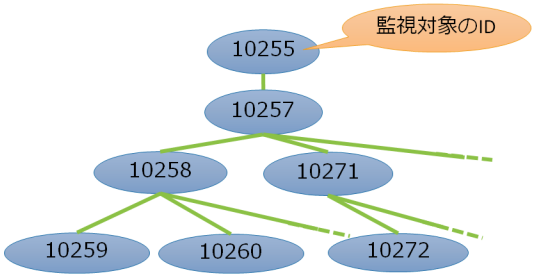

Fig. 4　ネットワークトポロジーのツリー構造

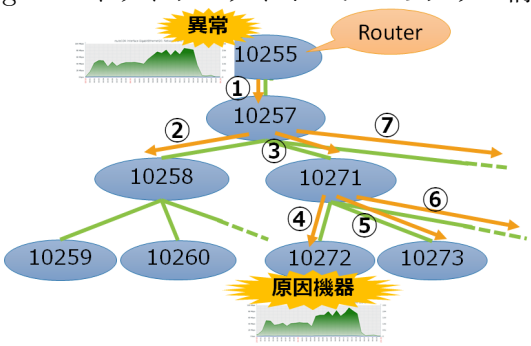

Fig. 5　探索の流れ

はZabbixが監視対象機器に割り振っているID情報を設定した(Fig.4)．Fig.5に探索の流れを示す．最初に異常トラフィックが確認されたインターフェイスの子ノードから探索を行う．もし，子ノードに異常トラフィックパターンに類似したパターンが検出された場合はさらにそのノードの子ノードに対して再帰的に探索を行う．子ノードに類似パターンが検出されなかった場合や子ノードがない場合は兄弟ノードに対して探索を行う．これにより，探索する必要のない機器を省くことができ，計算時間を短縮することができる．

2.2　並列処理

処理するデータが大量にある場合，一つのデータの処理が終わり次第次のデータを処理するというように順に処理すると計算時間が長くなる．探索の処理は並列化できるため並列処理を導入してさらに計算時間の短縮を試みた(Fig.6)．PHPで並列処理を実現するためにcURL関数を使用した[3]．cURL関数はHTTPリクエストにより外部サイトの情報を取得することができる関数である．並列処理では複数のサブプロセスに処理データを分割して渡し，各サブプロセスは受け取ったデータを同時に処理する．cURL関数を利用して各サブプロセスから結果を取得し，メインプロセスで統合する．

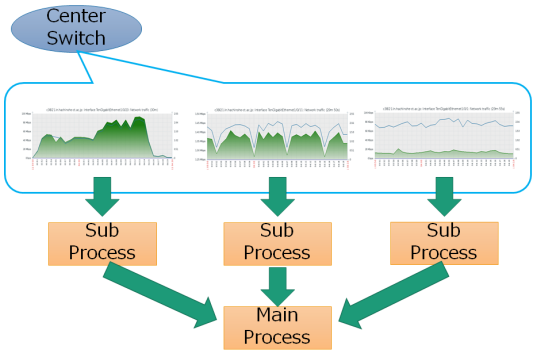

Fig. 6　並列処理の流れ

3.　実験結果

Fig.7，Fig.8に異常トラフィックパターンと検出された類似パターンを示す．結果からネットワークトポロジーに基づくツリー構造を用いた探索と並列処理により従来と同じように類似パターンを検出し，該当通信の経路上のインターフェイスを推定することができた．Fig.9に機器間の接続形態を考慮した探索をしない場合とした場合の計算時間を示す．機器間の接続形態を考慮した探索をしない場合の計算時間は260[s]であり，機器間の接続形態を考慮した探索をした場合は130[s]であった．結果からネットワークトポロジーをもとにツリー構造を構成することにより計算時間を50[%]改善することができた．Fig.10に並列数と計算時間の関係を示す．並列化しない場合の計算時間は140[s]であり，並列数2の場合の計算時間は72[s]であった．並列数を4，8，16と増やした場合の計算時間は並列数が2の場合とほぼ変わらないという結果だった．結果から並列化することにより，並列化しない場合よりも計算時間を49[%]改善することができた．並列数を増やしても計算時間がほぼ変わ

らないという結果については原因としてデータベースアクセスの処理時間が大きいことが考えられる．

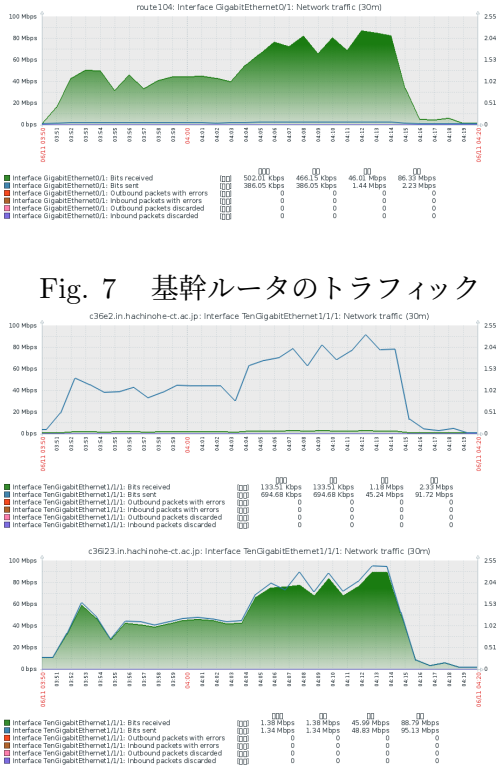

Fig. 7　基幹ルータのトラフィック

Fig. 8　検出された類似トラフィックの例

4.　おわりに

本稿では，ネットワークトポロジーに基づいた探索と並列処理により計算時間の短縮を図った．その結果，従来通りに異常トラフィックの発生源を特定することができ，また計算時間も改善することができた．しかし，実際に運用するにあたって計算時間のさらなる改善が必要であると考える．

今度の展望として，データベースアクセスの時間短縮のためにデータベースのチューニングや管理者への通知機能，処理が終わったものから順次 Web 上に表示するといった手法を検討中である．また、異常トラフィックパターンをモデル化して常にそのパターンとマッチングさせて自動化することでリアルタイムに拘らない手法も検討中である．

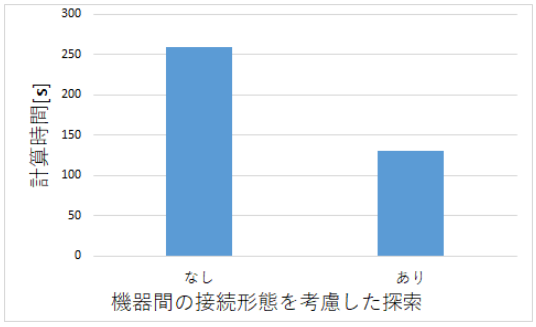

Fig. 9　機器間の接続形態を考慮した探索をしない場合とした場合の計算時間

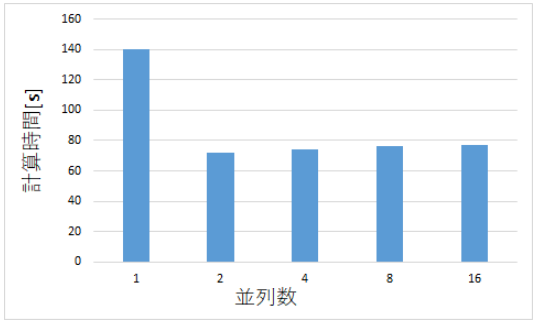

Fig. 10　並列数と計算時間

参考文献

1) Zabbix Documentation 3.0, https://www.zabbix.com/documentation/3.0/manual/api (2017.6.20)

2) 汎用ツリー構造クラスを作ってみた　，https://qiita.com/chiyoyo/items/f06e4d3f3e39d4238bca (2017.6.20)

3) PHP でマルチスレッド (バックグラウンド処理) を実現する方法，http://techblog.ecstudio.jp/tech-tips/php-multi.html (2018.4.25)

4) PHP: 関数リファレンス - Manual, http://php.net/manual/ja/funcref.php (2018.6.20)