イベントネットワークにおける syslogを用いた異常検知手法の提案と実データを用いた評価

阿部 博[1,2,a)]　敷田 幹文[3,b)]

概要：大規模なイベントネットワークではネットワーク管理手法の一つとして syslog を用いた運用監視が行われる．syslog メッセージに含まれるキーワード検知や閾値による異常検知などネットワークの異常が運用者に通知される．マルチベンダ機器によって構築される特殊なイベントネットワークでは，ログの意味解析やキーワードによる異常検知が行えない環境下であることが多い．本論文ではイベントネットワークで収集される syslog の総量による分析を行い異常を検知する手法を提案する．株式取引で用いられるボリンジャーバンドアルゴリズムを利用し，Interop Tokyo で構築される ShowNet で収集された syslog の実データを用いて統計学的手法において軽量な計算による異常検出を行い，ボリンジャーバンドアルゴリズムの有効性を評価する．

Proposal of the anomaly detection method analyzing syslog data using Bollinger Bands algorithm on event network

Hiroshi Abe[1,2,a)]　Mikifumi Shikida[3,b)]

1. はじめに

1.1 背景

大規模な展示会やカンファレンス，シンポジウムといったイベントでは，来場者や参加者に対してインターネットへのアクセスがサービスとして提供されることがある．これらのネットワークを総称してイベントネットワークと呼ぶ．イベントネットワークはマルチベンダのネットワーク機器を用いてシステムが構築されることが多い．1 週間から 2 週間という短期の準備期間内でネットワークの構築から運用/撤収を行うため，構築されるネットワークが安定稼働するまでに様々なトラブルが発生する．またマルチベンダの機材を用いることにより，機材の互換性に関するトラブルの把握や解決など運用者の経験に基づく問題解決に依存することが多く，トラブル対応の自動化が困難である．

イベントネットワークの運用では，運用者がネットワーク/サーバ機器/ソフトウェアの動作状況を把握するために syslog を解析する手法が用いられる．運用者にとって，マルチベンダ機器が出力する syslog に対する理解不足や未知のログに対する対応または膨大なログ量の解析に関して，出力されるログのどのキーワードがエラーやアラートであるのかを判断することは難しい．運用者が事前に把握している特定のキーワードに基づくエラーハンドリングは可能ではあるが，正常なログの急増や未知のログに対する対処は困難であり，そもそも膨大なログに埋もれて本来発見したい異常を発見できない場合もある．

本論文では，多数のマルチベンダ機器が混在する大規模なイベントネットワークにおいて，運用者にとって未知のログが多数出現する過酷な環境下であってもログの総量から異常を読み取り，運用者へと効率的に通知可能なアルゴリズムの提案を行う．提案アルゴリズムでは株式市場で用いられるテクニカル分析手法を導入し，正規分布に基づく確率理論からログ総量の突発的な上昇や下降を検知する．本手法を用いることで誤検知を減少させ，運用者への現実

1 株式会社 IIJ イノベーションインスティテュート
2 北陸先端科学技術大学院大学
3 高知工科大学
a) abe@iij.ad.jp/h-abe@jaist.ac.jp
b) shikida.mikifumi@kochi-tech.ac.jp

図 1 VMware vRealize LogInsight を使った syslog 監視例

的な異常発生回数の通知が可能となり，トラブルへの初動対応の高速化が行える可能性がある．

本提案ではマルチベンダ機器が投入される大規模なイベントネットワークの一例として，Interop Tokyo[1] で構築される ShowNet[2] で収集された syslog を用いて実データをもとに異常状態の分析を行う．

1.2 ShowNet とは？

ShowNet は，毎年千葉幕張メッセで開催される Interop Tokyo 内で構築される最新のネットワーク機器や技術を集めた相互接続検証とデモンストレーションを行う実験ネットワーク環境である．また，出展社へのインターネットアクセスをサービスとして提供する側面もあり，実験とサービス提供の 2 面性を有したイベントネットワークである．

ShowNet を構成する機材は，世界で初めて展開されるような最新の製品が多く，機材数は数百を超える．ShowNet には実験ネットワークという一面もあり，試作レベルの機材やソフトウェアが展開される．ShowNet を運用するメンバーは，これらの機材やソフトウェアを組み合わせサービスを提供するネットワークを運用構築する．

システム構築を安定的に行うために機材から出力される syslog を収集分析しバグやエラー，構成の成否を判断する運用が行われる．ShowNet を構成する機材は多岐にわたり，機材やソフトウェアを管理する運用メンバーは機材からどのようなログが出力されるかを事前に知ることは難しい．

1.3 ShowNet における syslog 監視

ShowNet では，監視システムの 1 つとして syslog を収集し分析するシステムが運用される．主にログの可視化を行うことが目的だが，機能の一つとして特定時間内に発生したログからキーワードを抽出し，閾値を超えた場合に運用者にアラートを通知することができる．例として図 1 に VMware vRealize LogInsight [3](以下，LogInsight) を使用した syslog 監視をあげる．LogInsight の機能として，

- ダッシュボードによる条件抽出したログの可視化
- 特定キーワード (OSPF down/BGP down/Storm detection など) の出現監視
- 特定キーワードのマッチング回数監視 (閾値ベース)

などが提供される．キーワードマッチングをトリガーとしてトラブルの原因究明を行う訳だが，マルチベンダの機材が多いために運用者は未知のログを扱うことが多く，どのようなキーワードがトラブルの原因になるかを瞬時に判別することは難しい．

ログの意味解析を行い，スコアリングに基づく異常値解析を行うことは可能ではあるが，キーワードの出現頻度だけでは，そのログが正常か異常かの判断を行うことは運用者にとって難しい．機械学習を用いて，教師データを精査し精度を上げ異常検知を行うことも可能ではあるが，イベントネットワークの特徴として開催期間が短すぎてネットワーク安定状態における学習データを集めることが難しい．また ShowNet の特性として実験ネットワークの意味合いも強く，debug メッセージや必要のない info メッセージも大量にログ出力され，機械学習を行う上ではノイズとなるデータが多すぎる．

そこで本論文では，ShowNet で集められるマルチベンダ機器から出力される膨大な syslog 実データの総量を移動平均と標準偏差を用い集計比較することで，計算量が少なく軽量に計算可能なアルゴリズムを利用し，運用者へ異常検知のトリガーとなる事象を通知する手法を提案する．

1.4 本論文の構成

2 章では関連研究に関する調査と問題点を提示し, 3 章では提案手法を示す．4 章では評価の前提と手法を開示し, 5 章では結果について述べる．6 章では得られた結果から考察を行い，7 章でまとめと今後の課題を述べる．

2. 関連研究

時系列データに周期的な規則性がある場合には，データの波形を予測し外れた場合に異常値とする Holt-Winters 法 [4] のような予測アルゴリズムが利用できるが，ShowNet は開催期間が短いためデータの周期性を観測することは困難であり，周期性に頼るアルゴリズムは適さない．

時系列データにおいてイベントが急増したことを検出する手法の一つとして，Jon Kleinberg のバースト検知アルゴリズム [5] がある．Kleinberg のバースト検知アルゴリズムでは，解析対象のテキストに含まれるキーワードに対し，確率モデルで定義されたコスト計算を行う．このアルゴリズムはバースト状態よりも定常状態に遷移する特徴があ

り，一時的なバーストに反応しにくくなるという特性がある．syslog のような時系列のログを解析し異常を検知するには有効な手法であるが，ログの意味解析を行う必要があり，ログの総量が多い場合には計算量が必然的に増加する．

また，バーストの変化点に着目するアルゴリズムとして ChangeFinder[6] の手法がある．ChangeFinder は統計的な処理を行い外れ値ではなく変化点を見つけるアルゴリズムで，ログ総量のような時系列なデータの値の急増のように定常状態を設定できないデータに対して有効に働く．しかしながら局所的な値の変動に関しては，スコアが平滑化され，時系列に連続する大きな変動ほど異常状態を見つけられない．一瞬の突発的なログ増加に関してはスコアが低くなり異常ではないと判断される可能がある．

運用者にとって未知のログメッセージが出現した場合に，運用者が障害と定める単語との関係性を計算し，障害キーワードに近いと判断した場合に運用者にアラートをあげることが可能となる．Google が提供する word2vec[7] の様な機械学習ライブラリを用いることで，未知のキーワードと障害キーワードの関連性を計算することが可能となるが，他の手法同様にログの意味解析が必要となる．

本研究では，syslog の総量を時系列なデータとして解析を行う．既存研究の手法では計算量が多い場合や，誤検知が多発する可能性が高い．

3. 提案手法

3.1 提案概要

本研究では，計算が軽量で誤検知が少ないアルゴリズムとして，正規分布に基づいたアルゴリズムを採用する．短期間で構築されるために安定状態を定義しづらいイベントネットワークの特徴を再現するため，ShowNet で収集された syslog を対象とすることで，大規模なイベントネットワークの異常を検知する．

ShowNet では以下のようなログの状態が発生した場合に異常状態であると想定している．

- 機材への攻撃によるログの急増
- 機材の不具合によるログの急増
- 機材の不調によるログの急増
- 機材の設定ミスによるログの急増
- 大量な正常アクセスによるログの急増
- ウィルスやワームが発生することによるログの急増

本提案において syslog 総量急増の検出は移動平均と標準偏差を用いる．具体的なアルゴリズムとしてボリンジャーバンド [8] を用い，ログ総量から異常を検知する．

3.2 ボリンジャーバンド

ボリンジャーバンドは株式の取引で利用されるテクニカル分析手法の 1 つで 1980 年前半に John Bollinger により公表された．移動平均を示す線と，その上下に値動きの幅

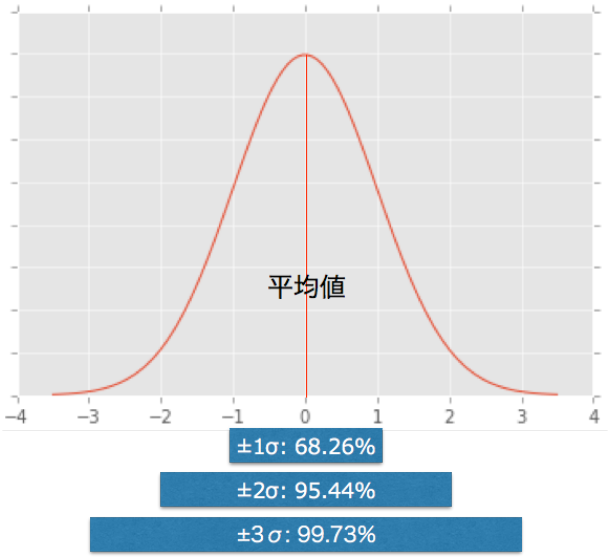

図 2 正規分布と σ の確率

を示す線を加えた指標のことをいい，価格の大半がこのバンドの中に収まるという統計学を応用したテクニカル分析の一つである．

ボリンジャーバンドは正規分布に基づく理論である．統計学の正規分布理論では，図 2 で示すように，

- 平均値± σ (標準偏差) に収まる確率は 68.26%
- 平均値± 2 σ (標準偏差) に収まる確率は 95.44%
- 平均値± 3 σ (標準偏差) に収まる確率は 99.73%.

となる．

ボリンジャーバンドの考え方では，株価は 95.44%の確率で± 2 σ のバンド幅に収まると仮定され，株価が+2 σ のラインを超えた場合には株が買われ過ぎであると判断し，逆に-2 σ のラインを株価が下げた場合には株が売られ過ぎであると判断され，適切な価格へ戻ることが予想できる．

つまり，

- 移動平均 + 2 σ (標準偏差) を UpperLimit(上限値)
- 移動平均 - 2 σ (標準偏差) を LowerLimit(下限値)

として上限値と下限値を採用し，それらの値と株価の比較により適正価格かどうかの判定を行う．

移動平均 (単純移動平均) は以下の式として表すことができる．

$$\overline{x} = \frac{1}{n}\sum_{i=0}^{n-1} x_i$$

新しい値を移動平均の計算に加えたい場合には，現在の移動平均の値に対し，新しい値を加え古い値を除くことで求めることができ，総和を求め直す必要はないので軽量な計算で済む．

また標準偏差 (σ) は以下の式で求めるられる．

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1} (x_i - \overline{x})^2}$$

表 1 実験環境

OS	CentOS 7.2
開発言語	Python 3.5.1
CPU	Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz
メモリ	128GB

$$= \sqrt{\frac{1}{n^2}\left\{n\sum_{i=0}^{n-1}x_i^2 - \left(\sum_{i=0}^{n-1}x_i\right)^2\right\}}$$

標準偏差 (σ) に± 2 をかけた値を，UpperLimit, LowerLimit として用いる．

3.3 syslog 総量計算への応用

本提案では syslog の総量の推移に関して株式取引と同様に統計的な推移が存在すると仮定し，移動平均± 2 σ (標準偏差) を超えた場合，もしくは下回った場合には異常状態であると判定する．syslog の総量はシステムが安定していれば株式と同等に一定範囲 (± 2 σ の間) で推移すると仮定すると，95.44%の確率で総量は UpperLimit と LowerLimit の間であるバンド内を推移ことを意味し，バンドの上限を超える，もしくは加減を下回る確率である 4.56%を異常値として検出する．

4.56%が異常値として適切な値かどうかは，監視運用者の判断に任せられることになるが，初期の気づきとして運用対応を行う現実的な回数内に収まれば運用に適用可能と仮定する．

4. 評価

4.1 実験概要

実験環境を**表 1** に示す．本実験では，ShowNet に接続された機材が出力した管理ログを収集して，ShowNet 終了後に実験環境において Python を用いて解析を行った．解析対象の syslog の容量は約 6.4GB で，行数にして約 4,350 万件を対象とした．なお可読できないバイナリ形式で出力されたログに関してはノイズとして除外してから解析を行った．

4.2 手法

本提案では，ログの意味分析を行わずに時間あたりのログ行数の出現回数を集計分析する手法を用いた．ログの総数を分析するにあたり，syslog から分析には必要ではない情報の削除を行った．syslog フォーマット [9] は大きく，ヘッダー部とメッセージ部からなる．ヘッダー部は，

- タイムスタンプ
- デバイス名

を必ず含み，タイムスタンプはローカル時刻でフォーマットは”Mmm dd hh:mm:ss”となる．また，デバイス名はホスト名または IP アドレスとして定義される．メッセージ部は，フリーフォーマットでテキストメッセージが出力される．

```
1 import pandas as pd
2 df = pd.read_csv('./2016-06-06-syslog.log'
      , delim_whitespace=True, ...)
3 count = df.groupby(pd.TimeGrouper('1Min'
      )).count()
4 mean  = count.rolling(window=60).mean()
5 std   = count.rolling(window=60).std()
6 std_plus  = std.apply(lambda x: x * 2)
7 std_minus = std.apply(lambda x: x * -2)
8 upper_limit = mean.add(std_plus)
9 lower_limit = mean.add(std_minus)
```

図 3 サンプルコード

ログの総数から分析を行うために本提案では，タイムスタンプとデバイス名の 2 カラムを用いて分析を行った．計算速度を速めるため，メッセージ部の情報を削除したファイルをフィルタプログラムにより csv データとして作成した．

事前処理を行った csv ファイルを，Python のデータ解析ライブラリである pandas[10] を用いてログの総量を時系列に解析した．読み込んだ csv データは pandas で定義される DataFrame 形式で処理される．DataFrame は，pandas で定義されるデータ構造の一つで，二次元のテーブルとしてデータを定義できる．各行，各列にはラベルをつけることができ，1 行を 1 つのデータとして表計算ソフトウウェアの様に処理できる．DataFrame 形式で読み込んだデータに対して，基本的な算術計算をメソッドとして呼び出すことができる．本提案では，移動平均と標準偏差を扱うがこれらに関しても，DataFrame に対するメソッド呼び出し (移動平均: mean メソッド，標準偏差: std メソッド) を行うことで実装する．

処理を行う前提は以下とする．

- 1 日ごとのログ総量を計算/描画対象とする
- 1 分単位でデータをグルーピングする
- 過去 1 時間分のログ総量を移動平均に利用する (0 時台の計算には前日の 23 時のデータを読み込む)
- 1 分単位のグルーピングにより，移動平均/標準偏差のウィンドウサイズは 1 時間で 60 とする

総量，移動平均，標準偏差を求めるサンプルコードは図 3 となる．

各行の処理は以下を意味する．

1) pandas ライブラリの読み込み
2) ログファイル (csv ファイル) の読み込み
3) DataFrame のデータを 1 分単位でまとめて集計
4) mean メソッドで移動平均値を計算
5) std メソッドで標準偏差を計算
6) 標準偏差の 2 倍を計算 (+2 σ)

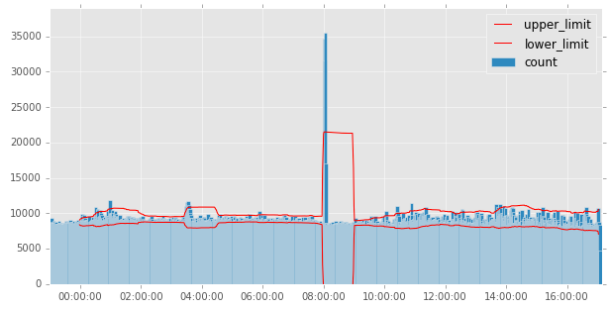

図 4 1 日のログ総量分析例

表 2 アラートレベル

Level	異常値の範囲
Low	1-100
Middle	100-1000
High	1000 以上

7) 標準偏差の-2 倍を計算 (-2 σ)
8) 移動平均と標準偏差の足しあわせ (UpperLimit)
9) 移動平均と標準偏差の足しあわせ (LowerLimit)

入力データの例外として，5/27 のみ初日ということで前日からの学習データは存在しない．

これらの計算データから画像を生成した例が図 4 となる．青い棒グラフである count が 1 分ごとにまとめられたログの総量を表す．総量の上方の赤い折れ線グラフが UpperLimit を意味し，下方の赤い折れ線グラフが LowerLimit を表す．ログの総量が UpperLimit と LowerLimit の範囲を推移する場合には，ログの総量は異常状態ではないと判断する．

本提案では異常値を見つける方法として，ボリンジャーバンドの上限のみにまずは着目した．つまり移動平均+2 σを指す UpperLimit が総量の合計を超えた回数を集計した．この時に UpperLimit を超える確率は，正規分布の+2 σのみとなり 2.28%となる．UpperLimit にのみ着目する理由は，一般的に機器の異常やネットワークの異常時には syslog が大量に出力されるという運用者の経験則に基づく．また正常アクセスであっても，DoS(Denial of Service) のような攻撃を受けた場合にはアクセスログが大量に出力され，上限値超えに気がつくことで正常ではない状態であると判断できる．

さらに異常値判定の精度を考慮し誤検知を抑えることを目的として，異常値が+2 σを超えた値の範囲に着目し，アラートのレベル分けを行うこととした (**表 2**)．アラートレベルは，Low, Middle, High の 3 つに分け異常値の範囲を超えた値が小さい場合には，+2 σを少しだけ超えたと判断し通知を行わない，という異常通知回数の軽減が行えるか実験を行った．

表 3 アラート発生率

日付	ログ総数	時間スロット数	アラート回数	発生率
5/27	192	617	34	5.51%
5/28	181,285	1440	111	7.71%
5/29	552,579	1440	96	6.67%
5/30	821,363	1440	88	6.11%
5/31	617,368	1440	71	4.93%
6/1	917,368	1440	82	5.69%
6/2	1,949,738	1440	91	6.32%
6/3	1,771,956	1440	69	4.79%
6/4	2,108,661	1440	75	5.21%
6/5	3,177,122	1440	71	4.93%
6/6	3,297,654	1440	51	3.54%
6/7	2,702,382	1440	67	4.65%
6/8	3,186,363	1440	87	6.04%
6/9	12,769,834	1440	65	4.51%
6/10	9,446,694	1083	65	6.00%
合計	43,500,499	20420	1123	5.50%

5. 結果

5.1 アラート発生率

ShowNet の期間中に発生した syslog の総量とボリンジャーバンドの上限を超えたアラートの発生率を**表 3** に示す．

表 3 は，
(1) 日付
(2) ログ総数
(3) 時間スロット数
(4) アラート回数
(5) 発生率
の 5 項目からなる．ログ総数は，日付ごとに集計した syslog の総数を表す．時間スロット数は，ログの集計単位 (1 分単位:60 秒) を 1 日 (86400 秒) で割った数 (86400/60=1440) を表す．5/27 の時間スロット数が 1440 より少ないのは，ログ収集が開始された当日なので，1 分ごとの集計が 192 回しか行われなかったからである．syslog の収集機構は，5/27 の午後には動作していたが，ラックへと積載された各機器の syslog 出力の設定時間が違うため，収集開始時刻も各機器により異なる．そのため ShowNet 全体として syslog を収集開始した時刻を定めることはできない．また 6/10 の時間スロット数が 1440 より少ないのは，17 時に ShowNet がシャットダウンされ撤収が開始されたためである．

アラート回数は，ボリンジャーバンドの UpperLimit を超えた回数を表し，発生率は時間スロット数に対して UpperLimit を超えた回数をパーセンテージで表したものとなる．ログ総量の合計から算出した ShowNet 期間中の全アラート発生率は 5.50%となり，集計されたログの 94.5%は UpperLimit を超えなかった．

ShowNet の開催期間は主に準備期間 (Hotstage) と会期

表 4 アラートレベル集計

日付	アラート回数	Low	Middle	High
5/27	34	34	0	0
5/28	111	86	25	0
5/29	96	30	64	2
5/30	88	30	42	16
5/31	71	30	30	11
6/1	82	32	38	12
6/2	91	27	39	25
6/3	69	38	23	8
6/4	75	26	38	11
6/5	71	25	39	7
6/6	51	15	22	14
6/7	67	28	36	3
6/8	87	24	57	6
6/9	65	25	39	1
6/10	65	19	35	11
合計	1123	469	527	127
割合		41.76%	46.93%	11.31%

準備期間，会期の 3 つに分けられる．Hotstage は 5/27 から 6/3 まで, 会期準備期間は 6/4 から 6/7 まで, 会期は 6/8 から 6/10 までとなる．Hotstage 期間中は，新しい機材の繋ぎこみやネットワークの構築，ソフトウェアの稼働などが進められ，syslog の総量が日を追うごとに増加する．会期準備期間には，出展者が出展準備のために会場に持ち込んだ端末を ShowNet へと接続し始める．会期準備期間，並びに会期中には 200 万行から 1,200 万行ほどでログの総量が推移しているが，ログの総量が増加してもアラートの発生率は 3%台から 6%台の間で推移している．

5.2 アラートの回数とアラートレベルの割合

次に表 2 に示したアラートレベルごとのアラート割合を集計した．**表 4** がアラートレベルごとに分けた集計結果となる．全アラート回数のうち，Low アラートが 41.76%, Middle アラートが 46.93%, High アラートが 11.31%という割合を占める．Low アラートと Middle アラートがアラートの大部分である約 88%を占めている．

5.3 特徴的な異常値の分析

ShowNet の期間中にログ総量が大きく跳ね上がる事例が何度か発生していた．これらは全て High アラートであり，その中から特に目に付いた特徴的な異常値を syslog にどのようなメッセージが出力されていたかを確認し，3 つのパターンを分析してみる．

5.3.1 6/6 の場合

6/6 に発生した異常で大きく変化が見られるものとして，18 時過ぎに発生した総量の突然の急上昇と急減が見られる (図 5)．それ以外にも異常は発生しているが，+2 σを大きく超えたのはこの事象となる．

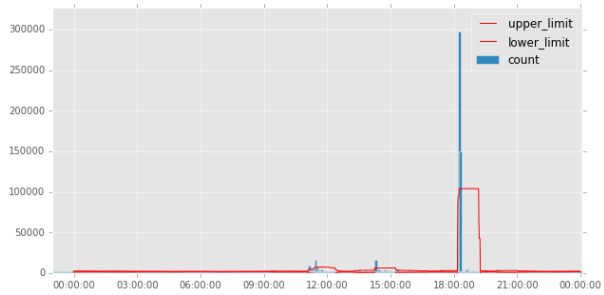

図 5 6/6 の異常値

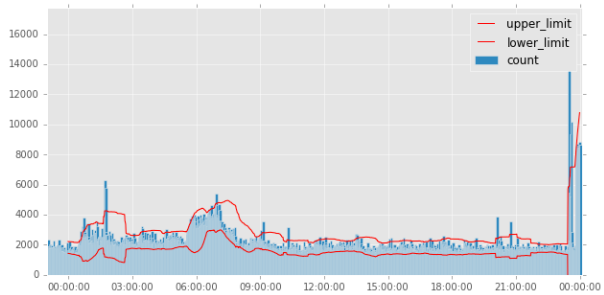

図 6 6/8 の異常値

この時に実際に出力されていたログの内容は以下となる．

```
SNMP Get request is recieved. : ...
SNMP Get response is sent. : ...
```

SNMP Get の request と response が対となり，18:10-18:16 の間に数十万行のログが出力されていた．このログは ShowNet のラックへ搭載されているアグリゲーションスイッチへのアクセスで，スイッチが保持するすべての OID に対しての SNMP Get リクエストが発行され，レスポンスが返されていることが記録されていた．6/6 には 18:10-18:16 の時間以外の SNMP へのアクセスが見られなかったため，手動での SNMP アクセス試験か，ログ出力負荷による ACL の投入による SNMP の遮断，もしくは SNMP を処理する Daemon の停止が行われたものと推測する．ボリンジャーバンドの上限値を大きく超える値となり，異常検知は成功している．

5.3.2 6/8 から 6/9 の場合

ログの総量が 6/9 から跳ね上がっていることが表 3 から見て取れる．6/1 から 6/8 までのログ総量は，100 万行弱から 300 万行強で推移しているのに対し，6/9 の総量は 1,200 万行を超えている．ログの急増は正確には 6/8 の 23 時頃から発生していることが図 6 から見て取れる．それ以降 6/10 の会期終了後まで 1 分平均で 9,000 件を超えるログが出力されている．

6/8 の 23 時台のログを調べてみると，6/6 と同様のアグリゲーションスイッチへの SNMP アクセスが再開されていた．これ以降，同スイッチに対する SNMP アクセスは定期的に行われ，会期が終わるまでこのペースを維持する形

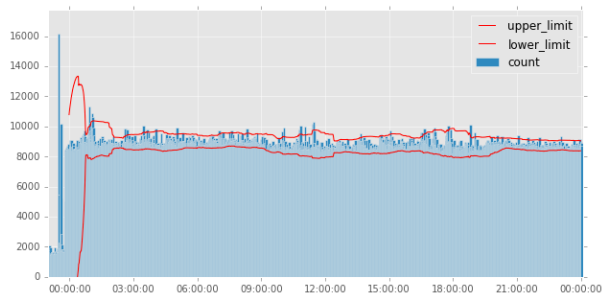

図 7 6/9 のログの推移

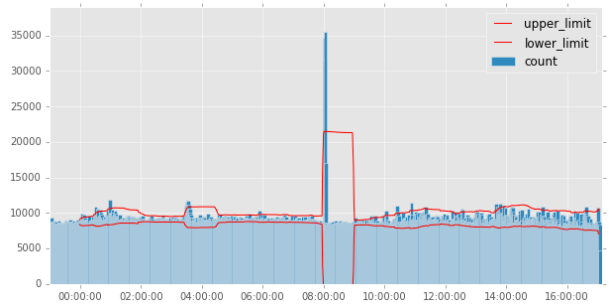

図 8 6/10 の異常値

でログが出力されていた．UpperLimit を超える異常は 6/8 に High アラートとして急増時に検知できており，6/9(図 7) には正常な状態としてボリンジャーバンドの幅の中で総量が推移しており，突出した異常とは判定されていない．6/9 は準備期間も含めた開催期間の中でログイベントの回数が 1200 万件を超える一番多い日であったが，アラート発生率は 4.51%と比較的少ない．

5.3.3 6/10 の場合

図 8 で示すように，6/10 の 8 時過ぎに突然ログの総量が急増している．ログの内容を確認してみると，ShowNet のバックボーンルータと攻撃トラヒック検知時にトラヒック経路を捩じ曲げるための，NFV(Network Functional Virtualization) をベースとした仮想ルータとの間でコネクションを維持していた BGP(Border Gateway Protocol) 接続にトラブルが発生していた．トラブルが収束するまで 5 分程度かかっているが，その後ログの総量は元の水準へと戻っている．ここでも，ボリンジャーバンドの上限を超える異常を High アラートで検知できており，ボリンジャーバンドと High アラートの組み合わせが有効に働いた．

6. 考察

6.1 アラートの発生率について

5.1 で示したアラート発生率の結果より，ShowNet における syslog の総量はボリンジャーバンドで仮定した+2 σ 以内である 2.28%を上回る水準となっており，統計予想回数を上回りアラートが発生していた．これは株式市場ほどイベントネットワークは安定しておらず，純粋なボリンジャーバンドのアルゴリズムでは，正規分布の予想値から外れることを意味する．しかしながら，会期が近づくにつれログの総量は純増しているがアラートの発生率はログの総量に比例して増加はしていない．つまりボリンジャーバンドのアルゴリズムを用いることでログの量が増加してもアラートの発生率はある水準で収まることを意味しており，統計的な正規分布の規則には当てはまっていると言える．

結果として異常通知回数はボリンジャーバンドの統計範囲には収まらなかったが，94.5%の割合で集計値がボリンジャーバンドの UpperLimit を下回っていた．この割合が実運用に耐えうる数値かどうかについては運用者の判断基準による．

6.2 レベル分けの有効性

次に 5.2 で示したアラート回数とアラートレベルの割合については，Low アラートと Middle アラートが約 88%を占めており，これらのアラートを取り除き High アラートのみを通知すれば，異常通知を減らす意味では有益なフィルタとなりうる．

High アラートを検知した具体的な例を 5.3 で 3 事例ほど分析したが，集計の値がボリンジャーバンドの UpperLimit を大きく超える場合には，ネットワーク内で実際に異常が起きていると判断できた．今回は決め打った閾値でアラートのレベル分けを行ったが，閾値を動的に計算することでアラートレベルを賢く実装し，フィルタの精度を向上できる可能性がある．

6.3 他の研究との比較

他の研究で必要なログの意味解析を行わないため，本研究では syslog のヘッダーのみを処理対象とし集計を行った．これは意味解析を行う計算を省略できるばかりか，集計処理を行う場合にも対象となるデータ量が少なくなり計算を高速に行える．またボリンジャーバンドに必要な計算は移動平均と標準偏差の計算のみであるため，全体的なアルゴリズムの計算量を少なく押さえられる．これは，他の手法に対して計算量が少なく異常検知が行えることを意味する．

6.4 イベントネットワーク特有の問題への対応

ShowNet のようなイベントネットワークでは，5.3 で示した例のようなログの急増が容易に発生する．ログの意味解析を行う手法の場合，ログの急増により処理しなければならない対象ログも急増し，解析処理自体が遅延してしまい運用者への異常通知が遅れる可能性がある．

イベントネットワーク運用はサービス提供を行うネットワークであり，トラブル対応の速度がネットワーク品質に大きく影響する．本手法のようにログの総量にのみ着目し，かつ統計的手法によって異常検出計算も軽量なアルゴリズムを用いることで運用者への異常通知が高速化し，対

応速度を重視した運用を行うことが可能となり，イベントネットワーク以外の様々なネットワークにも適用することで運用の品質を向上することが期待できる．

一般的に開催されるカンファレンスやシンポジウムの開催期間は長くて一週間程度や短くて 2-3 日という中でイベントネットワークが構築されるが、ShowNet は 2 週間という比較的長い時間をかけ構築運用される。本提案手法では，1 時間分のウィンドウ幅でデータが学習され計算される．短期間のイベントネットワークおいても，学習に必要なデータを短時間で蓄積でき計算することができ本提案アルゴリズムが有効に働くと推測する．

6.5 実データの利用

ShowNet という実際に運用構築される大規模なイベントネットワークにおける，マルチベンダ機材が混在する特殊な環境において収集された syslog の実データを用いて評価が行えたということは，本実験がシミュレーションではなく，実際の運用へと応用可能な大きなアドバンテージとなる．

7. まとめと今後の課題

7.1 まとめ

ShowNet という大規模でマルチベンダー機材が大量に投入されるイベントネットワークでの異常検知を，syslog の総量を集計してボリンジャーアルゴリズムを用いて解析し検知することは，+2 σ の値をわずかに超える結果となったが，94.5%の割合で UpperLimit を下回った．そこにアラートのレベル分けを追加することで，小さな変動を除去するフィルタの役割となり，大きな変動のみを通知する機構として動作することが有効に機能した．

ログの意味を事前に知りえないという特殊な環境であるがゆえに，ログの意味解析をあえて行わず本来は意味解析が有効である機械学習や意味解析による統計分析では実現できない，軽量で高速に計算できる単純なアルゴリズムが有効に働く．結果として運用者に対してトラブル対応への初動を素早く行動させることが可能となり，イベントネットワーク運用に大きく貢献できる可能性がある．

7.2 今後の課題

今回の結果から，ボリンジャーバンドを用いるアルゴリズムがイベントネットワーク運用に有効に働くと判断できるので，アルゴリズムを組み込んだ実システムが動作する基盤の設計と実装を行い，次年度の ShowNet 運用へと組み込むことを目標とする．実システムとして動作するには，リアルタイムに処理する基盤が必要となる. 本実験では ShowNet で収集したログを事後処理として分析をしたが，リアルタイム処理を行うためにはログ到達時間の遅延やリアルタイム処理の処理時間/ログの集計間隔など考慮する必要がある．

本実験ではレベル分けを閾値ベースで行ったが，動的な閾値の計算を行いレベル分けを行う方法が考えられるので今後の課題とする．今回の分析では UpperLimit だけに着目し，ログの急増を対象としたが，LowerLimit に関してどのような異常値を捉えることが可能かの分析は今後の課題とする．

また，総量にのみ着目するのではなく送信元ホスト別に異常を検出する仕組みを作ることで，運用者に対し初動としての気づき以上の，直接的な問題解決の提示を行えるプロアクティブな syslog 監視システムの実装を検討したい．

本論文で扱った syslog は，機器やソフトウェアのマネージメントに必要なログのみであって，セキュリティ機器が出力するセキュリティに関するログはターゲットには含まれていない．本アルゴリズムをセキュリティ機器のログにも対応させることにより，セキュリティ機器の運用に関しても異常検知の点で貢献できる可能性がある．

参考文献

[1] Interop Tokyo, http://www.interop.jp/
[2] ShowNet, http://www.interop.jp/2016/shownet/
[3] VMware vRealize Log Insight, http://www.vmware.com/jp/products/vrealize-log-insight.html
[4] Kalekar, Prajakta S.: Time series forecasting using holt-winters exponential smoothing. Kanwal Rekhi School of Information Technology 4329008 (2004): 1-13.
[5] Kleinberg,J.: Bursty and Hierarchical Structure in Streams,Proc,8th SIGKDD，pp.91101,2002.
[6] J. Takeuchi and K. Yamanishi : A Unifying Framework for Detecting Outliers and Change Points from Time Series, IEEE transactions on Knowledge and Data Engineering, vol.18, no.4, pp.482-492, 2006.
[7] word2vec, https://github.com/dav/word2vec
[8] Bollinger, J. : Bollinger on Bollinger Bands. McGraw Hill, 2002
[9] Gerhards, R : RFC 5424,The Syslog Protocol, March 2009, https://tools.ietf.org/html/rfc5424
[10] pandas, Python Data Analysis Library: http://pandas.pydata.org/