サーバダウン時における ログファイルの非圧縮による検索時間の短縮

金子 拓磨[1]　高橋 風太[1]　大野 有樹[2]　串田 高幸[1]

概要：ログサーバはログを圧縮することでストレージの使用量を削減することができる．それにより，より多くのログを保存しておくことが出来る．課題は圧縮を行うことで非圧縮時に比べてログサーバ内のログを検索するときの時間が増加してしまうことである．本稿ではアプリケーションサーバがダウンしたときにログサーバに溜まるログを非圧縮の状態で保存しておき，検索時間の短縮を目指す．結果として常に圧縮した状態の検索時間と比較して，検索時間の短縮をすることが出来た．非圧縮ファイルの検索時間は実験の平均値で約 23.20 秒，圧縮ファイルの検索時間は実験の平均値で約 42.96 秒かかっており，検索時間の差が約 19.76 秒となった．

1. はじめに

背景

自社商品を販売する方法の一つとして EC サイトを用いることがある．EC サイトは Web サイトに自社商品を掲載することで，Web サイトにアクセスしたユーザとオンライン上で商品の売買が出来る．EC サイトにユーザがアクセスした場合，ログファイルにアクセスした記録がログメッセージとして書き込まれる．

ログとは，起こった出来事を一定の形式で時系列的に蓄積した記録のことである[*1]．ログはネットワーク障害，アプリケーションや OS の動作問題を解析するときに参照する．また，セキュリティに関わる問題が発生したときには，ログの内容を解析することで，過去にシステムに対してどのような操作がなされたかを確認することができる[*2]．イベントログやログファイルは，システムおよびネットワーク管理において重要な役割を果たしている [1].

テキストファイルを検索する方法として grep がある．grep は Global Regular Expression Print の頭文字を取ったものである．ソフトウェアエンジニアは日々の保守作業で grep を使用する [2].

ログサーバとはアプリケーションサーバとは別にアプリケーションで生成したログを保存するためのサーバである．大規模のアプリケーションを運用するときにそのアプリケーションとログを分けて保存する．ログサーバは，ログが集まったときにログを圧縮して保存を行う．それによりログサーバ内のストレージ容量が減り，より多くのログを保存することが出来る．しかし，圧縮を行うことで非圧縮時と比較してログを検索するときの時間が増加する．

課題

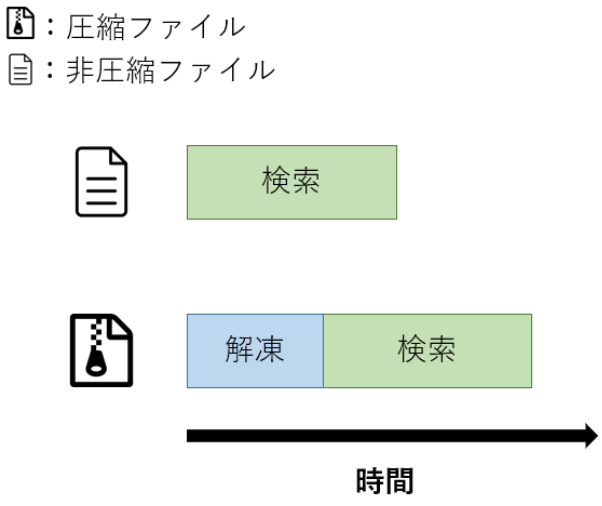

図 1　圧縮ファイルと非圧縮ファイルの検索の差

本稿における課題はログサーバ内にある圧縮されたログファイルを検索するときに，非圧縮状態のログファイルと

[1] 東京工科大学コンピュータサイエンス学部
〒 192–0982 東京都八王子市片倉町 1404-1
[2] 東京工科大学院バイオ・情報メディア研究科コンピュータサイエンス専攻
〒 192–0982 東京都八王子市片倉町 1404-1

[*1] https://it-trend.jp/words/log
[*2] `https://www.manageengine.jp/products/EventLog_Analyzer/collect-eventlogs.html`

比較して検索する時間が増加することである．図 1 は非圧縮ファイルと圧縮ファイルの検索時間の差を示している．非圧縮ファイルは検索するときに実際に検索している時間のみが検索全体の時間となる．それに対し圧縮ファイルは圧縮ファイルを解凍する時間が含まれ，解凍する時間と実際に検索している時間が検索全体に要する時間となる．

各章の概要

第 2 章では，関連研究について述べる．第 3 章では，本稿の提案方式の説明とユースケースの説明する．第 4 章では，提案方式を用いた実際の実装方法と実験方法について述べる．第 5 章では，評価方法と分析手法について述べる．第 6 章では，本稿の議論についてを述べる．第 7 章では，本稿のまとめを述べる．

2. 関連研究

Yu らは，クラウド規模のデータに対して拡張性の高い効率的な多次元メタデータインデックス・検索システムを提供することを目的としている [3]．しかし，本稿では使用しているデータが異なるため解決できるかどうかの判断が出来ない．

Studiawan らは，オペレーティングシステム（OS）ログに異常なアクティビティがあるかどうかを確認するためにディープラーニングを用いた新しいセンチメント分析手法を提案している [4]. 本稿とは使用しているログと述べている提案が異なり新しい分析手法を用いている．

Park らは，グループ内で共有する機密情報はサーバに保存されることがあるため，より厳重なセキュリティとプライバシー保護が要求されるが，暗号化されたデータに対して適用可能な検索方式が存在しないことを述べている．そして，グループ鍵が更新された場合でもサーバ内の全文書を再暗号化することなく暗号文書を検索することが出来る方式を提案している [5]．本稿とは検索する対象のデータが異なり文書ファイルである．

He らは，フリーテキストのログメッセージを構造化されたイベントに変換することを目的としたログ解析において，開発者が既存のログパーサの有効性や実用化のときの限界を気付きにくく，再実装や再設計のため時間と手間がかかることを問題として挙げた．その解決のため現状のログ解析器の特徴調査を行い，1000 万件以上のログメッセージを含むデータセットで有効性を評価した [6]．本稿とは，検索する対象がログであることは同じであるものの，目的が検索時間とは異なりデータセットでの有効性の評価である．

3. 提案方式

提案方式

本稿の提案はログを生成するアプリケーションサーバがダウンしているかに応じて，ログサーバ内に保存されているログファイルを圧縮して保存するかの判断をログサーバ内で行う．

アプリケーションサーバがダウンしたと監視によって判断した部分からログサーバに送られたファイルを非圧縮ファイルの状態で保存する．本稿でのサーバのダウンの定義はサーバが想定外の応答をしたときとする．

図 2 は，稼働時のログの保存方法を示している．この場合はアプリケーションサーバで生成されたログをファイル形式にしてログサーバに転送し，ログサーバ内でファイルの圧縮を行い保存する．以下にユーザのアクセスから圧縮して保存するまでの流れを説明する．まず，ユーザがアプリケーションサーバへアクセスをする．次にアプリケーションサーバはログサーバへログを転送する．監視サーバはアプリケーションサーバがダウンしているか監視している．ログサーバは監視サーバからの異常の通知がない場合にアプリケーションサーバから転送されたログを圧縮して保存する．

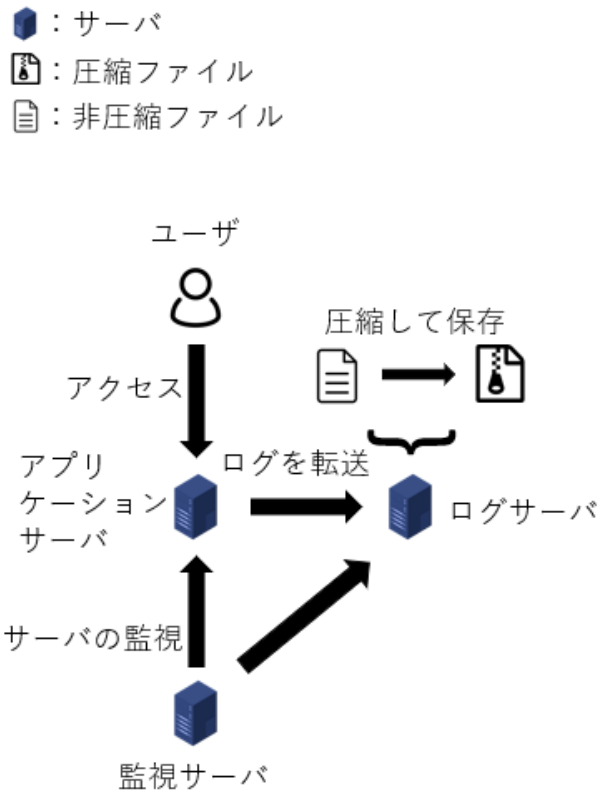

図 2 稼働時のログの保存方法

図 3 は，アプリケーションサーバがダウンしシステムが稼働しなくなり，ユーザがアプリケーションサーバへアクセス不可になった場合を示している．この場合は稼働時と同じくアプリケーションサーバで生成されたログをファイル形式でログサーバに転送するが，ログサーバ内でファイ

ルの圧縮を行わず，非圧縮の状態のまま保存する．以下にユーザのアクセス失敗からログサーバでログを保存するまでの流れを説明する．まず，ユーザがアプリケーションサーバへのアクセスに失敗する．次にアクセスが失敗したときのログをログサーバへ転送する．監視サーバはアプリケーションサーバを監視し，アクセスできないときに異常の通知をログサーバへ送る．ログサーバは監視サーバからの異常の通知を受けて，送られてきたログを圧縮せずに保存する．

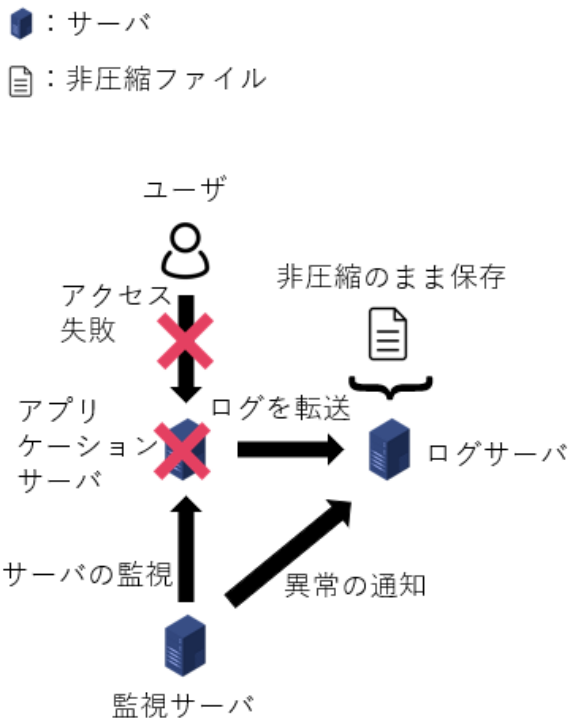

図 3　サーバダウン時のログの保存方法

ユースケース・シナリオ

本稿の提案方式は，パソコンやその周辺機器を販売しているショッピングサイトのアクセスログを想定している．アプリケーションサーバ内でショッピングサイトを運用している．アプリケーションサーバ内で出てきたアクセスログをログサーバで収集をする．

ログサーバのストレージ容量を削減をする．しかしログを削除するのはログを取っている意味がなくなるため取っておくことは前提とする．そのためログを圧縮してストレージ容量の削減をする．ログを圧縮してストレージ容量の削減をしつつ，検索全体の時間を短縮するために検索の対象となるアプリケーションサーバがダウンしているときのログを圧縮せずに保存する．それにより検索全体の時間を短縮することが出来る．

4.　実装と実験方法

実装

本稿の実装は，提案方式を用いて用意したファイルを VM 内で検索し，その検索速度の測定を行った．実装のソフトウェアは，アプリケーションサーバがダウンしていることを検知するシステム，アプリケーションサーバから送られてきたログファイルを提案方式で用いた方法で保存するシステムの 2 つである．

図 4 は実装の全体図を表している．サーバーはアプリケーションサーバと監視サーバ，ログサーバがある．ユーザはアプリケーションサーバにアクセスすることで，ログ生成によってログが生成される．本提案手法では監視システムによってログを圧縮するかどうかの判断をしている．そのため，実装の流れは監視システムとログの圧縮の 2 つに分けることができる．

まず，監視システムの流れについて説明する．監視システムの流れは図 4 の (i) と (ii) が示している．(i) は監視サーバにある監視システムがアプリケーションサーバのログを監視している．アプリケーションサーバのダウンを検出したときに監視システムは (ii) の処理へ移行する．(ii) は監視サーバにある監視システムがアプリケーションサーバのダウンを検出したときにログサーバにある圧縮の判断へアプリケーションサーバがダウンしていることを通知する．

次に，ログ圧縮について説明する．ログ圧縮の流れは図 4 の①から③が示している．①でアプリケーションサーバにあるログ転送はログ生成から出力されたログを検出し，アプリケーションサーバのログをログサーバへ転送する．ログサーバの圧縮の判断では (ii) で通知されたメッセージに従ってログを圧縮するか判断する．圧縮するかどうかの条件は監視システムからアプリケーションサーバがダウンしているかどうかの通知の有無である．アプリケーションサーバがダウンしている通知があった場合，②′ に移行しログファイルの保存先に圧縮せずに保存される．アプリケーションサーバがダウンしている通知がなかった場合，②に移行し，圧縮のプログラムを実行する．③は圧縮されたログをログファイル保存先に指定して出力する．

実験環境

実験環境は ESXi の VM を用いる．

実験に使用するサーバは仮想マシンを用いる．以下に，課題の実験に用いる VM の構成情報を示す．

- VM の構成要素
- OS：Ubuntu-20.04
- vCPU：1 コア
- RAM：1GB

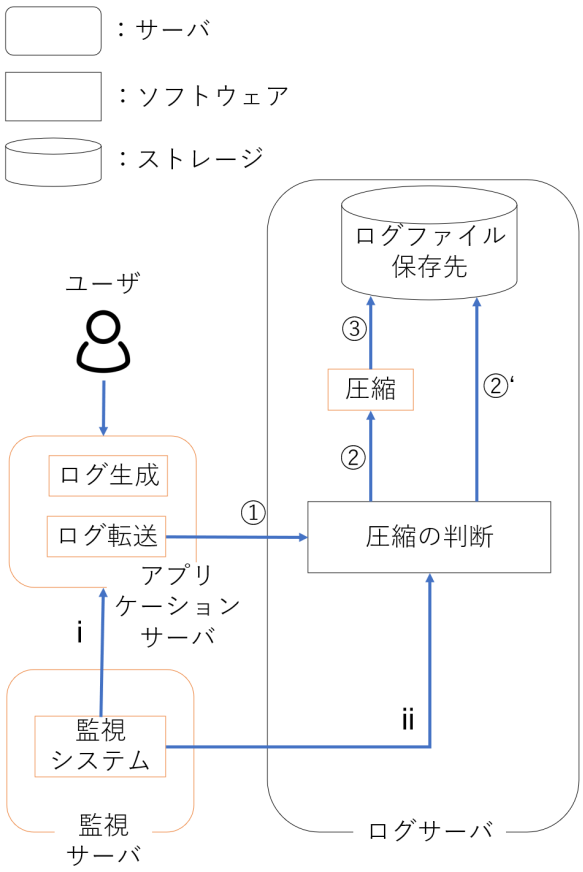

図 4 アプリケーションサーバの監視によるログファイルの圧縮選定

- HDD：30GB

ファイルの検索には grep コマンドを使用した．検索時間の計測は/usr/bin/time コマンドを用いる．ファイルは EClog 半年分 (約 9.2GB) とする．EClog は半年分で 35,157,691 件あった．

5. 評価と分析

EClog を全て圧縮した場合と EClog を圧縮しなかった提案方式を用いた場合を比較し，検索速度を評価とする．

今回実験で用いた条件は，HTTP のステータスコードの 400 番台と 500 番台である．検索のコマンドは grep を使用した．実験時に使用したコマンドは，ソースコード 1 に示す．

ソースコード 1 検索で使用したクエリ

```
/usr/bin/time grep "1.1,"[4-5][0-9][0-9]","
    eclog.csv
```

図 5 に非圧縮ファイルの最小，最大の検索時間と圧縮ファイルの最小，最大の検索時間を示している．図 6 に非圧縮ファイルと圧縮ファイルの平均の検索時間を示している．非圧縮ファイルの検索時間は最小で約 22.80 秒，最大で約 23.86 秒，平均で約 23.20 秒かかっており，圧縮ファイルの検索時間は最小で約 42.40 秒，最大で約 44.96 秒，平均で約 42.96 秒かかっている．この結果により検索時間の差が最小で約 18.54 秒，最大で約 22.16 秒，平均で約 19.76 秒早くなること分かった．なぜ非圧縮ファイルと圧縮ファイルで検索時間の差が出たのかは，圧縮ファイルの検索では検索中にファイルの解凍を行っているため，その時間分検索に時間を要することが挙げられる．

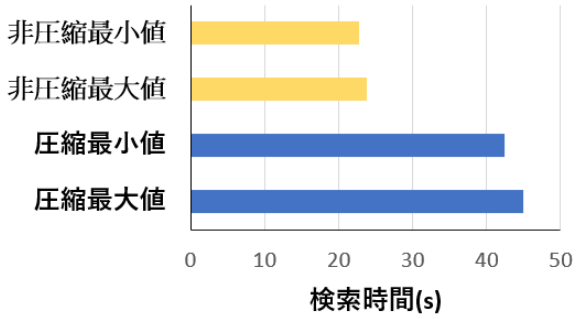

図 5 圧縮時と非圧縮時の検索時間の比較

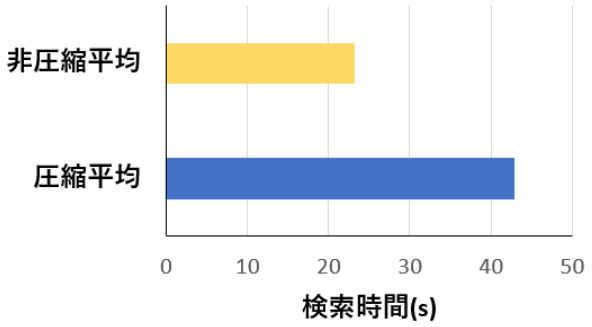

図 6 圧縮時と非圧縮時の平均検索時間の比較

6. 議論

本提案方式では，アプリケーションサーバがダウンしたときにファイルを圧縮せずにログサーバ内で保存する．しかし，この提案ではサーバがダウンする前のダウンに影響した範囲のログが非圧縮として残したログファイルの範囲外の場合も存在する．解決方法として，サーバがダウンしたのログとは別に，その前のログに関しても閾値を用いて範囲を設定し非圧縮ファイルとして保存する．

7. おわりに

本稿では，圧縮ファイルを検索するときに非圧縮ファイルと比較し時間がかかるという課題に焦点を当てた．解決

方法として，稼働時には圧縮して保存しているログファイルをアプリケーションサーバがダウンしたときに非圧縮のまま保存することで検索全体の時間を削減した．今回の実験で削減できた時間は，最小で約 18.54 秒，最大で約 22.16 秒，平均で約 19.76 秒である．検索時間の削減が出来たのは，圧縮ファイルの検索の中には解凍の時間も含まれているためである．

謝辞 本テクニカルレポートを執筆にあたりご指導頂きました東京工科大学院バイオ・情報メディア研究科コンピュータサイエンス専攻の飯島貴政さん，東京工科大学コンピュータサイエンス学部の牧丈晴さん，森井佑誠さんに御礼申し上げます．

参考文献

[1] Vaarandi, R.: A data clustering algorithm for mining patterns from event logs, *Proceedings of the 3rd IEEE Workshop on IP Operations & Management (IPOM 2003)(IEEE Cat. No. 03EX764)*, Ieee, pp. 119–126 (2003).

[2] Abou-Assaleh, T. and Ai, W.: Survey of global regular expression print (grep) tools, *Proceedings of Citeseer, Topics in Program Comprehension*, pp. 1–8 (2004).

[3] Yu, Y., Zhu, Y., Ng, W. and Samsudin, J.: An efficient multidimension metadata index and search system for cloud data, *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*, IEEE, pp. 499–504 (2014).

[4] Studiawan, H., Sohel, F. and Payne, C.: Anomaly detection in operating system logs with deep learning-based sentiment analysis, *IEEE Transactions on Dependable and Secure Computing*, Vol. 18, No. 5, pp. 2136–2148 (2020).

[5] Park, H.-A., Lee, D. H., Zhan, J. and Blosser, G.: Efficient keyword index search over encrypted documents of groups, *2008 IEEE International Conference on Intelligence and Security Informatics*, IEEE, pp. 225–229 (2008).

[6] He, P., Zhu, J., He, S., Li, J. and Lyu, M. R.: Towards automated log parsing for large-scale log data analysis, *IEEE Transactions on Dependable and Secure Computing*, Vol. 15, No. 6, pp. 931–944 (2017).