すばる望遠鏡ステータスログのデータベース化

◯中村京子*1、小杉城治*1、佐藤立博*2、（小林剛志*1）
（*1 国立天文台 アルマプロジェクト *2 国立天文台 ハワイ観測所）

概要(Abstract)

すばる望遠鏡は約 20 年間に渡って運用観測を行っているが、毎日の運用状況の把握のため、望遠鏡の各種ステータスログを定期的に取得している。ただし、それらはデータベース化されておらず、なんらかの問題が発生した際には、手作業主体で状況確認等を行っているのが実情である。そのため、ログの可視化を容易にし、問題の早期発見や発生予測を目指して、目下、望遠鏡ステータスログのデータベース化に取り組んでいる。今回の発表では、大規模データの解析で多くの企業で実績のある、エラスティックサーチ（Elasticsearch）へのデータ投入について紹介する。

1. Elasticsearch とは

Elasticsearch は「全文検索」を行うソフトウェアであり、2010 年に登場した。「全文検索」は、複数の文書から特定の文字列が含まれるものを探し出す作業である。Elasticsearch は Java で実装されたオープンソースソフトウェアである。無料で使用できるが、サポートや高度なセキュリティ機能等を含む有料版もある。今回我々が使用したのは、無料版である。

Elasticsearch の特徴は、クラスタ構成前提の「分散処理システム」、「高速検索」、JSON に対応した「ドキュメント指向データベース」等である。複数ノード構成のクラスタで稼働させることを想定して設計されており、面倒な設定変更なしで運用中でもノードを簡単に追加できる。全文検索エンジンライブラリ Apache Lucene をベースとし、例えばクラスタ構成をとるなどで Lucene の機能を補完し、高速検索に寄与している。さらに JSON 形式のデータを受け入れているため、様々なデータ構造に対応できる。以上、これらは DB との大きな違いにもなっている。なお、Elasticsearch と連動したログ収集および可視化ツールが充実していることも特徴の一つに挙げられるが、それらプロダクトをまとめて「Elastic Stack」と呼んでいる。最後に、REST API にも対応しているため、HTTP プロトコルを使ったアクセスも可能である。特に今回のログ投入では、大規模データ対応の Bulk API 機能を利用し、REST API をインタフェースとして使用した。

ここで、Elasticsearch を導入しているいくつかの企業、組織を紹介する。

「株式会社リコー」 https://www.elastic.co/jp/customers/ricoh

IT 環境の見える化でセキュリティを強化するため、リコーグループのシステム環境のほぼすべての IT デバイスのログを Elastic Stack に集約。

「富士通株式会社」 https://www.elastic.co/jp/customers/fujitsu

セキュリティログの高精度な検知のため、リアルタイムにログを集計、分析する基盤として、Elastic

Stack を活用。

「Docker 社」 https://www.elastic.co/jp/customers/docker

すさまじい量のコンテナライブラリの検索では、従来の DB の検索機能の実装には限界があったため、インフラを Elasticsearch に変更。

「ALMA」

運用中の観測ログを、2017 年 10 月から Elasticsearch に収集。日本が開発する ACA 相関器の問題解析ツール Hastings は、Elasticsearch を使ってエラー原因となるログをピンポイントで抽出。

2. ログ投入試験の概要

2.1 試験システム

まず初めに、試験システムは故小林剛志氏が構築したものであることを明記する。

試験システムでは、天文台クラウドの 3 ノードを使って Elasticsearch クラスタを構成した。OS は CentOS7、Elasticsearch のバージョンは 6 である。CPU、メモリ、ディスク容量等は全ノード共通で、それぞれ 4CPU（Xeon X5675）、8GB メモリ、80GB SSD である。

2.2 すばる望遠鏡ステータスログ

対象ログは、2018 年 7 月 23 日 08:00 から 31 日 07:59:59(HST)に実際にすばる望遠鏡で取得されたものである。もとデータはバイナリ形式のため、tsv フォーマットでテキスト形式に変換されたファイルを使用した。ファイルの種類は 76 あり、一日分のデータは 08:00:00〜16:59:59 と 17:00:00〜翌 07:59:59 の二つに分かれているため、一日分のログファイル総数は 152 個である。なお、一日あたりのログファイルの大きさは平均 1.77GB、八日間の総数は 14GB であった。ところで、ステータス情報は全部で 9448 種のため、Elasticsearch に投入する際のフィールド数（データベースにおけるカラム数に相当）は 9448 個である。なお、ファイル種別ごとにインデックス（データの格納場所）を作成したため、インデックス数は 76 である。

2.3 ログ投入の手順

当初は、Elastic Stack の一つである Logstash を試した。すばるログ対応のコンフィギュレーションファイルをファイル種別毎にあらかじめ作成し、その情報を元に logstash はログファイルを読み込み、データ変換し、Elasticsearch に投入（インデクシングと呼ぶ）する。様々な試みの結果、投入元のマシン性能や投入方法によってインデクシングの時間が大きく変わることがわかったが、投入所要時間がどうしても期待した範囲におさまらず、結局、この方法は断念せざるを得なかった。

次に、Elasticsearch の Bulk API 機能を使ってインデクシングを行った。この方法では、すばるログデータを json 形式に変更し、複数のリクエストをまとめて、REST API を利用して Elasticsearch に送信する形となる。Bulk API は、大量のデータを投入するための言わばバッチ処理のようなものであるが、これにより大幅な時間短縮が達成できた。

3. ログ投入試験結果

3.1 Logstash

様々な条件で、インデクシングにかかる所要時間の多寡を調べた。結果は以下の通り。

・ログファイルのフィールド数が同じ場合、総データ量が少ない方が所要時間は短い。

・ログファイルの総データ量が同じ場合、フィールド数が少ない方が所要時間は短い。

・Logstash がクラスタノード内にあってもなくても、所要時間に大きな変化はない。

・Logstash 実行マシンの処理性能が高い方が、低いものより所要時間は短い。

・Logstash 実行マシン一台を単独で使うより、二台（同程度の性能）で同時インデクシングする方が、所要時間は短い。

特に最後の同時インデクシングに間しては、共に同じインデックスに投入するよりも、別インデックスに投入する方が所要時間は短いことがわかった。また、クラスタ各ノードの負荷をユーザ使用率で調べたところ、二台同時は一台単独よりも平均 10%負荷が高い程度で、サーバにかかる負担は思いのほか低かった。

3.2 Bulk API

一転送単位が 100MB 程度になるようログファイルを加工し、REST API を使ってインデクシングを行った。なおインデクシングの前に、あらかじめインデックスを作成する必要がある。Elasticsearch は「1 インデックスあたり最大フィールド数 1000 個」のデフォルト制限があるため、必要に応じて、インデックス作成時にフィールド数もあわせて変更した。インデクシングにかかった結果は以下の通りである。

・同一規模のデータ量におけるインデクシングの所要時間は、Bulk API が Logstash の 1/4 以下。

・一度に送るデータ量が多いほどインデクシング所要時間は短い。（100MB は 50MB の 7/10 程度。）

ところで、Bulk API がデフォルトで扱える最大データ量は 100MB である。この値は変更可能だが、投入実行マシンやクラスタ側の性能にも依存し、性能が十分でないとインデクシング最中にエラーが起こるため、これの変更はしなかった。

3.3 結果

以下の方法で、すばるログを Elasticsearch にインデクシングした。

・Bulk API 方式

・1Bulk ファイルの大きさは 90MB（100MB では、インデクシングでエラーが出る場合があったため）

・ファイルの種類毎に 1 インデックス（全 76 インデックス作成）

・Bulk 実行マシンは、クラスタ外のものを使用

結果は、すばるログ一日分のインデクシング所要時間が平均 30 分間程度であった。試験に使った 8 日間のログは、4 時間程度でインデクシングができた。現在すばるには約 20 年間分のステータスデー

タがあるので、それらを仮に試験システムにインデクシングした場合の所要時間は（ディスク容量やシステム構成等は無視する）、0.5[h]*365[日]*20[年]=5ヵ月2日2時間、と試算できる。

ちなみに、ログファイルの赤経赤緯等の値の単位変換や時間等の記述形式の変更、TSVからJSON形式へのフォーマット変更、投入用ファイルの90MB分割等、前処理にかかる時間は、一日分のログあたり約15分間であった。この前処理はスクリプトで実行でき、インデクシングと並列実行可能であるので、上の試算には含めていない。

4. 今後の予定

現試験システムでの確認はここまでで、目下、新たなクラスタシステムを構築中である。ALMAコンピューティングが所有しているVMware上で1ノード構成クラスタを試しに作り、VMwareのハイパースレッド機能を無効化した上でインデクシングの所要時間を測定したところ、一日分のログが10分間程度で投入できた。VMwareのハイパースレッド機能はデフォルトで有効となっているが、この機能は、Bulk APIによるインデクシングでは悪い方へ働くようである。

今後やっていきたいことを以下に挙げる。

1) 動作やデータ確認のため、半年間程度のすばる望遠鏡ステータスログのインデクシングを次期システムで試す。

2) Elastic StackのKibana等で投入データの分析を進める。

3) 望遠鏡ステータスログのほかに、観測装置からのコマンドログもElasticsearchに納め、両者を併せたデータの分析を行う。

最終的には、すばる望遠鏡の問題の早期発見、望遠鏡で発生した問題の速やかな原因究明、望遠鏡で将来発生するかもしれない問題の予測等を実現することで、すばる望遠鏡の安定運用にいくばくかなりとも貢献できれば、と願っている。