ISSN:0975-9646

Log Aggregator for Better Root-Cause-Analysis

Anamika, Gayatri Bisht, Kanhaiya and Poonam

Department Of Computer Engineering
Savitribai Phule Pune University
Pune – Maharashtra – 411015

Abstract - Problem tracking in large software systems is a time consuming and difficult task. To do the Root-Cause-Analysis (RCA) of an issue occurred in a system, typically the concerned person has to go through different log files and event data. The complexity and size of the logs makes it often difficult for human operators and administrators to track the problem and perform root cause analysis. A big challenge in this area is to provide the necessary tools and techniques for the user to focus on specific parts of the data logs thus reducing the complexity of the analysis process. So in this paper, a model is proposed to tackle this issue. A better analysis of log files can be done, if aimed logs are specified in unique format and filtered to focus on specific logs so that flow of events or faults can be tracked. More specifically, the proposed framework provides simplified interface to user by displaying logs of specified type in formatted form in chronological order in single pane.

Index Terms : Log Analysis, Root-Cause-Analysis, Chronological Order, Performance.

I. INTRODUCTION

Root Cause Analysis (RCA hereafter) refers to a set of techniques that are performed to discover events or faults that produce a system failure. A common approach to RCA is to analyze log files and identify problems occurred in system. These problems are then examined so that potential causes can be identified. RCA performance is complicated because of the size of the logs considered. This problem becomes more frequent when heavy software systems are analysed for RCA. These systems can have many components each of which may have different logging mechanism so that, the

data logs may often be too many and too complex for a human operator to analyze. So ,the problem becomes is to parse data logs and convert into unique format and to reduce the size of the logged data by filtering out data that are not relevant to a particular application, and to help human operators to focus on events and logs that may relate to root cause of a problem. This paper presents a framework that helps on the analysis of log data and interprets a selected subset of this data for the purpose of RCA as in service oriented systems. In this system the logs/events should be looked into the chronological order in which they have been added. Also, the format structure of the files/sources should be unique. It should show all these log/event from multiple in a single pane, in the order they have occurred (chronological). The tool should be able to load and handle different log files, Windows event viewer etc - irrespective of their log entry format etc (e.g. by defining a RegEx for each format).It should also be able to load relatively larger log files. This paper is structured as follow. Section II presents related research work. Section III provides a description of the components of the proposed framework. Section IV describes the implementation of system in a programming environment. Details about new features and expected result are described in in Section V. Section VI summarizes the future work and conclusions.

II. LITERATURE SURVEY

For developing a new application, first we need to study existing tools which are performing same task. Some of the similar Tools that we have referred for our Project are as follow :

Baretail

A lightweight and effective utility that was especially created in order to help users track and monitor changes in their files. Main features of this tool are Real-time file viewing, Follow tail mode, Tail multiple files, Configurable highlighting, International character sets, Many file formats, Flexible configuration options and storage and Single small executable, no installer is required. But limitation of this tool is that it can be used with only Windows Operating system.

LogExpert

LogExpert is a Windows tail program (a GUI replacement for the Unix tail command). LogExpert is free for non-commercial or commercial use.
Features of this software are : Tail mode, MDI Interface with Tabs, Search function (including RegEx), Bookmarks, A very flexible filter view, Highlighting lines via search criteria, Columnizers(splitting log lines into columns for some well defined logfile formats) Unicode support, log4j XML file support, built in schedular 3rd party plugin support, SFTP support (loading files directly from SFTP), Plugin API for more log file data sources.
You need Visual Studio 2008 (maybe 2005 works as well).
It is capable of automatically detecting log format and can create HTML,PDF and CSV reports. It also supports command line mode. LogExpert is written in C and needs .NET 2.0. Limitation of this software is log files are tab-delimited.

Glogg

Glogg enables you to use regular expressions to search for interesting events in your log files. It presents a result window which, together with complex regular expressions allows easy isolation of the meaningful lines amongst the noise. Glogg has been primarily developed to help spot and understand problems in huge logs generated by embedded systems. It can be equally useful to a sysadmin digging through logs from databases or web servers. The main design goals for Glogg are: it is fast ,no limit on the size of files it can handle, provides a clear view of the matches even in heavily cluttered files. The main features of Glogg are that it runs on Unix like systems,Windows and also on Mac. It also provides a second Window showing the result of the current search.Moreover it is Open Source and thus freely available.

SnakeTail

SnakeTail is a Windows tail program that can monitor growing log files: Monitor large text log files, Windows Event Logs (Without needing administrator rights), Window Modes supported (MDI, Tabbed, Floating) , Save and load entire window session, start and stop services directly. It can load session file at startup when given as command line parameter and sentence highlight with color based on keyword match (includes regex support. It does flitering of Windows Event Logs using regular expressions. It tails log directory where the latest log file is displayed. It works well over remote desktop.

III. PROPOSED MODEL

Model proposed in this paper is a utility tool which will help the programmer to track the application crash and do better Root Cause Analysis. This can be implemnted for a standalone system as well as distributed system.

A. System Design

The tool accesses the logs with respect to the application from the selected sources. Currently, developer rely on experience and expert information to extract useful information out of log files. As an example a service support personnel at a customer site may manually browse through log files to detect cause of system crash. Our eventual goal is to display the logs that developer wants to analyze in a manner which will help him better to find cause of crash.

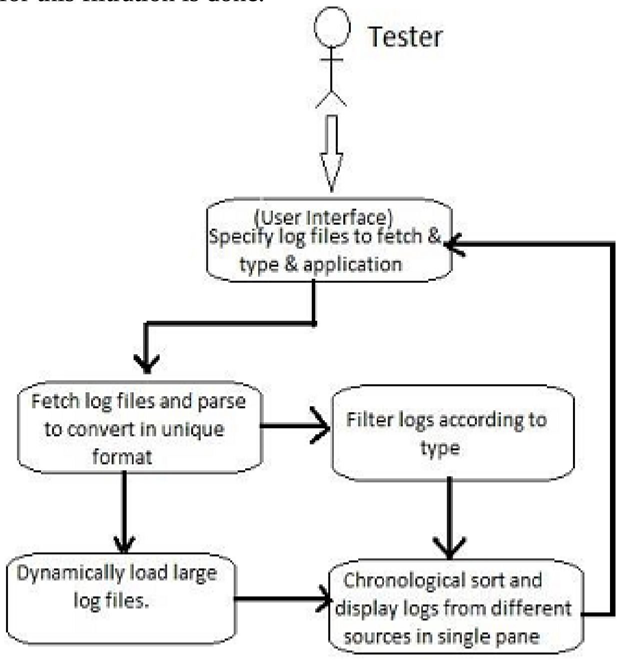
Proposed System is a UI-based system that provides user with a interface to access different type of log files on to a single pane so that user can get better idea of order in which events went wrong for better analysis. System provide user with many features that makes log analysis easy ,fast and efficient like highlighting required logs and differentiate between different types of logs. System consist of different modules as specified in figure.

User specifies the log sources and data points which he wants to analyze by providing path for respective files.The tool will load all given files from different data points.

There are different type of log files present in computer systems like syslog, kernel.log application specific logs etc. All these logs are of different types and are specified in different formats.

for example date in syslog is specified as jan 2 2015 whereas in kernel.log it is specified as 2-01-2015. Hence for better log analysis these logs from different sources needs to be converted into single format. For this parsing of logs needs to be done. For parsing, format of logs are specified in form

of regular expression, logs are matched with this RegEx and converted into unique format so that analysis and comparison between logs can be done. Log files are normally very large, hence for log analysis and interpretation, size of log files need to be minimized and for this filtration is done.


Tester
(User Interface) Specify log files to fetch & type & application
Fetch log files and parse to convert in unique format
Filter logs according to type
Dynamically load large log files.
Chronological sort and display logs from different sources in single pane

Different type of logs are present in a log file for example info logs, error, warning etc. Filtering of log files is done to retrieve precise logs that are needed. For example in some applications we only need error logs so log files can be filtered according for error logs.

Once required logs are retrieved ,these logs from different sources are arranged in chronological order on a single pane so that sequence in which logs are recorded in different log files can be traced ,which helps in finding out order in which different events has occured and root of given problem. This chronological sorting is core feature of this tool which will save user’s time. Additionally logs pertaining to a specific application are highlighted by using keyword based search so that user can easily look on the required logs. For handling large log files, dynamic loading of files is done .In this instead of loading complete file at a time which is time consuming, files are loaded in chunks on demand so that performance of system can be enhanced. Hence this system provides a better way to do Root cause analysis of given problem.

B. System Implementation

Environmental setup required for implementing proposed system includes ubuntu 14.04 as platform,ruby 2.1.2 with rails framework as programming language.Modular paradigm is used for implementing the system.

In first module ,all files from different sources are parsed using Regular Expressions based parser.Format of different logs are specified in form of regular expression in grammar. Grammar for each file format is defined which parses the logs by matching with these grammar and stores them in single format in an array as time, severity and message.

For example for the authentication logs in the auth.log, regular expression can be specified as

```
Feb 2 22:17:01 anamika-HpPavilipn CRON[5389]:
pamunix(cron:session): session opened for user root by
(uid=0)

” date time message ”
date : [0-3][0-9]’-’month’-’[0-9][0-9][0-9][0-9]
month : janjfebjmarch..
message : [a-zA-Z0-9]+
month : Feb
date : 2
time : 22:17:01
message : session opened for user root by (uid=0)
```

In the second module the logs are sorted in chronological order. Here the parser parses the timestamp of each log in one format () and these are sorted in the order of their occurence. For this logs from different sources are merged and sorted in a single array and then displayed on portal in single pane with multiple fields for each requested log file. It is important to know after sorting that to which log file the particular log is from. Here along with each log a flag is used as a mark for the log type so that after sorting it can be easily made which log is from which file.

In next module , filtering of logs is done on basis of their type. Type of logs are specified within the logs (error,warning,info etc) hence firstly type of logs need to be retrieved from logs which is obtained from the flag associated with each log. It is matched with type specified by user. If it matches, it is loaded else it is ignored.

Performance of system can be improved by loading large log files dynamically. Instead of loading whole file at a time, it can be loaded in chunks. This will improve the time required for loading the files from source location.

In last module, highlighting of logs related to a specific application is done. In order to do that, user will specify a keyword which is needed to be searched in logs, then logs having this keyword are searched and will be highlighted with a different color which will provide user with understanding of order of logs related to specific application since all the related logs are displayed in a single pane with the timestamps.

Efficiency of tool can be further increased by implementing it as multi-threaded application. We can have different threads for different modules. Main thread will be UI thread which will be doing integration and synchronization of other threads. This helps in implementing tool as a light-weighted application. Parallel threads are used for sorting, parsing, filtering and for highlighting the logs. This will enhance the efficiency of the utility tool.

IV. Expected Result

Proposed model is expected to fetch the log files and information on basis of information provided by user (location of log files, type ,application),parse the logs into single format, filter them according to type mentioned by user, sort them in chronological order and finally display logs from different data points in single pane so that it can help user to find out root-cause of the problem. This utility provides the order of occurence of the logs from different data points and gives a graphical view in a single pane. This makes it easier to track the fault or error or the cause for application crash. System should perform the task efficiently ,hence for that dynamic loading of files is done and implemented as multi-threaded application.

V. Conclusion

In this paper, we proposed a framework for filtering logs according to specific analysis goals and diagnostic hypotheses set by the user or by an automated process. More specifically, the proposed framework provides simplified interface to user by displaying logs of specified location in formatted form (unique format),ordering them chronologically in single pane. For enhancing the performance we included features as dynamic loading of large log files and developing system as multi-threaded application. Above mentioned features of system helps user to better analyze the logs and find root cause of problem.

VI. Future Work

Initially tool is proposed for Linux based System. Further it can be extended as platform independent application so that can be used in heterogeneous environment. This tool can be developed for distributed system by synchronizing clocks of different systems so that logs from different points can be accessed and sorted in proper order. It can be further designed as Plug-in based application in which for parsing different log file, grammar files can be added as plug-in files. This will enhance the performance of the system as it will become lightweight.

References

[1] Towards structured log analysis Computer Science and Software Engineering(JCSSE).

[2] C. Yuan, N. Lao, J.-R. Wen, J. Li, Z. Zhang, Y.-M. Wang, and W.-Y. Ma. Automated known problem diagnosis with event traces In EuroSys 06: Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006 , pages 375388, New York, NY, USA, 2006.

[3] Suriadi,C. Ouyang,Root Cause Analysis with Enriched Process Logs.

[4] Hamzeh Zawawy,Kostas Kontogiannis,John Mylopoulos,Log Filtering and Interpretation for Root Cause Analysis.