NORTHWEST NAZARENE UNIVERSITY

Creating a Drive Log Tool for Analysis

THESIS
Submitted to the Department of Mathematics and Computer Science
in partial fulfillment of the requirements
for the degree of
BACHELOR OF SCIENCE

Elizabeth Catalina Bistriceanu
2021

THESIS
Submitted to the Department of Mathematics and Computer Science
in partial fulfillment of the requirements
for the degree of
BACHELOR OF SCIENCE

Elizabeth Catalina Bistriceanu
2021

Creating a Drive Log Tool for Analysis

Author: Elizabeth Bistriceanu
Elizabeth Catalina Bistriceanu

Approved: Kevin S McCarty
Kevin McCarty, Ph.D., Associate Professor
Department of Mathematics and Computer Science, Faculty Advisor

Approved: James D. Butkus, Jr.
Jim Butkus, MA, Assistant Director of Admissions
Department of Admissions, Second Reader

Approved: Barry Myers
Barry L. Myers, Ph.D., Chair,
Department of Mathematics & Computer Science

ABSTRACT

Creating a Drive Log Tool for Analysis.

BISTRICREANU, ELIZABETH (Department of Mathematics and Computer Science), MYERS, DR. BARRY (Department of Mathematics and Computer Science), MCCARTY, KEVIN (Department of Mathematics and Computer Science).

Large Enterprises like Hewlett Packard Enterprise currently have a very large amount of data logs that they receive every second. It is helpful to be able to analyze this log data at times that disk drive failures may occur. Although there is a lot of useful information that can help pin down a reason for failure, there is also a lot of unnecessary and repeated log data mixed in that is needed to be parsed through to find this helpful data. Currently, all this data is being found and shown to suppliers and customers of Hewlett Packard Enterprise by hand. Large enterprises often need to quickly parse and analyze log messages to determine whether specific hard drive errors have occurred to reduce malfunctions. To provide better analysis of log entries, we created a data analysis tool intended to allow for automated processing of drive logs. This tool parses and tokenizes log messages to allow for future analysis. Overall, this tool has greatly reduced employee error and work time. With just a few more steps, the tool will soon be completed to its full potential to help those at Hewlett Packard Enterprise analyze log data in a timely and efficient manner.

Table of Contents

LIST OF FIGURES

I. Introduction

At Hewlett Packard Enterprise, also known as HPE, disk drives generate billions of log files every day. To pinpoint failures and provide customers and suppliers reasons why a failure may have happened, these log files are searched through to find moments of errors that could correlate to the failure. “A log file is a computer-generated data file that contains information about usage patterns, activities, and operations within an operating system, application, server or another device (CIP, 2021).” Software and hardware engineers enjoy having log files because it is an easy way to debug their creations. Each individual device has its own set of log data, which allows software developers to be able to pinpoint an error or problem. As of now, HPE currently searches through these log files manually by looking for the log files of that error the date that it happened and pulling out any abnormal logs that may have been reported. Not only does this take hours to do, but it is also prone for errors since the likeliness of skipping over a line is high when it comes to looking at files with only the human eye. Therefore, HPE put together a project to create a tool that tracks down a log file that an employee would like to see and pull out any special information that would be important for HP’s supplier to view. This idea currently centers around disk drive failures only, however with a few tweaks and edits, this tool in the future could help with a large amount of log information. This tool is designed to make the lives of employees at HP much simpler once all the final ideas and components are pieced together.

II. Background

As previously mentioned, before the idea of this tool was introduced, employees were parsing through log data by hand. Whenever one of HP’s suppliers or customers had an issue with their disk drive, they would go to HP looking for some information as to why this was happening. To help aid whatever failure had happened, HP employees would gather the date and relative time (if possible) of the drive failure and look through thousands of lines of log files to locate information that is pertinent to the drive failure. A lot of this information is either random unneeded lines of code that can be thrown out or have sensitive information that cannot be released to the customer, such as IP addresses or part numbers. Because of this, HP employees need to be careful not to release any data that could be competitive to another company or just be useless. The very large set of log data could take an employee hours to search through as well. Having a tool to quickly search for specific logs and data would eliminate this time greatly and help reduce any errors. It would also be a much simpler process therefore, someone who doesn't know much about the log files themselves could do it, since you would need a good understanding of the log files to look through them by hand.

What is An ETL Tool?

For businesses that are primarily data driven, a tool is needed to help with the organization and transformation of data to help load it into a central repository. ETL stands for extract, transform and load. What an ETL tool does is take a large amount of data and convert it into something that multiple applications would be able to use and read (Naeem, 2021). See Figure 1 below to get more of a visual interpretation of an ETL tool.

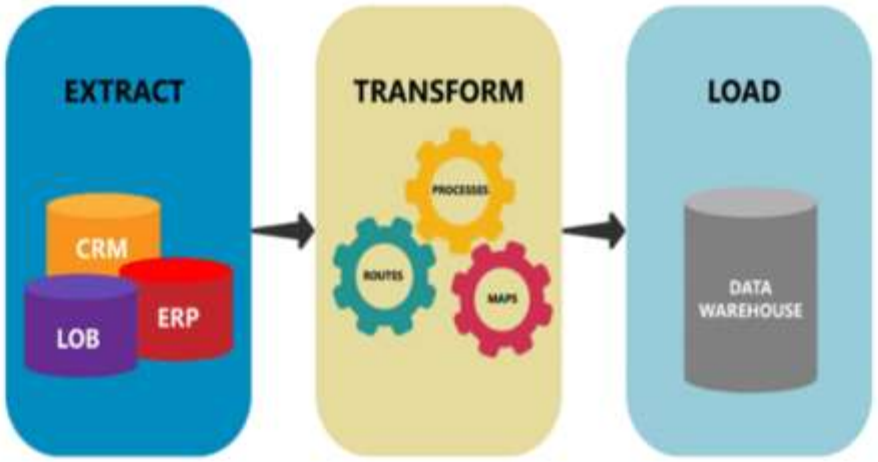


Figure 1 - ETL Tool Explanation

A few of the reasons why ETL tools are so great is because it helps with time-efficiency, handling long and complex data, reduces error probability and improve business intelligence (Naeem, 2021). Plenty of time and effort is saved when there is a tool to organize/transform data for you, and with an ETL tool, you can make sure that the data obtained for analysis is the finest quality possible which will help with decision making and business (See Appendix A).

Log Management Tool Example

Log management tools are applications that examine the data generated by a device or network. Log data is prevalent in many things, so it is no surprise to know that there are many different log management software applications. Log management plays a huge role in resource management, application troubleshooting, business analytics, and marketing insights (STH, 2021). In all these cases it is important to be able to pinpoint errors, alerts, or security issues. However, there needs to be a management system in place otherwise a large amount of an employee's time will be spent on digging through lines and lines of messages. Something like SolarWind's Log Analyzer will provide performs log aggregation, tagging, filtering, alerting (STH, 2021). To give some better meaning to these terms, log aggregation is the process of consolidating data to organize it in a better manner (Datadog, 2021) and tagging is including hashtag-style keywords into the log messages to help improve your ability to search and monitor log messages (Altvater, 2019). A software like SolarWind's Log Anaylizer includes a subscription starting at $1495, which sounds large, but it is important to compare this amount to the amount of time a company is paying an employee to search through log messages. It may prove to be beneficial to get into a subscription like this, however HPE has made the move to create their own tool. This in the end, will take much more time and trials then just buying a subscription; however, it will save the company money and the employee's time. It is important to remember that each minute, hour, or day that an employee is sorting through data is money that the company is paying them. So, creating at tool will save the company the money from buying almost a $1500 subscription and allow them to pay their employees to work on something more beneficial for the company.

III. The Project

Requirements

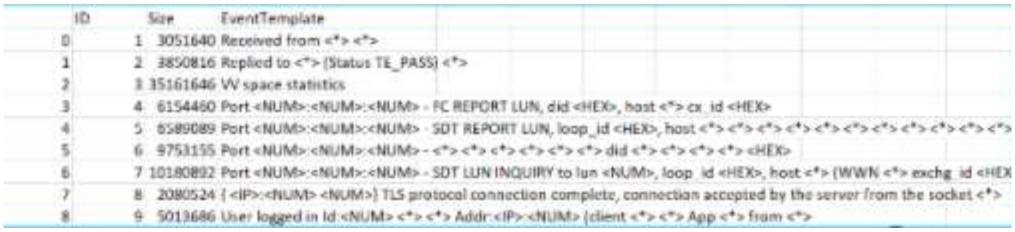
The purpose behind the log search analysis tool for Hewlett Packard Enterprise was to easily identify solutions as to why specific problems occurred in disk drives. See Figure 2 below for an image of an HPE disk drive that this tool could possibly help with (Amazon, 2021).

Figure 2 - Disk Drive

The time for the process of creating this tool was laid out over the span of two and a half months. Before even beginning the coding process, to get the best understanding possible of the tool and what it is used for, two training videos for Python and SQL were learned over the span of 2 weeks and a little bit about the log files themselves and how they are laid out. After getting a good understanding of the tool itself and the languages that would be used for it, our requirements as interns were to create a way to organize the large amount of data coming in, in more of a templatized way. This way, everything would be organized and fit nicely into a tool. The task of organizing the data was given over the course of three weeks. While an intern worked on organizing the data, another task was delegated out as well in the span of those three weeks to be able to locate and pinpoint any specific log files that may need to be analyzed. Since there are billions of files that are logged every day, this is a very important piece of the project that can be used more than just in the span of this tool. Once the organization and the templatization was complete for the next 3 weeks, the interns worked closely on gathering a file with over 3 million logs to throw through the templatizer created. The rest of the 3 weeks was spent trying to put all the pieces together and identify and problems that may need aid. A big issue was being able to store log messages in a hash table, however, the amount of time left was not enough to approach that problem, so the rest of the project was spent perfecting and organizing the created code/tool.

Implementation

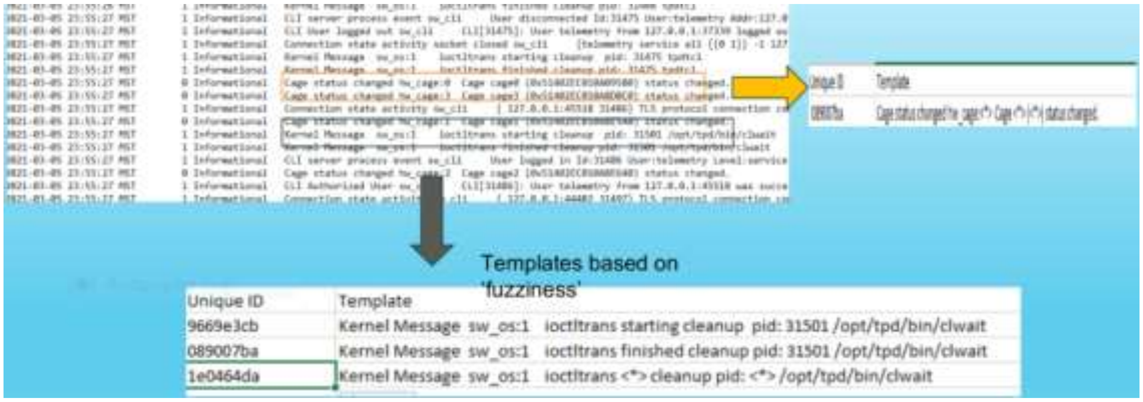
To begin the process of creating a log search tool, we first created a method that would easily search and pull-out specific files from the database where log files are stored in. Typically, when a supplier or customer has an error, they have a part number, serial number, date, etc. that they can give you to be able to investigate the error a little deeper. After the specified log files are located, this data is used to retrieve and download the needed files by searching the Amazon S3 Store. The Amazon S3 Store is a storage services that industries use to store and protect their data (Price Milburn, AWS), see Appendix B for more detailed information. The chosen downloaded files are downloaded to a destination of choice on a local disk in a single file. In this single file the lines of log information are organized by date, so any overlapping logs are deleted if they are duplicate lines. This combined log data file is then pushed through a drain3 algorithm tool. Drain3 is an algorithm that is used to identify log patterns (Ohana, 2020). This drain3 algorithm tool creates two different files. One of these files contains the unique messages pulled from the log files in a templatized manner, see Figure 3 below to view the organized messages.



	ID	Size	EventTemplate
0	1	3051640	Received from <*> <*>
1	2	3850816	Replied to <*> (Status TE_PASS) <*>
2	3	35161646	VV space statistics
3	4	6154460	Port <NUM>:<NUM>:<NUM> - FC REPORT LUN, did <HEX>, host <*> cx_id <HEX>
4	5	6589089	Port <NUM>:<NUM>:<NUM> - SDT REPORT LUN, loop_id <HEX>, host <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*>
5	6	9753155	Port <NUM>:<NUM>:<NUM> - <*> <*> <*> <*> <*> <*> did <*> <*> <*> <*> <HEX>
6	7	10180892	Port <NUM>:<NUM>:<NUM> - SDT LUN INQUIRY to lun <NUM>, loop_id <HEX>, host <*> (WWN <*> exchg_id <HEX
7	8	2080524	(<IP>:<NUM> <NUM>) TLS protocol connection complete, connection accepted by the server from the socket <*>
8	9	5013686	User logged in Id:<NUM> <*> <*> Addr:<IP>:<NUM> (client <*> <*> App <*> from <*>

Figure 3 - Organized Log Messages

For our project, we create a log file with over two million lines of log messages that were condensed to about 420 unique messages. A unique ID is given to each of these unique log messages to help aid for future use and any overlapping numbers between unique messages is tokenized, meaning they are replaced with characters such as # or *, shown in Figure 4 below where the yellow arrow is pointing.



```
2021-03-05 23:55:26 MST    1 Informational  Kernel Message  sw_os:1    ioctltrans finished cleanup pid: 31466 tpdtcl
2021-03-05 23:55:27 MST    1 Informational  CLI server process event sw_cli     User disconnected Id:31475 User:telemetry Addr:127.0
2021-03-05 23:55:27 MST    1 Informational  CLI User logged out sw_cli     CLI[31475]: User telemetry from 127.0.0.1:37330 logged ou
2021-03-05 23:55:27 MST    1 Informational  Connection state activity socket closed sw_cli     [telemetry service all ([0 1]) -1 127
2021-03-05 23:55:27 MST    1 Informational  Kernel Message  sw_os:1    ioctltrans starting cleanup  pid: 31475 tpdtcl
2021-03-05 23:55:27 MST    1 Informational  Kernel Message  sw_os:1    ioctltrans finished cleanup pid: 31475 tpdtcl
2021-03-05 23:55:27 MST    0 Informational  Cage status changed hw_cage:0  Cage cage0 ([illegible]) status changed.
2021-03-05 23:55:27 MST    0 Informational  Cage status changed hw_cage:3  Cage cage3 ([illegible]) status changed.
2021-03-05 23:55:27 MST    1 Informational  Connection state activity sw_cli     ( 127.0.0.1:45318 31486) TLS protocol connection co
2021-03-05 23:55:27 MST    0 Informational  Cage status changed hw_cage:1  Cage cage1 ([illegible]) status changed.
2021-03-05 23:55:27 MST    1 Informational  Kernel Message  sw_os:1    ioctltrans starting cleanup  pid: 31501 /opt/tpd/bin/clwait
2021-03-05 23:55:27 MST    1 Informational  Kernel Message  sw_os:1    ioctltrans finished cleanup pid: 31501 /opt/tpd/bin/clwait
2021-03-05 23:55:27 MST    1 Informational  CLI server process event sw_cli     User logged in Id:31486 User:telemetry Level:service
2021-03-05 23:55:27 MST    0 Informational  Cage status changed hw_cage:2  Cage cage2 ([illegible]) status changed.
2021-03-05 23:55:27 MST    1 Informational  CLI Authorized User sw_cli     CLI[31486]: User telemetry from 127.0.0.1:45318 was succe
2021-03-05 23:55:27 MST    1 Informational  Connection state activity sw_cli     ( 127.0.0.1:44482 31497) TLS protocol connection co
```

Unique ID	Template
[illegible]	Cage status changed hw_cage:<*> Cage <*> (<*>) status changed.

Templates based on 'fuzziness'

Unique ID	Template
9669e3cb	Kernel Message sw_os:1 ioctltrans starting cleanup pid: 31501 /opt/tpd/bin/clwait
089007ba	Kernel Message sw_os:1 ioctltrans finished cleanup pid: 31501 /opt/tpd/bin/clwait
1e0464da	Kernel Message sw_os:1 ioctltrans <*> cleanup pid: <*> /opt/tpd/bin/clwait

Figure 4 - Log Message Templatization

The second file contains all the log messages that were pushed through, but each with their unique ID number that correlates with the first file created, Figure 3 shows an example of these IDs on their templatized log message. Another method was then created to produce CSV files that will be given to suppliers/customers or kept internally based on four categories: 0 meant that the log message can be deleted completely, 1 meant that the unedited log data message can be kept in the supplier’s csv file and kept internally, 2 meant that there was content in the message that needs to be edited before put into the supplier’s csv file but the original can be kept internally and 3 meant that the data message will only be kept internally and excluded from the supplier.

IV. Results

By the end of this project, a tool to track down specific log data messages was created. This specific part of the project allowed an employee to download a specified log file to their local disk on their computer. The drain3 algorithm that was implemented also significantly condensed the large log files and was able to pull out only unique messages, making them easier to go through and read. With the created tool, we are then able to easily swap out any critical information like IP address and part numbers with a different token, so other companies and suppliers will not be able to see these things. These three pieces of the project are very helpful in paving the path to an easier way to sort through HPE’s log data. There is, however, a few issues that were ran into towards the end of the project. The drain3 algorithm did not implement a function that gives each unique log message a uniquely stored ID. Each time you ran the algorithm, a new ID was given to each message. Therefore, the tool cannot be efficiently used because there is no way to store and look back at specific log messages. Because of this, we are unable to give the supplier and customers their data in this way just yet. This is a simple fix though and if it was something we did not forget during the process, we most likely could have fit it into our schedule. However, once we started approaching the deadline it was a crucial piece of the project, we realized that we did not have time to focus on.

V. Future Work

For future work and to improve the log message ID issue, we believe adding a HashMap with the drain3 algorithm will address the log message ID issue. A HashMap (or hash table) is a way to organize data so that values can be looked up with a given key (Cake Labs, Inc.). Once messages are starting to be stored in this created hash map, a large library of keys can be created for all the unique messages that log files have. When those log messages are stored, we can hardcode them into the method that creates CSV files for the suppliers and customers. This tool has also only been used on Event Log Data. So, for future use, the format of the logs would need to be taken into consideration if someone would like to use this tool on something else. However, this is quite a simple thing to do and will eliminate a lot of time on other issues, not just disk drive failures. This tool has the potential for helping anybody find information much more efficiently. However, it just needs to be updated a bit more to recognize the many other formats and information that is out there.

References

Altvater, A. (2019, May 6). *Log tagging creates smarter application logs #awesomelogs*. Stackify. Retrieved November 15, 2021, from https://stackify.com/get-smarter-log-management-with-log-tags/.

Amazon. (n.d.). *Amazon.com: 1TB Sata internal laptop hard drive/HDD for ...* Amazon. Retrieved November 15, 2021, from https://www.amazon.com/Internal-Laptop-Drive-Inspiron-E1405/dp/B00TOMPC26.

Cake Labs, Inc. (n.d.). *Hash table/hash map data structure: Interview cake*. Interview Cake: Programming Interview Questions and Tips. Retrieved November 15, 2021, from https://www.interviewcake.com/concept/java/hash-map.

Continuous Intelligence Platform . (2021). *What is a log file?* Sumo Logic. Retrieved November 15, 2021, from https://www.sumologic.com/glossary/log-file/.

Datadog. (2021, August 3). *Datadog*. Log Aggregation: What It Is & How It Works. Retrieved November 15, 2021, from https://www.datadoghq.com/knowledge-center/log-aggregation/#:~:text=Log%20aggregation%20is%20the%20process,to%20facilitate%20streamlined%20log%20analysis.

Naeem, T. (2021, November 10). *What is an ETL tool: Definition, uses, and use-cases* . Astera. Retrieved November 15, 2021, from https://www.astera.com/type/blog/what-is-etl-tool/.

Ohana, D. (2020, May 13). *Use open source drain3 log-template mining project to monitor* *for network outages*. IBM Developer. Retrieved November 15, 2021, from https://developer.ibm.com/blogs/how-mining-log-templates-can-help-ai-ops-in-cloud-scale-data-centers/.

Price Milburn. (1980). *AWS*. Amazon. Retrieved November 15, 2021, from https://aws.amazon.com/pm/serv-s3/?trk=ps_a134p000004f2aOAAQ&trkCampaign=acq_paid_search_brand&sc_channel=PS&sc_campaign=acquisition_US&sc_publisher=Google&sc_category=Storage&sc_country=US&sc_geo=NAMER&sc_outcome=acq&sc_detail=amazon+s3&sc_content=S3_e&sc_matchtype=e&sc_segment=488982706722&sc_medium=ACQ-P%7CPS-GO%7CBrand%7CDesktop%7CSU%7CStorage%7CS3%7CUS%7CEN%7CText&s_kwcid=AL%214422%213%21488982706722%21e%21%21g%21%21amazon+s3&ef_id=Cj0KCQjwtrSLBhCLARIsACh6RmhDj0UHVufL3rG5K_WreDt-xSevlH1h5sf9L9ItKGIRhL2O-Y7jLrsaAkt1EALw_wcB%3AG%3As.

Software Testing Help. (2021, November 1). *Top 8 best log management software: Log Analysis Tool Review 2021*. Software Testing Help. Retrieved November 15, 2021, from https://www.softwaretestinghelp.com/log-management-software/.

Appendix A

ETL Tools

An ETL tool (an extract, transport and load tool) transforms data into a format that is both satisfying and operational for the requirements of a particular business. An ETL tool that is well designed is going to enforce quality standards, conform and deliver data so that it is ready to use by developers and readily available for the use of applications.

Why Use an ETL Tool

Being able to consolidate data is something that can be quite time consuming, especially with large amounts of data that most businesses find themselves dealing with. An ETL tool's most important and beneficial use is the ability to cut down and save time. It consolidates data in an automated way and as a result, plenty of time and effort is saved – instead of someone spending their own time importing data manually. ETL tools also help format large amounts of data that may be coming into businesses from all over the world. Things from multiple countries with distinct addresses, part IDs, etc. are easily dealt with an ETL tool. Along with that, error probability is greatly reduced since the likeliness of causing errors manually is high, especially with large amounts of data.

Types of ETL Tools

ETL tools can be categorized into a few main types of tools. Batch ETL tools are batch processing that is used to acquire data from the source systems. Whatever data that is being dealt with, is extracted, transformed, and loaded into the repository in the batches of ETL jobs. This method is cost-effective because in a time-bounded way it uses limited resources. Another method is a real-time ETL tool, which is data that is extracted, cleansed, enriched and loaded to the target in real-time. This tool is helpful cause it offers faster access to information. Because of this, these ETL tools are quite popular with businesses. On-Premise ETL Tools have the data and the repository configured on-premise. This method is used mainly because of data security, so that the tool can be deployed on site. The last method is a Cloud ETL Tool which are tools deployed on the cloud as various cloud-based applications. This method helps with flexibility in the ETL process cause it helps with data transfer and management.

Appendix B

Amazon S3 Store

The Amazon S3 Store is an object storage service that allows businesses and industries to have a better amount of data availability with both security and enhanced performance. Any customer of any size can use the S3 Store to store and protect their data. Things like data lakes, websites, cloud-native applications, backups, and many more are just a few of the things that can be stored in Amazon’s S3 Store. Millions of customers with unique requirements and amount of data can integrate and use the S3 Store because of the focus on customer’s needs.

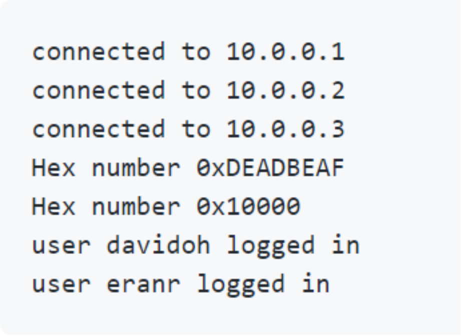
Uses

Things like building data lakes to run data analytics for artificial intelligence and machine learning can be run on the S3 Store. The possibility of running cloud-native applications and having backup/restoration for critical data are also amazing perks to the store. On top of all these great perks is also the low cost compared to other data storages.

Appendix C

Drain3 Algorithm

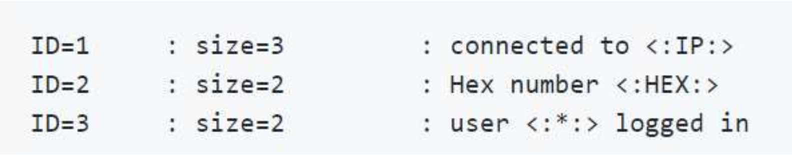
Drain3 is an open-source streaming log template miner. From a stream of log messages, Drain3 will extract templates in an efficient time. How it works, is by employing a parse tree that has a fixed depth to guide the group search process along, which avoids constructing an unbalanced tree. What is so great about the algorithm is that it really adapts and learns as it goes and automatically extracts log templates from raw log entries. See Figure 5 below of the raw log input given to the algorithm.



```
connected to 10.0.0.1
connected to 10.0.0.2
connected to 10.0.0.3
Hex number 0xDEADBEAF
Hex number 0x10000
user davidoh logged in
user eranr logged in
```

Figure 5 - Raw Input

After taking in the raw input shown above, Drain3 then extracts the following templates shown in Figure 6 below.



```
ID=1     : size=3         : connected to <:IP:>
ID=2     : size=2         : Hex number <:HEX:>
ID=3     : size=2         : user <:*:> logged in
```

Figure 6 - Drain3 Templates