LOG FILE ANALYZER

Project report submitted in partial fulfillment of the requirement for the Internship

By

Manish (768940)

Vikas (768997)

Under the supervision of

Dr. Pankaj Anadure

To

Cognizant Technology Solutions India Pvt Ltd , Manyata Tech Park, Nagawara, Bengaluru, Karnataka 560045

CERTIFICATE

Candidate's Declaration

We hereby declare that the work presented in this report entitled **"**Log File Analyzer**"** in partial fulfillment of the requirements for the award of the internship is an authentic record of my own work carried out over a period from Feb 2019 to April 2019 under the supervision of Mr. Pankaj Anadure**.**

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Manish (768940) Vikas (768997)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Mr. Pankaj Anadure

Dated :

ACKNOWLEDGEMENT

Learning through the project under the guidance of our esteemed mentor Mr. Pankaj Anadure, whose expertise knowledge in the domain of Frontend and Backend Development, not only cleared all our ambiguities but also generated a high level of interest and gusto in the subject. We are truly grateful for his guidance and support throughout the project. We would also like to thank our SME for allocation of the project as well as its resources.

The prospect of working in a group with high level of accountability fostered a spirit of teamwork and created a feeling of oneness which thus, expanded our ken, motivated us to perform to the level best of our ability and create a report of the highest quality.

To do the best quality work, with utmost sincerity and precision has been our constant endeavor.

Date:

Manish (768940)

Vikas (768997)

TABLE OF CONTENT

LIST OF FIGURES

ABSTRACT

Log analysis is the process of turning your log files into data and then making intelligent decisions based on that data. In software testing, the analysis of logs is designed to check and monitor the application work. Also logs help to fix some errors. Logs are the text lines which contains systematic information about service work and actions such as: IP-address, dates, time, and viewed sites, potential buyers domains, status code, and the size of the answer, the general info about query, loading time, user-agent, etc. of particular operations that occur in different environments - application itself or the system that runs an app. Logs include the messages of several types: informational, warning and error.

Logs help to specify who uses an application and how often it is used. Moreover, testers can define whether the session was successful, and to detect possible mismatches or mistakes a users of the app can face. The log analysis helps to design new scenarios for further web or mobile testing.

Chapter – 1

INTRODUCTION

In computer log management, log analysis (or system and network log analysis) is an art and science seeking to make sense out of computer-generated records (also called log or audit trail records). The process of creating such records is called data logging.

Logs are emitted by network devices, operating systems, applications and all manner of intelligent or programmable device. A stream of messages in time-sequence often comprise a log. Logs may be directed to files and stored on disk, or directed as a network stream to a log collector.

Log messages must usually be interpreted with respect to the internal state of its source (e.g., application) and announce security-relevant or operations-relevant events (e.g., a user login, or a systems error).

Logs are often created by software developers to aid in the debugging of the operation of an application or understanding how users are interacting with a system, such as search engine. The syntax and semantics of data within log messages are usually application or vendor-specific. Terminology may also vary; for example, the authentication of a user to an application may be described as a login, a logon, a user connection or authentication event. Hence, log analysis must interpret messages within the context of an application, vendor, system or configuration in order to make useful comparisons to messages from different log sources.

Log message format or content may not always be fully documented. A task of the log analyst is to induce the system to emit the full range of messages in order to understand the complete domain from which the messages must be interpreted.

1.1 PROBLEM STATEMENT

This is the project has its motivation from the log files which are cumbersome to read. Log Files contains huge amount of useful information about which must be taken care. This project

is created for debugging of the operation of the application or understanding how users are interacting with a system. The overall goal of this research is to invent and design a good model of generic processing of log files.

The following list sums up areas involved by this research :

- formal definition of a log file
- formal description of the structure and syntax of a log file (metadata)
- formal specification of a programming language for easy and efficient log analysis
- design of a basic library/API and functions or operators for easy handling of logs within the programming language
- deployment of data mining/warehousing techniques if applicable
- design of user interface The expected results are both theoretical both practical.

1.2 OBJECTIVE

The essential goal of the venture is:

a. To design a framework
b. To extract the raw data and convert that data into useful or readable form.
c. To analyze the data and perform certain actions like searching, sorting, etc.
d. To visualize different parameters of the data to get some results.

1.3 METHODOLOGY

A diagram of our strategy is given in Figure 1. We first preprocess the raw data into some useful information. Utilizing this preprocessed information, we perform certain actions on that information like searching particular rows across the data, perform sorting among different parameters, also try to visualize the effect of different parameters.

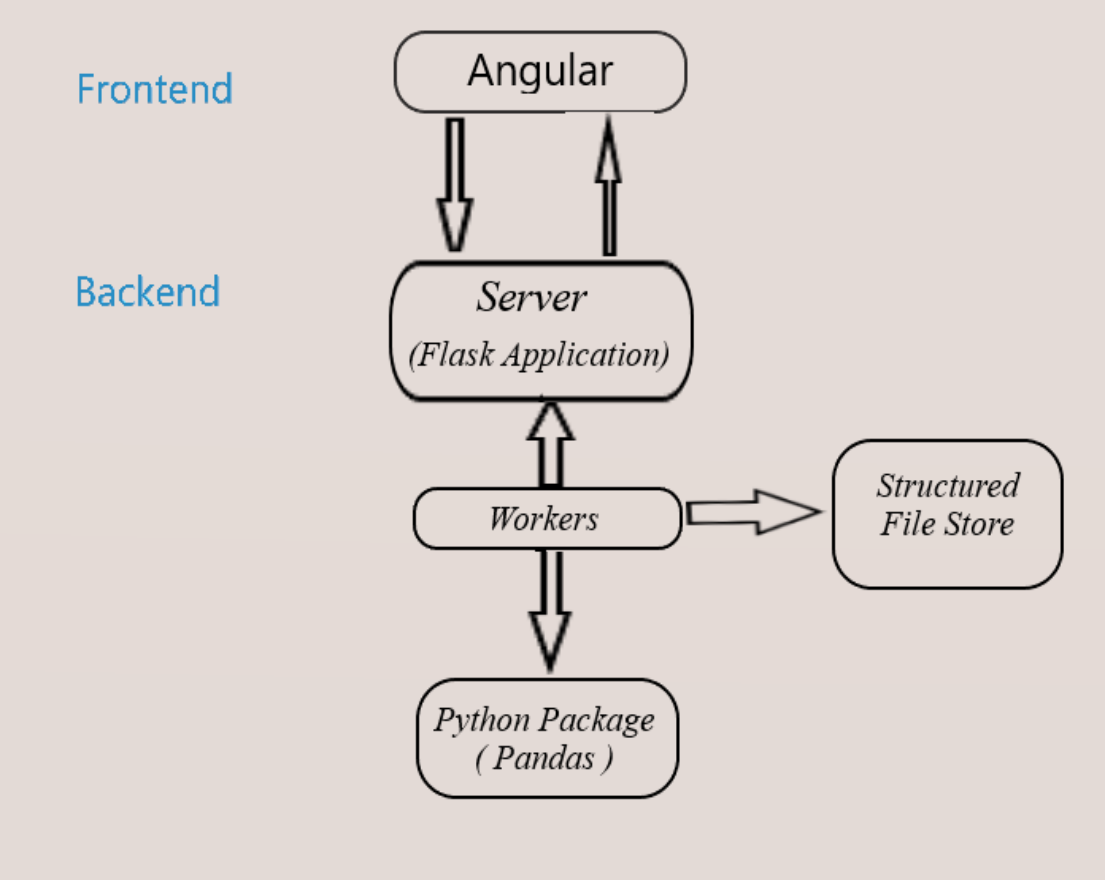


Figure 1: Overview of the methodology

Chapter – 2

LITERATURE SURVEY

There are various applications (known as log file analyzers or log files visualization tools) that can digest a log file of specific vendor or structure and produce easily human readable summary reports. Such tools are undoubtedly useful, but their usage is limited only to log files of certain structure. Although such products have configuration options, they can answer only built-in questions and create built-in reports. The initial motivation of this work was the lack of Cisco NetFlow analyzer that could be used to monitor and analyze large computer networks like the metropolitan area network of the University of West Bohemia (WEBNET) or the country-wide backbone of the Czech Academic Network (CESNET) using Cisco NetFlow data exports. Because the amount of log data (every packet is logged!), evolution of the NetFlow log format in time and wide spectrum 4 of monitoring goals/questions, it seems that introduction of an new, systematic, efficient and open approach to the log analysis is necessary. There is also a belief that it is useful to research in the field of log files analysis and to design an open, very flexible modular tool, that would be capable to analyze almost any log file and answer any questions, including very complex ones. Such analyzer should be programmable, extendable, efficient (because of the volume of log files) and easy to use for end users. It should not be limited to analyze just log files of specific structure or type and also the type of question should not be restricted.

2.1 Various others online Log File Analyzer :

Log MX

It’s an intuitive tool for analyzing log files. Log MX is not just reading log files, it parses log events from any file or data stream, in order to display a structured view of your logs. No need to install a web server or update your logs producers, Log MX is a standalone application weighing only about 6 MB.

The disadvantage of this analyzer you can’t analyze your log files for free you have to make an account and buy this tool.

G suite

It's another online log file analyzer tool. Purpose is same but the disadvantage is it can analyze limited no of log files. It's domain to analyze various log files is very less. You can analyze only certain no of log files such as Chrome OS, GSSMO, GSMME, GSMMO, GCDS, GSPS.

2.2 What Can We Get from Log Files

This paragraph summarizes application of log files in software development, testing and monitoring. The desired useful information that resides in log files can be divided into several classes:

- Generic statistics (peak and average values, median, modus, deviations..)
  Goal: finding and processing of set of report elements
  Useful for: setting hardware requirements, accounting

- Program/system warnings (power failure, low memory)
  Goal: finding all occurrences of reports
  Useful for: system maintenance

- Security related warnings
  Goal: finding all occurrences of reports in report trace

- Validation of program runs
  Goal: construction and examination of state machine
  Useful for: software testing

- Time related characteristics
  Goal: computing or guessing time periods between multiple occurrences of report
  Useful for: software profiling & benchmarking

- Causality and trends

  Goal: finding a set of reports that often (or every time) preceed (or follow) within a given time period (or withing a given count of reports)

  Useful for: data mining

- Behavioral patterns

  Goal: using a set of reports as the set of transitions of Markov process

  Useful for: determining performance and reliability

2.4 Text Processing Languages

There are several languages designed for easy text processing. Their operation is based on regular expressions that makes their usage surprisingly efficient. The well-known representatives are AWK and PYTHON. Log analysis (or at least simple log analysis) is in fact a text processing task, so the need to examine such languages is obvious. We will concentrate on PYTHON, because its application is in text processing. The basic function of awk is to search files for lines (or other units of text) that contain certain patterns. When a line matches one of the patterns, awk performs specified actions on that line. awk keeps processing input lines in this way until the end of the input files are reached. Programs in awk are different from programs in most other languages, because awk programs are data-driven; that is, you describe the data you wish to work with, and then what to do when you find it. Most other languages are procedural; you have to describe, in great detail, every step the program is to take. When working with procedural languages, it is usually much harder to clearly describe the data your program will process. For this reason, awk programs are often refreshingly easy to both write and read. When you run awk, you specify an awk program that tells awk what to do. The program consists of a series of rules. (It may also contain function definitions, an advanced feature which we will ignore for now. See section User-defined Functions.) Each rule specifies one pattern to search for, and one action to perform when that pattern is found.

2.5 Sample Questions

Although web server logs are not the primary objective of this thesis, this section provides a

brief overview of typical questions concerning web usage.

• Who is visiting your site. The fundamental point is who are the readers of your server, who are they like, from what countries, institutions etc. Many servers identify returning visitors and offers personal or personalized pages for frequent visitors.

• The path visitors take through your pages. Where do visitors start reading your server, where they come from, how easily they navigate, do they often fail to find the right hyperlink and thus often return... In brief, the objectives are trends, paths, and patterns.

• How much time visitors spend on each page. From the answer you can deduce what pages are interesting and what are very confusing or dull.

• Where visitors leave your site. The last page a visitor viewed can be a logical place where end the visit or it can be a place where visitor bailed out.

• The success of users experiences at your site. Purchases transacted, downloads completed, information viewed, tasks accomplished. Such indicators can show whether the site it well made, organized and maintained, whether the offered products are well presented.

2.6 **Chapter Summary**

Log file analysis has many applications, such as site security. In terms of search engine optimization, the process usually involves downloading the file from your server and importing it into a log file analysis tool, where all the information about every “hit” on the site (whether bot or human) can be analyzed to inform SEO decisions and learn about previously unknown issues.

Log file analysis is an arduous process that frequently results in the discovery of critical technical problems that could be found no other way. Log files contain incredibly accurate data that allow a brand to better understand how search engines are crawling their site and the kind of information they are finding.

Log file data includes a record of the URL/resource that was requested, action taken, time and date, IP of the machine it originated from, user agent/browser type, and other pieces of information.

Chapter – 3

SYSTEM REQUIREMENTS

3.1 TOOLS & FRAMEWORKS

3.1.1 Visual Studio Code

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences.

Visual Studio Code is based on Electron, a framework which is used to deploy Node.js applications for the desktop running on the Blink layout engine.

Visual Studio Code is a source code editor that can be used with a variety of programming languages. Instead of a project system it allows users to open one or more directories, which can then be saved in workspaces for future reuse.

It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many of Visual Studio Code features are not exposed through menus or the user interface, but can be accessed via the command palette.

3.1.2 ANGULAR

Angular (commonly referred to as "Angular 2+" or "Angular v2 and above") is a TypeScript-based open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.

Angular is a platform that makes it easy to build applications with the web. Angular combines declarative templates, dependency injection, end to end tooling, and integrated best practices to solve development challenges. Angular empowers developers to build applications that live

on the web, mobile, or the desktop.

Why Angular ?

- Angular presents you not only the tools but also design patterns to build your project in a maintainable way. When an Angular application is crafted properly, you don't end up with a tangle of classes and methods that are hard to modify and even harder to test. The code is structured conveniently and you won't need to spend much time in order to understand what is going on.
- It's JavaScript, but better. Angular is built with TypeScript, which in turn relies on JS ES6. You don't need to learn a totally new language, but you still receive features like static typing, interfaces, classes, namespaces, decorators etc.
- No need to reinvent the bicycle. With Angular, you already have lots of tools to start crafting the application right away. You have directives to give HTML elements dynamic behavior. You can power up the forms using Form Control and introduce various validation rules. You may easily send asynchronous HTTP requests of various types. You can set up routing with little hassle. And there are many more goodies that Angular can offer us!
- Components are decoupled. Angular strived to remove tight coupling between various components of the application. Injection happens in NodeJS-style and you may replace various components with ease.
- All DOM manipulation happens where it should happen. With Angular, you don't tightly couple presentation and the application's logic making your markup much cleaner and simpler.
- Testing is at the heart. Angular is meant to be thoroughly tested and it supports both unit and end-to-end testing with tools like Jasmine and Protractor.
- Angular is mobile and desktop-ready, meaning you have one framework for multiple platforms.
- Angular is actively maintained and has a large community and ecosystem. You can find lots of materials on this framework as well as many useful third-party tools.

3.1.2.1 Modules:

Angular *NgModules* differ from and complement JavaScript (ES2015) modules. An NgModule declares a compilation context for a set of components that is dedicated to an

application domain, a workflow, or a closely related set of capabilities. An NgModule can associate its components with related code, such as services, to form functional units.

Every Angular app has a *root module*, conventionally named AppModule, which provides the bootstrap mechanism that launches the application. An app typically contains many functional modules.

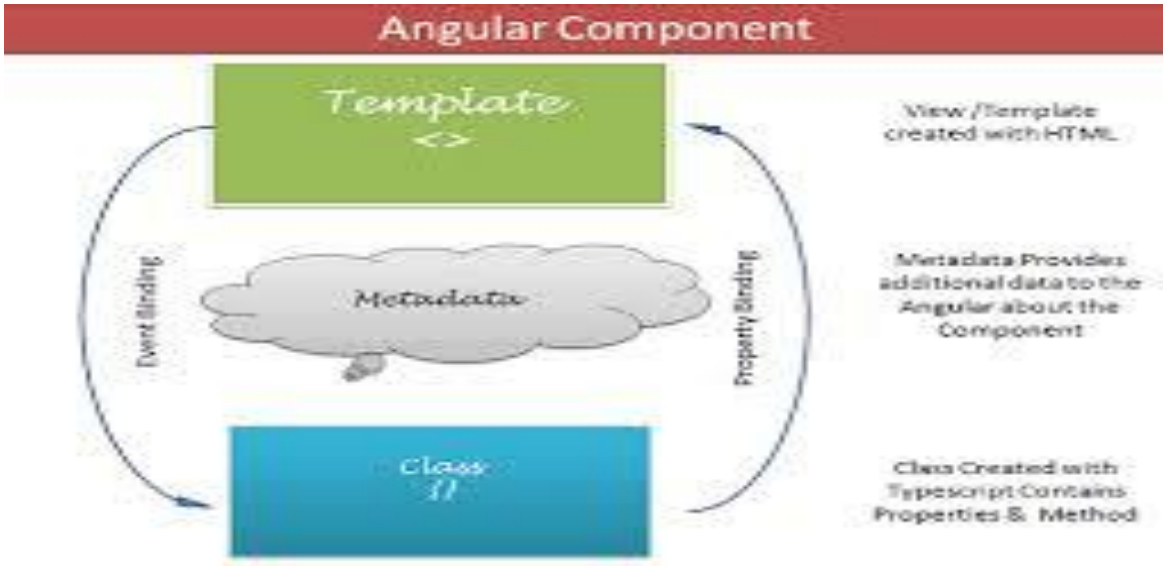
Like JavaScript modules, NgModules can import functionality from other NgModules, and allow their own functionality to be exported and used by other NgModules. For example, to use the router service in your app, you import the Router NgModule.

Organizing your code into distinct functional modules helps in managing development of complex applications, and in designing for reusability. In addition, this technique lets you take advantage of *lazy-loading*—that is, loading modules on demand—to minimize the amount of code that needs to be loaded at startup.

3.1.2.2 Components

Every Angular application has at least one component, the *root component* that connects a component hierarchy with the page document object model (DOM). Each component defines a class that contains application data and logic, and is associated with an HTML *template* that defines a view to be displayed in a target environment.

The @Component() decorator identifies the class immediately below it as a component, and provides the template and related component-specific metadata.



3.1.2.3 Templates, directives, and data binding:

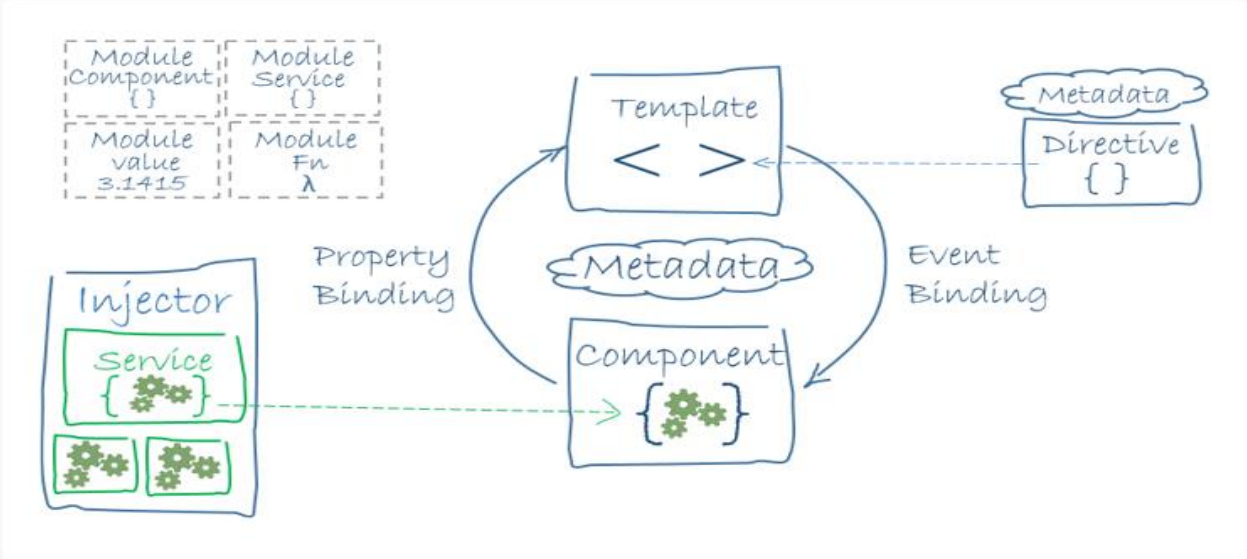
A template combines HTML with Angular markup that can modify HTML elements before they are displayed. Template *directives* provide program logic, and *binding markup* connects your application data and the DOM. There are two types of data binding:

Event binding lets your app respond to user input in the target environment by updating your application data.

Property binding lets you interpolate values that are computed from your application data into the HTML.

Before a view is displayed, Angular evaluates the directives and resolves the binding syntax in the template to modify the HTML elements and the DOM, according to your program data and logic. Angular supports *two-way data binding*, meaning that changes in the DOM, such as user choices, are also reflected in your program data.

Your templates can use *pipes* to improve the user experience by transforming values for display. For example, use pipes to display dates and currency values that are appropriate for a user's locale. Angular provides predefined pipes for common transformations, and you can also define your own pipes.



3.1.2.4 Services and dependency injection:

For data or logic that isn't associated with a specific view, and that you want to share across components, you create a *service* class. A service class definition is immediately preceded by the @Injectable() decorator. The decorator provides the metadata that allows other providers to be injected as dependencies into your class.

Dependency injection (DI) lets you keep your component classes lean and efficient. They don't fetch data from the server, validate user input, or log directly to the console; they delegate such tasks to services.

3.1.2.5 Routing:

The Angular Router NgModule provides a service that lets you define a navigation path among the different application states and view hierarchies in your app. It is modeled on the familiar browser navigation conventions:

Enter a URL in the address bar and the browser navigates to a corresponding page.

Click links on the page and the browser navigates to a new page.

Click the browser's back and forward buttons and the browser navigates backward and forward through the history of pages one has seen.

The router maps URL-like paths to views instead of pages. When a user performs an action, such as clicking a link, that would load a new page in the browser, the router

intercepts the browser's behavior, and shows or hides view hierarchies.

If the router determines that the current application state requires particular functionality, and the module that defines it hasn't been loaded, the router can *lazy-load* the module on demand.

The router interprets a link URL according to your app's view navigation rules and data state. You can navigate to new views when the user clicks a button or selects from a drop box, or in response to some other stimulus from any source. The router logs activity in the browser's history, so the back and forward buttons work as well.

To define navigation rules, you associate *navigation paths* with your components. A path uses a URL-like syntax that integrates your program data, in much the same way that template syntax integrates your views with your program data. You can then apply program logic to choose which views to show or to hide, in response to user input and your own access rules.

3.1.2.6 LATEST VERSIONS

- **Version 2:** Angular 2.0 was announced at the ng-Europe conference 22-23. October 2014. The drastic changes in the 2.0 version created considerable controversy among developers. The final version was released on September 14, 2016.
- **Version 4:** On 13 December 2016 Angular 4 was announced, skipping 3 to avoid a confusion due to the misalignment of the router package's version which was already distributed as v3.3.0. The final version was released on March 23, 2017. Angular 4 is backward compatible with Angular 2.
- **Version 7:** Angular 7 was released on October 18, 2018. Updates regarding Application Performance, Angular Material & CDK, Virtual Scrolling, Improved Accessibility of Selects, now supports Content Projection using web standard for custom elements, and dependency updates regarding Typescript 3.1, RxJS 6.3, Node 10 (still supporting Node 8).

3.1.3 TYPESCRIPT

TypeScript is an open-source programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript, and adds optional static typing to the language.

TypeScript is designed for development of large applications and trans compiles to JavaScript. As TypeScript is a superset of JavaScript, existing JavaScript programs are also valid TypeScript programs.

TypeScript lets you write JavaScript the way you really want to. TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. TypeScript is pure object oriented with classes, interfaces and statically typed like C# or Java. The popular JavaScript framework Angular 2.0 is written in TypeScript. Mastering TypeScript can help programmers to write object-oriented programs and have them compiled to JavaScript, both on client-side and server-side (Node.js) execution.

3.1.4 PYTHON

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales Van Rossum led the language community until stepping down as leader in July 2018.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object oriented, imperative, functional and procedural. It also has a comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

3.1.4.1 ADVANTAGES

Python has top the charts in the recent years over other programming languages like C, C++ and Java and is widely used by the programmers. The language has undergone a drastic change since its release 25 years ago as many add-on features are introduced. The Python 1.0 had the module system of Modula-3 and interacted with Amoeba Operating System with varied functioning tools. Python 2.0 introduced in the year 2000 had features of garbage collector and Unicode Support. Python 3.0 introduced in the year 2008 had a constructive design that avoids duplicate modules and constructs. With the added features, now the companies are using Python 3.5.

The programmers of big companies use Python as it has created a mark for itself in the software development with characteristic features like-

- Interactive
- Interpreted
- Modular
- Dynamic
- Object-oriented
- Portable
- High level
- Extensible in C++ & C

3.1.4.2 BENEFITS

- **Extensive support libraries:** It provides large standard libraries that include the areas like string operations, Internet, web service tools, operating system interfaces and protocols. Most of the highly used programming tasks are already scripted into it that limits the length of the codes to be written in Python.
- **Integration Feature**: Python integrates the Enterprise Application Integration that makes it easy to develop Web services by invoking COM or COBRA components. It has powerful control capabilities as it calls directly through C, C++ or Java via Jython. Python also processes XML and other markup languages as it can run on all modern operating systems through same byte code.

- **Improved Programmers Productivity:** The language has extensive support libraries and clean object-oriented designs that increase two to tenfold of programmer's productivity while using the languages like Java, VB, Perl, C, C++ and C#.
- **Productivity:** With its strong process integration features, unit testing framework and enhanced control capabilities contribute towards the increased speed for most applications and productivity of applications. It is a great option for building scalable multi-protocol network applications.

3.1.4.3 PYTHON 2 AND PYTHON 3

Python 3 was introduced in 2008 with the aim of making Python easier to use and change the way it handles strings to match the demands placed on the language today. Programmers who first learned to program in Python 2 sometimes find the new changes difficult to adjust to, but newcomers often find that the new version of the language makes more sense.

Python 3.0 is fundamentally different to previous Python releases because it is the first Python release that is not compatible with older versions. Programmers usually don't need to worry about minor updates (e.g. from 2.6 to 2.7) as they usually only change the internal workings of Python and don't require programmers to change their syntax. The change between Python 2.7 (the final version of Python 2) and Python 3.0 is much more significant — code that worked in Python 2.7 may need to be written in a different way to work in Python 3.0.

3.1.4.4 KEY DIFFERENCES

- **Print:** In Python 2, "print" is treated as a statement rather than a function. In contrast, Python 3 explicitly treats "print" as a function, which means you have to pass the items you need to print to the function in parentheses in the standard way, or you will get a syntax error.

- **Integer Division:** Python 2 treats numbers that you type without any digits after the decimal point as integers, which can lead to some unexpected results during division.

This is because Python 2 assumes that you want the result of your division to be an integer, so it rounds the calculation down to the nearest whole number. Python 3 evaluates 3 / 2 as 1.5 by default, which is more intuitive for new programmers.

- **List Comprehension Loop Variables:** In previous versions of Python, giving the variable that is iterated over in a list comprehension the same name as a global variable could lead to the value of the global variable being changed. This bug has been fixed in Python 3, so you can use a variable name you already used for the control variable in your list comprehension without worrying about it leaking out and messing with the values of the variables in the rest of your code.

- **Unicode Strings:** Python 3 stores strings as Unicode by default, whereas Python 2 requires you to mark a string with a "u" if you want to store it as Unicode. Unicode strings are more versatile than ASCII strings, which are the Python 2 default, as they can store letters from foreign languages as well as emoji and the standard Roman letters and numerals. You can still label your Unicode strings with a "u" if you want to make sure your Python 3 code is compatible with Python 2.

- **Raising Exceptions:** Python 3 requires different syntax for raising exceptions. If you want to output an error message to the user, you need to use the syntax:

  raise IOError("your error message")

  This syntax works in Python 2 as well. The following code works only in Python 2, not Python 3:

  raise IOError, "your error message"

3.1.5 NODE JS

Node.js is an open-source, cross-platform JavaScript run-time environment that executes JavaScript code outside of a browser. JavaScript is used primarily for client-side scripting, in which scripts written in JavaScript are embedded in a webpage's HTML and run client-side by a JavaScript engine in the user's web browser.

Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web application development around a single programming language, rather than different languages for server side and client side scripts.

3.1.6 FLASK

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.[3] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program.

3.1.6.1 HISTORY

Flask was created by Armin Ronacher of Pocoo, an international group of Python enthusiasts formed in 2004. According to Ronacher, the idea was originally an April Fool's joke that was popular enough to make into a serious application.

When Ronacher and Georg Brandl created a bulletin board system written in Python, the Pocoo projects Werkzeug and Jinja were developed. Despite the lack of a major release, Flask has become popular among Python enthusiasts. As of mid-2016, it was the most popular Python web development framework on GitHub.

3.1.6.2 COMPONENETS

The microframework Flask is based on the Pocoo projects Werkzeug and Jinja2.

- **Werkzeug:** Werkzeug is a utility library for the Python programming language, in other words a toolkit for Web Server Gateway Interface (WSGI) applications, and is licensed under a BSD License. Werkzeug can realize software objects for request, response, and utility functions. It can be used to build a custom software framework on top of it and supports Python 2.6, 2.7 and 3.3
- **Jinja:** Jinja, also by Ronacher, is a template engine for the Python programming language and is licensed under a BSD License. Similar to the Django web framework, it provides that templates are evaluated in a sandbox.

3.1.6.3 FEATURES

- Contains development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja2 templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Extensive documentation
- Google App Engine compatibility
- Extensions available to enhance features desired

3.2 LIBRARIES

3.2.1 NUMPY

Numpy is a Library which provide the facility of easy manipulation of multi-dimensional array in Python. Without this library the facility of use of multidimensional array is little bit difficult. By importing this package, multidimensional array can be manipulated easily. Also,

other high dimensional operation of mathematics can be used with the help of this package.

3.2.2 PANDAS

Data analysis can be done very easily with the help of Python. Python provides multiple packages and libraries which provides user to different type of facilities. Pandas is one of them. This library is used for data analysis. With the help of this library user can optimized performance.

Data can be analyzing in Pandas with: - series and data frames. Series means 1-D array in Pandas. With the help of Series user can store any datatype. DataFrames means rows and columns.so Data frame is 2-D array.

3.2.3 MATPLOTLIB

Matplotlib is one of the most important library in Python. This library generally used by the user for creation of graphs. Data can be visualized very easily with the help of this library. This library supports users to create different type of graphs such as- Histogram, Power spectra, Error charts etc.

3.3 LOG FILES FORMAT

3.3.1 Android

Format: ['Date', 'Time', 'Pid', 'Tid', 'Level', 'Component', 'Content']

```
03-17 16:13:38.811  1702  2395 D WindowManager: printFreezingDisplayLogsopening app wtoken =
AppWindowToken{9f4ef63 token=Token{a64f992 ActivityRecord{de9231d u0
com.tencent.qt.qtl/.activity.info.NewsDetailXmlActivity t761}}}, allDrawn= false, startingDisplayed =
false, startingMoved =  false, isRelaunching =  false
03-17 16:13:38.819  1702  8671 D PowerManagerService: acquire lock=233570404, flags=0x1, tag="View
Lock", name=com.android.systemui, ws=null, uid=10037, pid=2227
03-17 16:13:38.820  1702  8671 D PowerManagerService:
ready=true,policy=3,wakefulness=1,wksummary=0x23,uasummary=0x1,bootcompleted=true,boostinprogress=false,
waitmodeenable=false,mode=false,manual=38,auto=-1,adj=0.0userId=0
03-17 16:13:38.839  1702  2113 V WindowManager: Skipping AppWindowToken{df0798e token=Token{78af589
ActivityRecord{3b04890 u0 com.tencent.qt.qtl/com.tencent.video.player.activity.PlayerActivity t761}}} --
going to hide
03-17 16:13:38.859  2227  2227 D TextView: visible is system.time.showampm
03-17 16:13:38.861  2227  2227 D TextView: mVisiblity.getValue is false
03-17 16:13:38.869  2227  2227 D TextView: visible is system.charge.show
03-17 16:13:38.871  2227  2227 D TextView: mVisiblity.getValue is false
03-17 16:13:38.875  2227  2227 D TextView: visible is system.call.count gt 0
```

3.3.2 Windows

Format: ['Date', 'Time', 'Level', 'Component', 'Content']

```
2016-09-28 04:30:30, Info                  CBS    Loaded Servicing Stack v6.1.7601.23505 with Core: C:
\Windows\winsxs\amd64_microsoft-windows-
servicingstack_31bf3856ad364e35_6.1.7601.23505_none_681aa442f6fed7f0\cbscore.dll
2016-09-28 04:30:31, Info                  CSI    00000001@2016/9/27:20:30:31.455 WcpInitialize (wcp.dll
version 0.0.0.6) called (stack @0x7fed806eb5d @0x7fef9fb9b6d @0x7fef9f8358f @0xff83e97c @0xff83d799
@0xff83db2f)
2016-09-28 04:30:31, Info                  CSI    00000002@2016/9/27:20:30:31.458 WcpInitialize (wcp.dll
version 0.0.0.6) called (stack @0x7fed806eb5d @0x7fefa006ade @0x7fef9fd2984 @0x7fef9f83665 @0xff83e97c
@0xff83d799)
2016-09-28 04:30:31, Info                  CSI    00000003@2016/9/27:20:30:31.458 WcpInitialize (wcp.dll
version 0.0.0.6) called (stack @0x7fed806eb5d @0x7fefa1c8728 @0x7fefa1c8856 @0xff83e474 @0xff83d7de
@0xff83db2f)
2016-09-28 04:30:31, Info                  CBS    Ending TrustedInstaller initialization.
2016-09-28 04:30:31, Info                  CBS    Starting the TrustedInstaller main loop.
2016-09-28 04:30:31, Info                  CBS    TrustedInstaller service starts successfully.
2016-09-28 04:30:31, Info                  CBS    SQM: Initializing online with Windows opt-in: False
2016-09-28 04:30:31, Info                  CBS    SQM: Cleaning up report files older than 10 days.
```

3.3.3 Apache

Format: ['Day', 'Month', 'Date', 'Time', 'Year', 'Level', 'Content']

```
[Sun Dec 04 04:47:44 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:47:44 2005] [error] mod_jk child workerEnv in error state 6
[Sun Dec 04 04:51:08 2005] [notice] jk2_init() Found child 6725 in scoreboard slot 10
[Sun Dec 04 04:51:09 2005] [notice] jk2_init() Found child 6726 in scoreboard slot 8
[Sun Dec 04 04:51:09 2005] [notice] jk2_init() Found child 6728 in scoreboard slot 6
[Sun Dec 04 04:51:14 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:14 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:14 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:18 2005] [error] mod_jk child workerEnv in error state 6
[Sun Dec 04 04:51:18 2005] [error] mod_jk child workerEnv in error state 6
[Sun Dec 04 04:51:18 2005] [error] mod_jk child workerEnv in error state 6
[Sun Dec 04 04:51:37 2005] [notice] jk2_init() Found child 6736 in scoreboard slot 10
[Sun Dec 04 04:51:38 2005] [notice] jk2_init() Found child 6733 in scoreboard slot 7
[Sun Dec 04 04:51:38 2005] [notice] jk2_init() Found child 6734 in scoreboard slot 9
[Sun Dec 04 04:51:52 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:52 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:55 2005] [error] mod_jk child workerEnv in error state 6
```

3.3.4 HDFS

Format: ['Date', 'Time', 'Pid', 'Level', 'Component', 'Content']

```
081109 203615 148 INFO dfs.DataNode$PacketResponder: PacketResponder 1 for block blk_38865049064139660
terminating
081109 203807 222 INFO dfs.DataNode$PacketResponder: PacketResponder 0 for block blk_-
6952295868487656571 terminating
081109 204005 35 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated:
10.251.73.220:50010 is added to blk_7128370237687728475 size 67108864
081109 204015 308 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_8229193803249955061
terminating
081109 204106 329 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_-
6670958622368987959 terminating
081109 204132 26 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated:
10.251.43.115:50010 is added to blk_3050920587428079149 size 67108864
081109 204324 34 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated:
10.251.203.80:50010 is added to blk_7889946331804732825 size 67108864
081109 204453 34 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated:
10.250.11.85:50010 is added to blk_2377150260128098806 size 67108864
081109 204525 512 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_572492839287299681
```

3.3.5 BGL

Format: ['Label', 'Timestamp', 'Date', 'Node', 'Time', 'Node Repeat', 'Type', 'Component', 'Level', 'Content']

```
- 1117838570 2005.06.03 R02-M1-N0-C:J12-U11 2005-06-03-15.42.50.675872 R02-M1-N0-C:J12-U11 RAS KERNEL
INFO instruction cache parity error corrected
- 1117838573 2005.06.03 R02-M1-N0-C:J12-U11 2005-06-03-15.42.53.276129 R02-M1-N0-C:J12-U11 RAS KERNEL
INFO instruction cache parity error corrected
- 1117838976 2005.06.03 R02-M1-N0-C:J12-U11 2005-06-03-15.49.36.156884 R02-M1-N0-C:J12-U11 RAS KERNEL
INFO instruction cache parity error corrected
- 1117838978 2005.06.03 R02-M1-N0-C:J12-U11 2005-06-03-15.49.38.026704 R02-M1-N0-C:J12-U11 RAS KERNEL
INFO instruction cache parity error corrected
- 1117842440 2005.06.03 R23-M0-NE-C:J05-U01 2005-06-03-16.47.20.730545 R23-M0-NE-C:J05-U01 RAS KERNEL
INFO 63543 double-hummer alignment exceptions
- 1117842974 2005.06.03 R24-M0-N1-C:J13-U11 2005-06-03-16.56.14.254137 R24-M0-N1-C:J13-U11 RAS KERNEL
INFO 162 double-hummer alignment exceptions
- 1117843015 2005.06.03 R21-M1-N6-C:J08-U11 2005-06-03-16.56.55.309974 R21-M1-N6-C:J08-U11 RAS KERNEL
INFO 141 double-hummer alignment exceptions
- 1117848119 2005.06.03 R16-M1-N2-C:J17-U01 2005-06-03-18.21.59.871925 R16-M1-N2-C:J17-U01 RAS KERNEL
INFO CE sym 2, at 0x0b85eee0, mask 0x05
APPREAD 1117869872 2005.06.04 R04-M1-N4-I:J18-U11 2005-06-04-00.24.32.432192 R04-M1-N4-I:J18-U11 RAS APP
```

3.3.6 Hadoop

Format: ['Date', 'Time', 'Level', 'Process', 'Component', 'Content']

```
2015-10-18 18:01:47,978 INFO [main] org.apache.hadoop.mapreduce.v2.app.MRAppMaster: Created MRAppMaster
for application appattempt_1445144423722_0020_000001
2015-10-18 18:01:48,963 INFO [main] org.apache.hadoop.mapreduce.v2.app.MRAppMaster: Executing with
tokens:
2015-10-18 18:01:48,963 INFO [main] org.apache.hadoop.mapreduce.v2.app.MRAppMaster: Kind:
YARN_AM_RM_TOKEN, Service: , Ident: (appAttemptId { application_id { id: 20 cluster_timestamp:
1445144423722 } attemptId: 1 } keyId: -127633188)
2015-10-18 18:01:49,228 INFO [main] org.apache.hadoop.mapreduce.v2.app.MRAppMaster: Using mapred
newApiCommitter.
2015-10-18 18:01:50,353 INFO [main] org.apache.hadoop.mapreduce.v2.app.MRAppMaster: OutputCommitter set
in config null
2015-10-18 18:01:50,509 INFO [main] org.apache.hadoop.mapreduce.v2.app.MRAppMaster: OutputCommitter is
org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2015-10-18 18:01:50,556 INFO [main] org.apache.hadoop.yarn.event.AsyncDispatcher: Registering class
org.apache.hadoop.mapreduce.jobhistory.EventType for class
org.apache.hadoop.mapreduce.jobhistory.JobHistoryEventHandler
2015-10-18 18:01:50,556 INFO [main] org.apache.hadoop.yarn.event.AsyncDispatcher: Registering class
```

3.3.7 HPC

Format: ['Log Id', 'Node', 'Component', 'State', 'Time', 'Flag', 'Content']

```
134681 node-246 unix.hw state_change.unavailable 1077804742 1 Component State Change: Component
\042SCSI-WWID:01000010:6005-08b4-0001-00c6-0006-3000-003d-0000\042 is in the unavailable state
(HWID=1973)
350766 node-109 unix.hw state_change.unavailable 1084680778 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=3180)
344518 node-246 unix.hw state_change.unavailable 1084270955 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=5089)
344448 node-153 unix.hw state_change.unavailable 1084270952 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=4088)
366633 node-200 unix.hw state_change.unavailable 1085100843 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=2538)
366463 node-122 unix.hw state_change.unavailable 1085084674 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=2480)
438190 node-228 unix.hw state_change.unavailable 1097194780 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=3713)
225111 node-10 unix.hw state_change.unavailable 1117296789 1 Component State Change: Component
\042alt0\042 is in the unavailable state (HWID=3891)
```

3.3.8 Linux

Format: ['Month', 'Date', 'Time', 'Level', 'Component', 'PID', 'Content']

```
Jun 14 15:16:01 combo sshd(pam_unix)[19939]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=218.188.2.4
Jun 14 15:16:02 combo sshd(pam_unix)[19937]: check pass; user unknown
Jun 14 15:16:02 combo sshd(pam_unix)[19937]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=218.188.2.4
Jun 15 02:04:59 combo sshd(pam_unix)[20882]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=220-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20884]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=220-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20883]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=220-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20885]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=220-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20886]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=220-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20892]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=220-135-151-1.hinet-ip.hinet.net  user=root
```

3.3.9 Mac

Format: ['Month', 'Date', 'Time', 'User', 'Component', 'PID', 'Address', 'Content']

```
Jul  1 09:00:55 calvisitor-10-105-160-95 kernel[0]: IOThunderboltSwitch<0>(0x0)::listenerCallback -
Thunderbolt HPD packet for route = 0x0 port = 11 unplug = 0
Jul  1 09:01:05 calvisitor-10-105-160-95 com.apple.CDScheduler[43]: Thermal pressure state: 1 Memory
pressure state: 0
Jul  1 09:01:06 calvisitor-10-105-160-95 QQ[10018]: FA||Url||taskID[2019352994] dealloc
Jul  1 09:02:26 calvisitor-10-105-160-95 kernel[0]: ARPT: 620701.011328:
AirPort_Brcm43xx::syncPowerState: WWEN[enabled]
Jul  1 09:02:26 authorMacBook-Pro kernel[0]: ARPT: 620702.879952: AirPort_Brcm43xx::platformWoWEnable:
WWEN[disable]
Jul  1 09:03:11 calvisitor-10-105-160-95 mDNSResponder[91]: mDNS_DeregisterInterface: Frequent
transitions for interface awdl0 (FE80:0000:0000:0000:D8A5:90FF:FEF5:7FFF)
Jul  1 09:03:13 calvisitor-10-105-160-95 kernel[0]: ARPT: 620749.901374: IOPMPowerSource Information:
onSleep,  SleepType: Normal Sleep,  'ExternalConnected': Yes, 'TimeRemaining': 0,
Jul  1 09:04:33 calvisitor-10-105-160-95 kernel[0]: ARPT: 620750.434035: wl0: wl_update_tcpkeep_seq:
Original Seq: 3226706533, Ack: 3871687177, Win size: 4096
Jul  1 09:04:33 authorMacBook-Pro kernel[0]: ARPT: 620752.337198: ARPT: Wake Reason: Wake on Scan
offload
```

3.3.10 OpenSSH

Format: ['Date', 'Day', 'Time', 'Component', 'Pid', 'Content']

```
Dec 10 06:55:46 LabSZ sshd[24200]: reverse mapping checking getaddrinfo for ns.marryaldkfaczcz.com
[173.234.31.186] failed - POSSIBLE BREAK-IN ATTEMPT!
Dec 10 06:55:46 LabSZ sshd[24200]: Invalid user webmaster from 173.234.31.186
Dec 10 06:55:46 LabSZ sshd[24200]: input_userauth_request: invalid user webmaster [preauth]
Dec 10 06:55:46 LabSZ sshd[24200]: pam_unix(sshd:auth): check pass; user unknown
Dec 10 06:55:46 LabSZ sshd[24200]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0
tty=ssh ruser= rhost=173.234.31.186
Dec 10 06:55:48 LabSZ sshd[24200]: Failed password for invalid user webmaster from 173.234.31.186 port
38926 ssh2
Dec 10 06:55:48 LabSZ sshd[24200]: Connection closed by 173.234.31.186 [preauth]
Dec 10 07:02:47 LabSZ sshd[24203]: Connection closed by 212.47.254.145 [preauth]
Dec 10 07:07:38 LabSZ sshd[24206]: Invalid user test9 from 52.80.34.196
Dec 10 07:07:38 LabSZ sshd[24206]: input_userauth_request: invalid user test9 [preauth]
Dec 10 07:07:38 LabSZ sshd[24206]: pam_unix(sshd:auth): check pass; user unknown
Dec 10 07:07:38 LabSZ sshd[24206]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0
tty=ssh ruser= rhost=ec2-52-80-34-196.cn-north-1.compute.amazonaws.com.cn
Dec 10 07:07:45 LabSZ sshd[24206]: Failed password for invalid user test9 from 52.80.34.196 port 36060
```

3.3.11 OpenStack

Format: ['Log record', 'Date', 'Time', 'Pid', 'Level', 'Component', 'ADDR', 'Content']

```
nova-api.log.1.2017-05-16_13:53:08 2017-05-16 00:00:00.008 25746 INFO nova.osapi_compute.wsgi.server
[req-38101a0b-2096-447d-96ea-a692162415ae 113d3a99c3da401fbd62cc2caa5b96d2
54fadb412c4e40cdbaed9335e4c35a9e - - -] 10.11.10.1 "GET
/v2/54fadb412c4e40cdbaed9335e4c35a9e/servers/detail HTTP/1.1" status: 200 len: 1893 time: 0.2477829
nova-api.log.1.2017-05-16_13:53:08 2017-05-16 00:00:00.272 25746 INFO nova.osapi_compute.wsgi.server
[req-9bc36dd9-91c5-4314-898a-47625eb93b09 113d3a99c3da401fbd62cc2caa5b96d2
54fadb412c4e40cdbaed9335e4c35a9e - - -] 10.11.10.1 "GET
/v2/54fadb412c4e40cdbaed9335e4c35a9e/servers/detail HTTP/1.1" status: 200 len: 1893 time: 0.2577181
nova-api.log.1.2017-05-16_13:53:08 2017-05-16 00:00:01.551 25746 INFO nova.osapi_compute.wsgi.server
[req-55db2d8d-cdb7-4b4b-993b-429be84c0c3e 113d3a99c3da401fbd62cc2caa5b96d2
54fadb412c4e40cdbaed9335e4c35a9e - - -] 10.11.10.1 "GET
/v2/54fadb412c4e40cdbaed9335e4c35a9e/servers/detail HTTP/1.1" status: 200 len: 1893 time: 0.2731631
nova-api.log.1.2017-05-16_13:53:08 2017-05-16 00:00:01.813 25746 INFO nova.osapi_compute.wsgi.server
[req-2a3dc421-6604-42a7-9390-a18dc824d5d6 113d3a99c3da401fbd62cc2caa5b96d2
54fadb412c4e40cdbaed9335e4c35a9e - - -] 10.11.10.1 "GET
/v2/54fadb412c4e40cdbaed9335e4c35a9e/servers/detail HTTP/1.1" status: 200 len: 1893 time: 0.2580249
nova-api.log.1.2017-05-16_13:53:08 2017-05-16 00:00:03.091 25746 INFO nova.osapi_compute.wsgi.server
```

3.3.12 Proxifier

Format: ['Time', 'Program', 'Content']

```
[10.30 16:49:06] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through proxy proxy.cse.cuhk.edu.hk:5070
HTTPS[10.30 16:49:06] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through proxy
proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30 16:49:06] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through
proxy proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30 16:49:07] chrome.exe - proxy.cse.cuhk.edu.hk:5070 close, 0
bytes sent, 0 bytes received, lifetime 00:01[10.30 16:49:07] chrome.exe - proxy.cse.cuhk.edu.hk:5070
open through proxy proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30 16:49:07] chrome.exe -
proxy.cse.cuhk.edu.hk:5070 open through proxy proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30 16:49:07]
chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through proxy proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30
16:49:07] chrome.exe - proxy.cse.cuhk.edu.hk:5070 close, 403 bytes sent, 426 bytes received, lifetime <1
sec[10.30 16:49:07] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through proxy
proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30 16:49:07] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through
proxy proxy.cse.cuhk.edu.hk:5070 HTTPS[10.30 16:49:07] chrome.exe - proxy.cse.cuhk.edu.hk:5070 close,
451 bytes sent, 18846 bytes (18.4 KB) received, lifetime <1 sec[10.30 16:49:08] chrome.exe -
proxy.cse.cuhk.edu.hk:5070 close, 445 bytes sent, 5174 bytes (5.05 KB) received, lifetime <1 sec[10.30
16:49:08] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through proxy proxy.cse.cuhk.edu.hk:5070 HTTPS
[10.30 16:49:08] chrome.exe - proxy.cse.cuhk.edu.hk:5070 close, 1190 bytes (1.16 KB) sent, 1671 bytes
(1.63 KB) received, lifetime 00:02[10.30 16:49:08] chrome.exe - proxy.cse.cuhk.edu.hk:5070 open through
```

3.3.13 Spark

Format: ['Date', 'Time', 'Level', 'Content']

```
17/06/09 20:10:40 INFO executor.CoarseGrainedExecutorBackend: Registered signal handlers for [TERM, HUP,
INT]
17/06/09 20:10:40 INFO spark.SecurityManager: Changing view acls to: yarn,curi
17/06/09 20:10:40 INFO spark.SecurityManager: Changing modify acls to: yarn,curi
17/06/09 20:10:40 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls
disabled; users with view permissions: Set(yarn, curi); users with modify permissions: Set(yarn, curi)
17/06/09 20:10:41 INFO spark.SecurityManager: Changing view acls to: yarn,curi
17/06/09 20:10:41 INFO spark.SecurityManager: Changing modify acls to: yarn,curi
17/06/09 20:10:41 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls
disabled; users with view permissions: Set(yarn, curi); users with modify permissions: Set(yarn, curi)
17/06/09 20:10:41 INFO slf4j.Slf4jLogger: Slf4jLogger started
17/06/09 20:10:41 INFO Remoting: Starting remoting
17/06/09 20:10:41 INFO Remoting: Remoting started; listening on addresses :
[akka.tcp://sparkExecutorActorSystem@mesos-slave-07:55904]
17/06/09 20:10:41 INFO util.Utils: Successfully started service 'sparkExecutorActorSystem' on port
55904.
17/06/09 20:10:41 INFO storage.DiskBlockManager: Created local directory at
```

3.3.14 ThunderBird

Format: ['Label', 'Timestamp', 'Date', 'User', 'Month', 'Day', 'Time', 'Location', 'Component', 'Content']

```
- 1131566461 2005.11.09 dn228 Nov 9 12:01:01 dn228/dn228 crond(pam_unix)[2915]: session closed for user
root
- 1131566461 2005.11.09 dn228 Nov 9 12:01:01 dn228/dn228 crond(pam_unix)[2915]: session opened for user
root by (uid=0)
- 1131566461 2005.11.09 dn228 Nov 9 12:01:01 dn228/dn228 crond[2916]: (root) CMD (run-parts
/etc/cron.hourly)
- 1131566461 2005.11.09 dn261 Nov 9 12:01:01 dn261/dn261 crond(pam_unix)[2907]: session closed for user
root
- 1131566461 2005.11.09 dn261 Nov 9 12:01:01 dn261/dn261 crond(pam_unix)[2907]: session opened for user
root by (uid=0)
- 1131566461 2005.11.09 dn261 Nov 9 12:01:01 dn261/dn261 crond[2908]: (root) CMD (run-parts
/etc/cron.hourly)
- 1131566461 2005.11.09 dn3 Nov 9 12:01:01 dn3/dn3 crond(pam_unix)[2907]: session closed for user root
- 1131566461 2005.11.09 dn3 Nov 9 12:01:01 dn3/dn3 crond(pam_unix)[2907]: session opened for user root
by (uid=0)
- 1131566461 2005.11.09 dn3 Nov 9 12:01:01 dn3/dn3 crond[2908]: (root) CMD (run-parts /etc/cron.hourly)
- 1131566461 2005.11.09 dn596 Nov 9 12:01:01 dn596/dn596 crond(pam_unix)[2727]: session closed for user
```

3.4 Restful Services to integrate Angular and Python

RESTful Web Services are basically REST Architecture based Web Services. In REST Architecture everything is a resource. RESTful web services are light weight, highly scalable and maintainable and are very commonly used to create APIs for web-based applications.

3.4.1 What is REST architecture?

REST stands for REpresentational State Transfer. REST is web standards based architecture and uses HTTP Protocol. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods. REST was first introduced by Roy Fielding in 2000.

In REST architecture, a REST Server simply provides access to resources and REST client accesses and modifies the resources. Here each resource is identified by URIs/ global IDs. REST uses various representation to represent a resource like text, JSON, XML. JSON is the most popular one.

3.4.2 HTTP methods

Following four HTTP methods are commonly used in REST based architecture.

- **GET** − Provides a read only access to a resource.
- **POST** − Used to create a new resource.
- **DELETE** − Used to remove a resource.

- **PUT** − Used to update a existing resource or create a new resource.

3.4.3 Introduction to RESTFul web services

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

Web services based on REST Architecture are known as RESTful web services. These webservices uses HTTP methods to implement the concept of REST architecture. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation such as JSON and set of HTTP Methods.

3.4.3.1 What is a Resource?

REST architecture treats every content as a resource. These resources can be Text Files, Html Pages, Images, Videos or Dynamic Business Data. REST Server simply provides access to resources and REST client accesses and modifies the resources. Here each resource is identified by URIs/ Global IDs. REST uses various representations to represent a resource where Text, JSON, XML. The most popular representations of resources are XML and JSON.

3.4.3.2 Representation of Resources

A resource in REST is a similar Object in Object Oriented Programming or is like an Entity in a Database. Once a resource is identified then its representation is to be decided using a standard format so that the server can send the resource in the above said format and client

3.4.3.3 Good Resources Representation

REST does not impose any restriction on the format of a resource representation. A client can ask for JSON representation whereas another client may ask for XML representation of

the same resource to the server and so on. It is the responsibility of the REST server to pass the client the resource in the format that the client understands.

Following are some important points to be considered while designing a representation format of a resource in RESTful Web Services.

- **Understandability** − Both the Server and the Client should be able to understand and utilize the representation format of the resource.
- **Completeness** − Format should be able to represent a resource completely. For example, a resource can contain another resource. Format should be able to represent simple as well as complex structures of resources.
- **Linkablity** − A resource can have a linkage to another resource, a format should be able to handle such situations.

However, at present most of the web services are representing resources using either XML or JSON format. There are plenty of libraries and tools available to understand, parse, and modify XML and JSON data.

Chapter - 4

System Design & Implementation

This section describes in more detail some design issues and explains decisions that have been made.

4.1 Language of Metadata

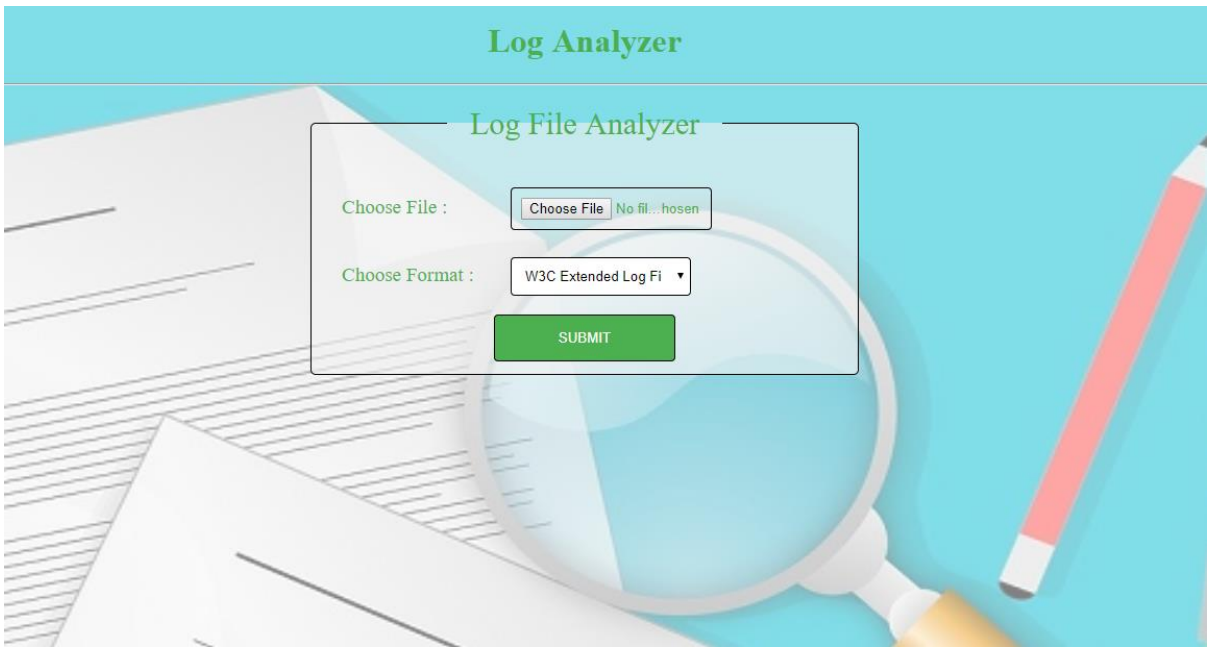
The first stage of log file analysis is the process of understanding log content. This is done by lexical scanning of log entries and parsing them into internal data structures according to a given syntax and semantics. The semantics can be either explicitly specified in the log file or it must be provided by a human (preferably by the designer of the involved software product) in another way, i.e. by a description in corresponding metadata file.

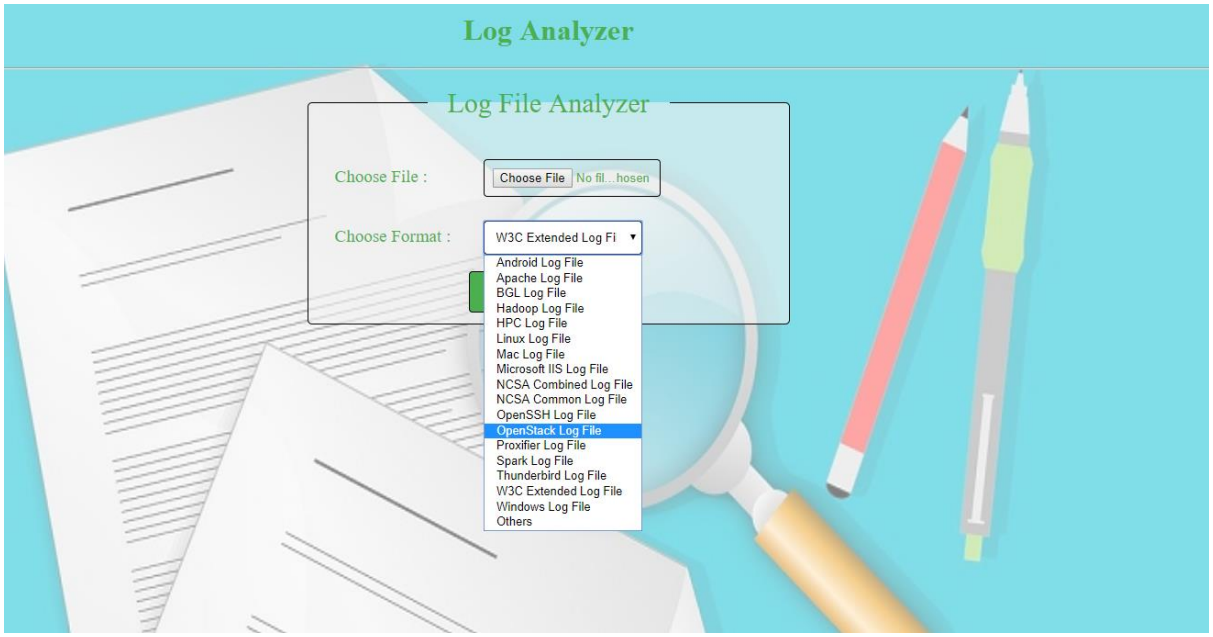
4.2 Operation

Before a detailed description of analyzer's components, let us enumerate the basic stages of log processing from user point of view and thus explain how the analyzer operates.

1. Job Setup :

In a typical session, the user specifies via an user interface one or more log files, a metadata file and selects modules that should participate in the job, each module for one analytical task.





2. Send Metadata (Angular):

The analyzer then reads metadata information (using lexical and syntax analyzers) and thus send the structure of the given log file to the server. We have created Rest API Service in Angular Client Side to send the data to the Backend Server i.e Flask.

```
fileChanged($event)
{
  this.file = $event.target.files[0];
  let fileReader = new FileReader();
  fileReader.onload = (file) => {
    this.file_text = fileReader.result
    console.log(this.file_text);
  }
  console.log(fileReader.readAsText(this.file));
}

    sendFile(f, format)
    {
      var a = f.split("\\");
      var file_name = a[a.length-1];
      this.rs.readFile(this.file_text, file_name, format)
      .subscribe
      (
        (response) =>
        {
          console.log(response);
          this.rs.responseData = response;
          this.auth.setToken("file");
          this.router.navigate([""]);

        },
        (error) =>
        {
          console.log("File doesn't exist..." + error);
          alert("File doesn't exist...");
        }

      )
    }
```

REST API Service (Angular) :

```
export class RestService {

  constructor(private http : Http) { }

  responseData : any;

  readUrl : string = "http://127.0.0.1:5000/readFile/";
  searchUrl : string = "http://127.0.0.1:5000/search?key1=";
  sortUrl : string = "http://127.0.0.1:5000/sort?key1=";

  readFile(file_text, file_name, format)
  {
    console.log(file_name);
    console.log(format);
    console.log(file_text);
    let arr = [file_name, format, file_text]
    return this.http.post(this.readUrl,arr)
    .map((response : any ) => response.json());
  }
```

3. Read Metadata (Flask) :

After sending the metadata from Angular i.e from Client Side, now we have to fetch that data. So for that we integrate Angular Rest Services and Python Rest Services using Flask Web Framework.

REST API Service (Python) :

```
@app.route("/readFile/", methods = ['POST'])
def ReadFile():
    global resList
    text = request.data
    ff = text.decode('utf-8').split(",")
    file_name = ''.join(ff[0][4:]).replace("\"",'').strip()
    file_format = ff[1].replace("\\n",'').replace("\"",'').strip()
    file_text = ''.join(ff[2:]).replace("\\r","")
    file_text = str(file_text.replace("\\n","\n").replace("\"",'').strip())
    file_text1 = file_text.strip().split("\n")
    del file_text1[len(file_text1)-1]

    format = titles[file_format]['log_format']
    rex =  titles[file_format]['regex'][0]

    mainList = []
    for line in file_text1:
        l = re.findall(rex, line)
        size = len(format)
        size1 = len(l)
        if(size1<size):
            pass
        else:
            l[size-1] = ' '.join(l[size-1:])
            del l[size:]

        d = OrderedDict()
        for k ,v in it.zip_longest(format,l):
            d[k] = v
        mainList.append(d)
        resList = mainList

        df = pd.DataFrame(mainList, columns=format)
        df.to_csv(file_name+"_structured.csv", sep=',', encoding='utf-8', index=False)

    return jsonify(mainList,format)
```

4. Filtering & Cleaning

After learning log structure, the analyzer starts reading the log file and according the metadata

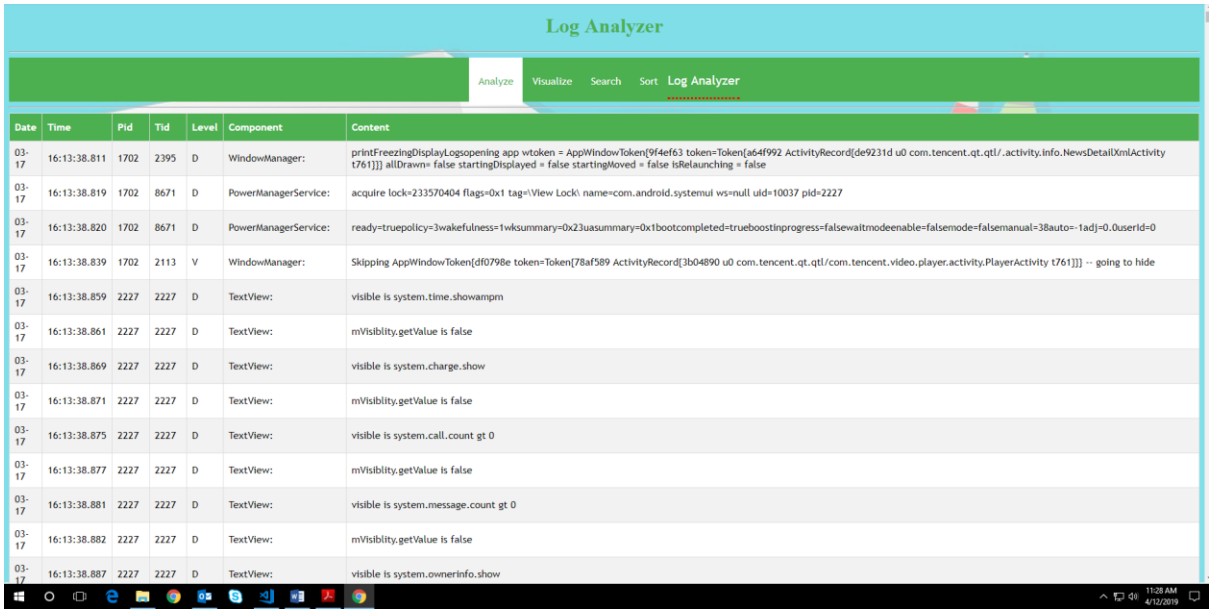
stores pieces of information into the structured file of .csv format. During this process some unwanted records can be filtered out or some selected parts of each record can be omitted to reduce the file size.

```
    d = OrderedDict()
    for k ,v in it.zip_longest(format,l):
        d[k] = v
    mainList.append(d)
    resList = mainList

    df = pd.DataFrame(mainList, columns=format)
    df.to_csv(file_name+"_structured.csv", sep=',', encoding='utf-8', index=False)
```

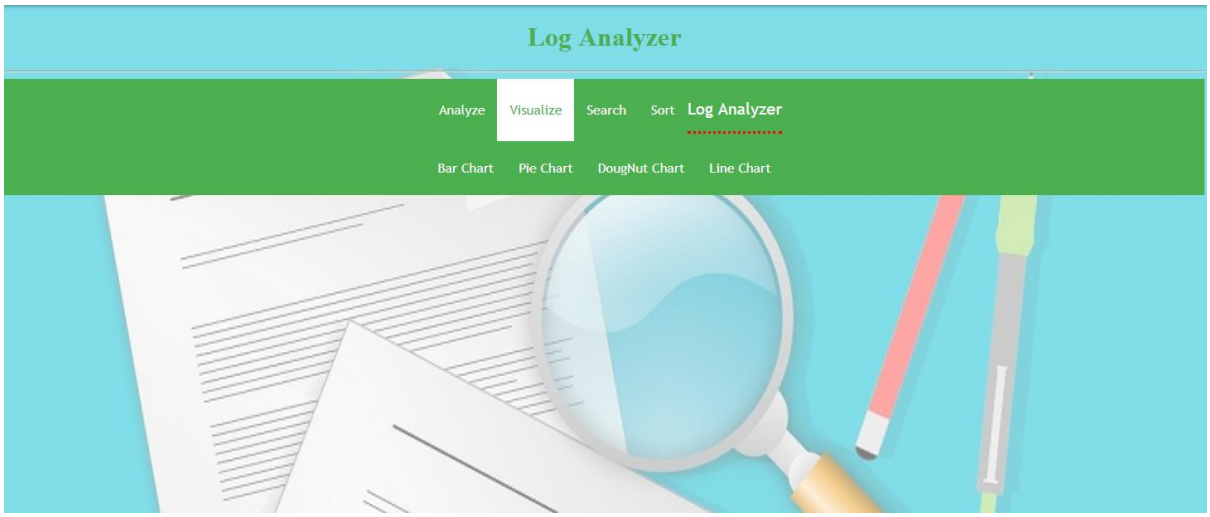
5. Data Analysis :

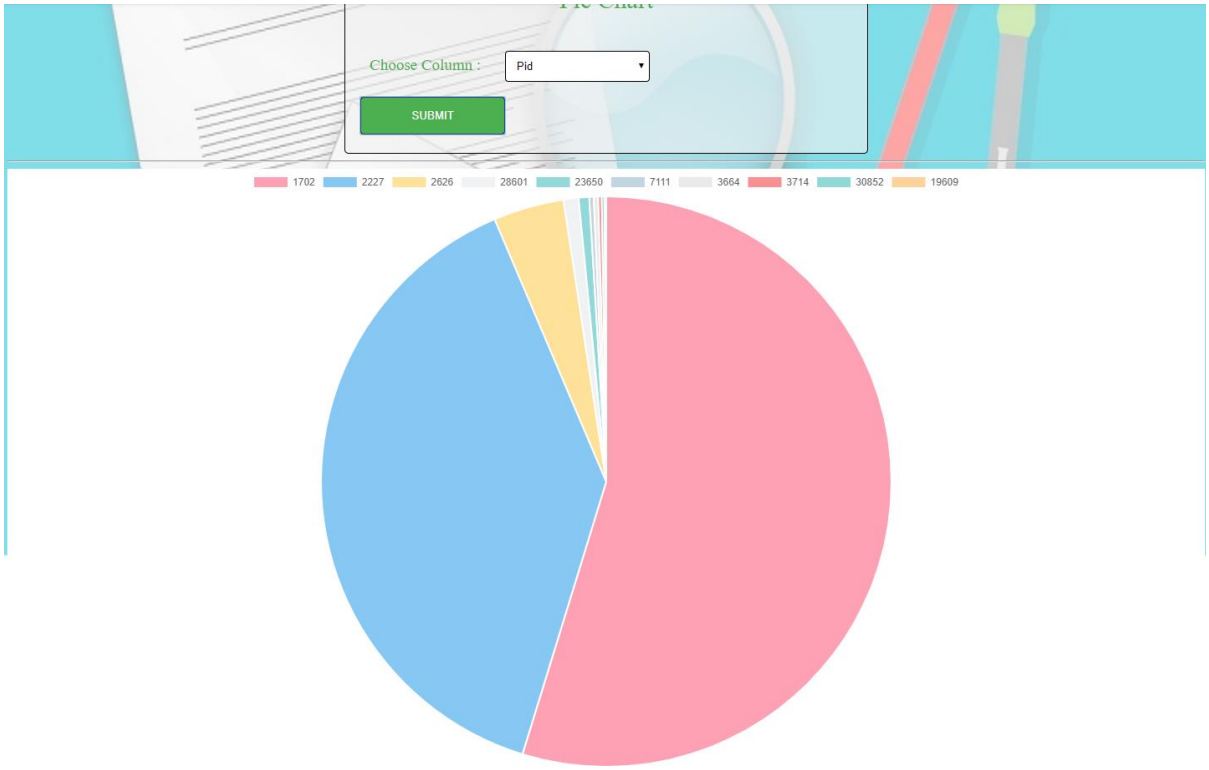
When all logs are read and stored in the file, the analyzers executes one-after-one all user written modules that perform analytical tasks. Each program is expected to produce some output in form of a set of variables or the output may be missing if the log does not contain the sought-after pattern etc.

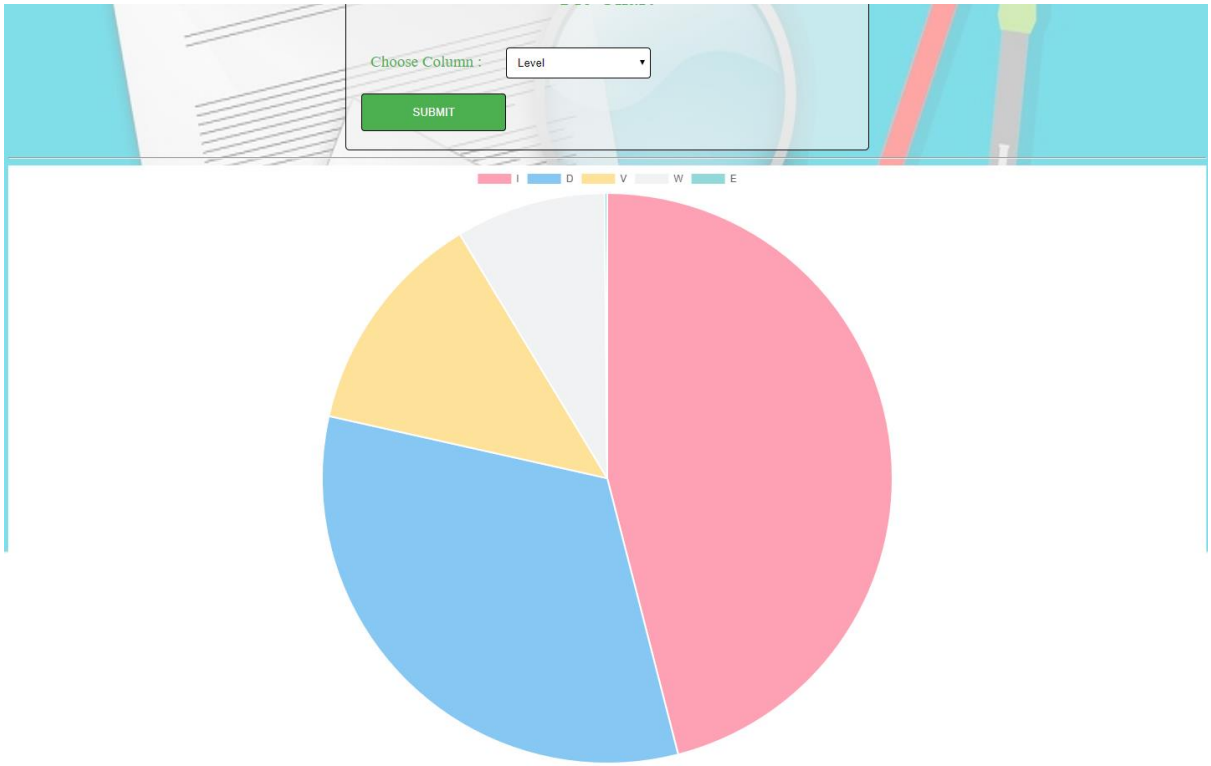


6. Processing of Results (Visualization) :

Output of each module (if any) is passed to the visualization module that uses yet another type of files (visual plug-ins) to learn how to present obtained results. For example, it can transform the obtained variables into pie charts, bar graphs, nice{looking tables, lists, sentences etc. Finally, the transformed results are either displayed to user via the user interface module or they are stored in an output file.



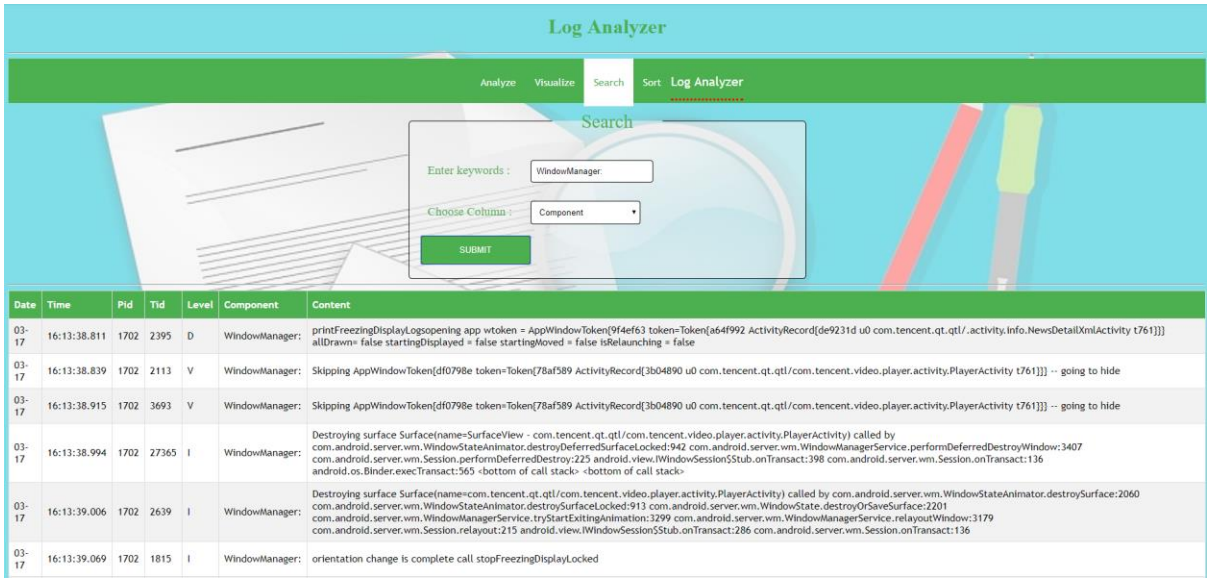




7. Searching :

In this operation, client can search the queries to get result of particular fields. The Search operation is fully performed by the Backend Server i.e using Python REST API Service and the results are directly displayed on the Angular i.e on Client Side.

Example : Client searches in the column of "Component" for Keywords as "WindowManager". The Server responds only that rows which have "WindowManager" in "Component" column.

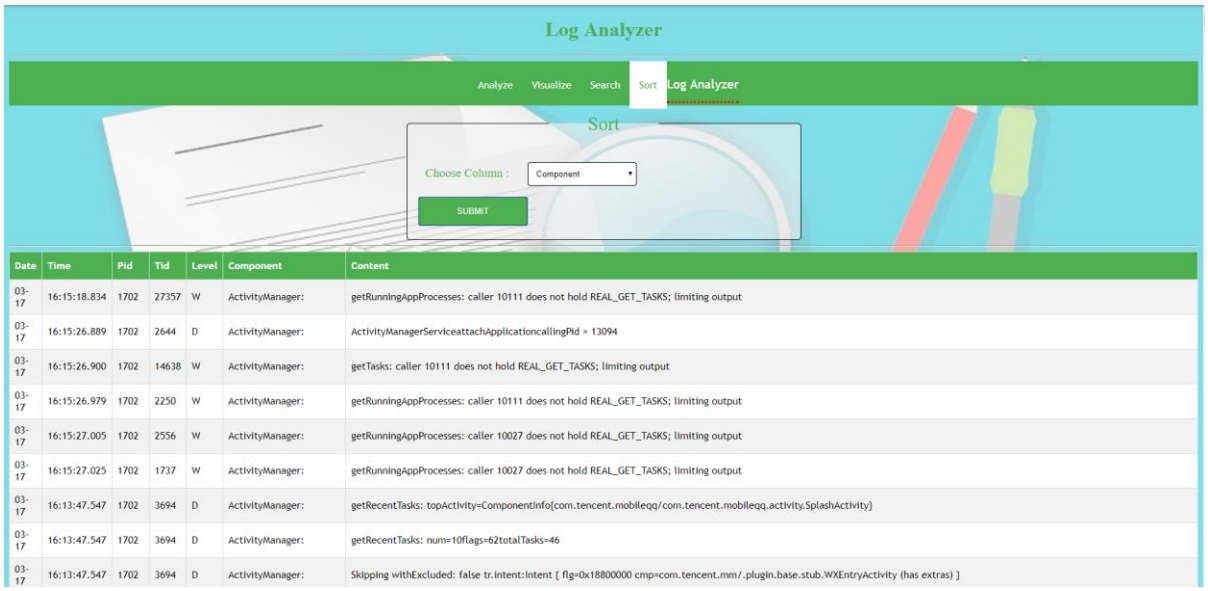


Date	Time	Pid	Tid	Level	Component	Content
03-17	16:13:38.811	1702	2395	D	WindowManager:	printFreezingDisplayLogsopening app wtoken = AppWindowToken{9f4ef63 token=Token{a64f992 ActivityRecord{de9231d u0 com.tencent.qt.qtl/.activity.info.NewsDetailXmlActivity t761}}} allDrawn= false startingDisplayed = false startingMoved = false isRelaunching = false
03-17	16:13:38.839	1702	2113	V	WindowManager:	Skipping AppWindowToken{df0798e token=Token{78af589 ActivityRecord{3b04890 u0 com.tencent.qt.qtl/com.tencent.video.player.activity.PlayerActivity t761}}} -- going to hide
03-17	16:13:38.915	1702	3693	V	WindowManager:	Skipping AppWindowToken{df0798e token=Token{78af589 ActivityRecord{3b04890 u0 com.tencent.qt.qtl/com.tencent.video.player.activity.PlayerActivity t761}}} -- going to hide
03-17	16:13:38.994	1702	27365	I	WindowManager:	Destroying surface Surface(name=SurfaceView - com.tencent.qt.qtl/com.tencent.video.player.activity.PlayerActivity) called by com.android.server.wm.WindowStateAnimator.destroyDeferredSurfaceLocked:942 com.android.server.wm.WindowManagerService.performDeferredDestroyWindow:3407 com.android.server.wm.Session.performDeferredDestroy:225 android.view.IWindowSession$Stub.onTransact:398 com.android.server.wm.Session.onTransact:136 android.os.Binder.execTransact:565 <bottom of call stack> <bottom of call stack>
03-17	16:13:39.006	1702	2639	I	WindowManager:	Destroying surface Surface(name=com.tencent.qt.qtl/com.tencent.video.player.activity.PlayerActivity) called by com.android.server.wm.WindowStateAnimator.destroySurface:2060 com.android.server.wm.WindowStateAnimator.destroySurfaceLocked:913 com.android.server.wm.WindowState.destroyOrSaveSurface:2201 com.android.server.wm.WindowManagerService.tryStartExitingAnimation:3299 com.android.server.wm.WindowManagerService.relayoutWindow:3179 com.android.server.wm.Session.relayout:215 android.view.IWindowSession$Stub.onTransact:286 com.android.server.wm.Session.onTransact:136
03-17	16:13:39.069	1702	1815	I	WindowManager:	orientation change is complete call stopFreezingDisplayLocked

8. Sorting

In this operation, client sort the data on the basis of columns fields. The Sort operation is fully performed by the Backend Server i.e using Python REST API Service and the results are directly displayed on the Angular i.e on Client Side.

Example : Client requested to sort data on the basis of column “Component”. The Server responds or sort data according to column “Component”.



Date	Time	Pid	Tid	Level	Component	Content
03-17	16:15:18.834	1702	27357	W	ActivityManager:	getRunningAppProcesses: caller 10111 does not hold REAL_GET_TASKS; limiting output
03-17	16:15:26.889	1702	2644	D	ActivityManager:	ActivityManagerServiceattachApplicationcallingPid = 13094
03-17	16:15:26.900	1702	14638	W	ActivityManager:	getTasks: caller 10111 does not hold REAL_GET_TASKS; limiting output
03-17	16:15:26.979	1702	2250	W	ActivityManager:	getRunningAppProcesses: caller 10111 does not hold REAL_GET_TASKS; limiting output
03-17	16:15:27.005	1702	2556	W	ActivityManager:	getRunningAppProcesses: caller 10027 does not hold REAL_GET_TASKS; limiting output
03-17	16:15:27.025	1702	1737	W	ActivityManager:	getRunningAppProcesses: caller 10027 does not hold REAL_GET_TASKS; limiting output
03-17	16:13:47.547	1702	3694	D	ActivityManager:	getRecentTasks: topActivity=ComponentInfo{com.tencent.mobileqq/com.tencent.mobileqq.activity.SplashActivity}
03-17	16:13:47.547	1702	3694	D	ActivityManager:	getRecentTasks: num=10flags=62totalTasks=46
03-17	16:13:47.547	1702	3694	D	ActivityManager:	Skipping withExcluded: false tr.intent:Intent { flg=0x18800000 cmp=com.tencent.mm/.plugin.base.stub.WXEntryActivity (has extras) }

Chapter - 5

TEST PLAN

Table 3: Number of different tests to be performed

Type of Test	Will Test Be Performed?	Comments/Explanations	Software Component
Requirements Testing	Yes	We need to check if the requirements of users are met or not	All Modules
Unit Testing	Yes	We need to test all the modules.	All Modules
Integration	Yes	We need to test if all the stages work synchronously	All Modules
Performance	Yes	We need to test the accuracy of the model at the last module.	All Modules
Stress	Yes	We need to test the stress on the system while executing the application.	All Modules
Compliance	Yes	The application must be compatible with the system.	All Modules
Security	No	There are no security issues	None
Load	Yes	We need to make sure that the system does not get overwhelmed by loading the data from social networking website or news websites.	All Modules

Chapter – 6

RESULT ANALYSIS

Output LOG FILES

- Android Log

LineId	Date	Time	Pid	Tid	Level	Compone	Content	EventId	EventTemplate
1	17-Mar	13:38.8	1702	2395	D	WindowM	printFreez	E100	printFreezingDisplayLogsopening app wtoken = AppWindowToken{<*> token=Token{<*> ActivityRecord{<*> u0 <*>/.<*> t761}}}, allDrawn= false, startingDisplayed = false, startingMoved = false, isRelaunching =
2	17-Mar	13:38.8	1702	8671	D	PowerMai	acquire lo	E10	acquire lock=<*>, flags=<*>, tag="<*>", name=<*>, ws=<*>, uid=<*>, pid=<*>
3	17-Mar	13:38.8	1702	8671	D	PowerMai	ready=tru	E103	ready=true,policy=<*>,wakefulness=<*>,wksummary=<*>,uasummary=<*>,bootcompleted=true,boostinprogress=false,waitmodeenable=false,mode=false,manual=<*>,auto=<*>,adj=<*>.0userId=<*>
4	17-Mar	13:38.8	1702	2113	V	WindowM	Skipping A	E131	Skipping AppWindowToken{<*> token=Token{<*> ActivityRecord{<*> u0 <*>}}} -- going to hide
5	17-Mar	13:38.9	2227	2227	D	TextView	visible is s	E165	visible is <*>
6	17-Mar	13:38.9	2227	2227	D	TextView	mVisiblity	E82	mVisiblity.getValue is false
7	17-Mar	13:38.9	2227	2227	D	TextView	visible is s	E165	visible is <*>
8	17-Mar	13:38.9	2227	2227	D	TextView	mVisiblity	E82	mVisiblity.getValue is false
9	17-Mar	13:38.9	2227	2227	D	TextView	visible is s	E166	visible is <*> gt <*>
10	17-Mar	13:38.9	2227	2227	D	TextView	mVisiblity	E82	mVisiblity.getValue is false
11	17-Mar	13:38.9	2227	2227	D	TextView	visible is s	E166	visible is <*> gt <*>
12	17-Mar	13:38.9	2227	2227	D	TextView	mVisiblity	E82	mVisiblity.getValue is false
13	17-Mar	13:38.9	2227	2227	D	TextView	visible is s	E165	visible is <*>
14	17-Mar	13:38.9	2227	2227	D	TextView	mVisiblity	E82	mVisiblity.getValue is false
15	17-Mar	13:38.9	1702	10454	D	PowerMai	release:lo	E108	release:lock=<*>, flg=<*>, tag="<*>", name=<*>, ws=<*>, uid=<*>, pid=<*>
16	17-Mar	13:38.9	1702	10454	D	PowerMai	ready=tru	E103	ready=true,policy=<*>,wakefulness=<*>,wksummary=<*>,uasummary=<*>,bootcompleted=true,boostinprogress=false,waitmodeenable=false,mode=false,manual=<*>,auto=<*>,adj=<*>.0userId=<*>
17	17-Mar	13:38.9	1702	3693	V	WindowM	Skipping A	E131	Skipping AppWindowToken{<*> token=Token{<*> ActivityRecord{<*> u0 <*>}}} -- going to hide

- Apache log

LineId	Time	Level	Content	EventId	EventTemplate
1	Sun Dec 0	notice	workerEnv	E2	workerEnv.init() ok <*>
2	Sun Dec 0	error	mod_jk ch	E3	mod_jk child workerEnv in error state <*>
3	Sun Dec 0	notice	jk2_init()	E1	jk2_init() Found child <*> in scoreboard slot <*>
4	Sun Dec 0	notice	jk2_init()	E1	jk2_init() Found child <*> in scoreboard slot <*>
5	Sun Dec 0	notice	jk2_init()	E1	jk2_init() Found child <*> in scoreboard slot <*>
6	Sun Dec 0	notice	workerEnv	E2	workerEnv.init() ok <*>
7	Sun Dec 0	notice	workerEnv	E2	workerEnv.init() ok <*>
8	Sun Dec 0	notice	workerEnv	E2	workerEnv.init() ok <*>
9	Sun Dec 0	error	mod_jk ch	E3	mod_jk child workerEnv in error state <*>
10	Sun Dec 0	error	mod_jk ch	E3	mod_jk child workerEnv in error state <*>
11	Sun Dec 0	error	mod_jk ch	E3	mod_jk child workerEnv in error state <*>
12	Sun Dec 0	notice	jk2_init()	E1	jk2_init() Found child <*> in scoreboard slot <*>
13	Sun Dec 0	notice	jk2_init()	E1	jk2_init() Found child <*> in scoreboard slot <*>
14	Sun Dec 0	notice	jk2_init()	E1	jk2_init() Found child <*> in scoreboard slot <*>
15	Sun Dec 0	notice	workerEnv	E2	workerEnv.init() ok <*>

- BGL log

LineId	Label	Timestam	Date	Node	Time	NodeRepo	Type	Compone	Level	Content	EventId	EventTemplate
1	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
2	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
3	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
4	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
5	-	1.12E+09	2005.06.0	R23-M0-N	2005-06-0	R23-M0-N	RAS	KERNEL	INFO	63543 dou	E3	<*> double-hummer alignment exceptions
6	-	1.12E+09	2005.06.0	R24-M0-N	2005-06-0	R24-M0-N	RAS	KERNEL	INFO	162 doubl	E3	<*> double-hummer alignment exceptions
7	-	1.12E+09	2005.06.0	R21-M1-N	2005-06-0	R21-M1-N	RAS	KERNEL	INFO	141 doubl	E3	<*> double-hummer alignment exceptions
8	-	1.12E+09	2005.06.0	R16-M1-N	2005-06-0	R16-M1-N	RAS	KERNEL	INFO	CE sym 2,	E18	CE sym <*>, at <*>, mask <*>
9	APPREAD	1.12E+09	2005.06.0	R04-M1-N	2005-06-0	R04-M1-N	RAS	APP	FATAL	ciod: faile	E33	ciod: failed to read message prefix on control stream (CioStream socket to <*>:<*>
10	APPREAD	1.12E+09	2005.06.0	R27-M1-N	2005-06-0	R27-M1-N	RAS	APP	FATAL	ciod: faile	E33	ciod: failed to read message prefix on control stream (CioStream socket to <*>:<*>
11	-	1.12E+09	2005.06.0	R30-M0-N	2005-06-0	R30-M0-N	RAS	KERNEL	INFO	CE sym 20	E18	CE sym <*>, at <*>, mask <*>
12	-	1.12E+09	2005.06.0	R25-M0-N	2005-06-0	R25-M0-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
13	-	1.12E+09	2005.06.0	R24-M1-N	2005-06-0	R24-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
14	-	1.12E+09	2005.06.0	R24-M1-N	2005-06-0	R24-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
15	-	1.12E+09	2005.06.0	R20-M1-N	2005-06-0	R20-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
16	-	1.12E+09	2005.06.0	R24-M0-N	2005-06-0	R24-M0-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
17	-	1.12E+09	2005.06.0	R21-M1-N	2005-06-0	R21-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>

- Hadoop log

LineId	Label	Timestam	Date	Node	Time	NodeRepo	Type	Compone	Level	Content	EventId	EventTemplate
1	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
2	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
3	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
4	-	1.12E+09	2005.06.0	R02-M1-N	2005-06-0	R02-M1-N	RAS	KERNEL	INFO	instructio	E77	instruction cache parity error corrected
5	-	1.12E+09	2005.06.0	R23-M0-N	2005-06-0	R23-M0-N	RAS	KERNEL	INFO	63543 dou	E3	<*> double-hummer alignment exceptions
6	-	1.12E+09	2005.06.0	R24-M0-N	2005-06-0	R24-M0-N	RAS	KERNEL	INFO	162 doubl	E3	<*> double-hummer alignment exceptions
7	-	1.12E+09	2005.06.0	R21-M1-N	2005-06-0	R21-M1-N	RAS	KERNEL	INFO	141 doubl	E3	<*> double-hummer alignment exceptions
8	-	1.12E+09	2005.06.0	R16-M1-N	2005-06-0	R16-M1-N	RAS	KERNEL	INFO	CE sym 2,	E18	CE sym <*>, at <*>, mask <*>
9	APPREAD	1.12E+09	2005.06.0	R04-M1-N	2005-06-0	R04-M1-N	RAS	APP	FATAL	ciod: faile	E33	ciod: failed to read message prefix on control stream (CioStream socket to <*>:<*>
10	APPREAD	1.12E+09	2005.06.0	R27-M1-N	2005-06-0	R27-M1-N	RAS	APP	FATAL	ciod: faile	E33	ciod: failed to read message prefix on control stream (CioStream socket to <*>:<*>
11	-	1.12E+09	2005.06.0	R30-M0-N	2005-06-0	R30-M0-N	RAS	KERNEL	INFO	CE sym 20	E18	CE sym <*>, at <*>, mask <*>
12	-	1.12E+09	2005.06.0	R25-M0-N	2005-06-0	R25-M0-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
13	-	1.12E+09	2005.06.0	R24-M1-N	2005-06-0	R24-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
14	-	1.12E+09	2005.06.0	R24-M1-N	2005-06-0	R24-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>
15	-	1.12E+09	2005.06.0	R20-M1-N	2005-06-0	R20-M1-N	RAS	KERNEL	INFO	generatin	E67	generating core.<*>

- HDFS log

LineId	Date	Time	Pid	Level	Compone	Content	EventId	EventTemplate
1	81109	203615	148	INFO	dfs.DataN	PacketRes	E10	PacketResponder <*> for block blk_<*> terminating
2	81109	203807	222	INFO	dfs.DataN	PacketRes	E10	PacketResponder <*> for block blk_<*> terminating
3	81109	204005	35	INFO	dfs.FSNan	BLOCK* N	E6	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*>:<*> is added to blk_<*> size <*>
4	81109	204015	308	INFO	dfs.DataN	PacketRes	E10	PacketResponder <*> for block blk_<*> terminating
5	81109	204106	329	INFO	dfs.DataN	PacketRes	E10	PacketResponder <*> for block blk_<*> terminating
6	81109	204132	26	INFO	dfs.FSNan	BLOCK* N	E6	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*>:<*> is added to blk_<*> size <*>
7	81109	204324	34	INFO	dfs.FSNan	BLOCK* N	E6	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*>:<*> is added to blk_<*> size <*>
8	81109	204453	34	INFO	dfs.FSNan	BLOCK* N	E6	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*>:<*> is added to blk_<*> size <*>
9	81109	204525	512	INFO	dfs.DataN	PacketRes	E10	PacketResponder <*> for block blk_<*> terminating
10	81109	204655	556	INFO	dfs.DataN	Received	E11	Received block blk_<*> of size <*> from /<*>
11	81109	204722	567	INFO	dfs.DataN	Received	E11	Received block blk_<*> of size <*> from /<*>
12	81109	204815	653	INFO	dfs.DataN	Receiving	E13	Receiving block blk_<*> src: /<*>:<*> dest: /<*>:<*>
13	81109	204842	663	INFO	dfs.DataN	Receiving	E13	Receiving block blk_<*> src: /<*>:<*> dest: /<*>:<*>
14	81109	204908	31	INFO	dfs.FSNan	BLOCK* N	E6	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*>:<*> is added to blk_<*> size <*>
15	81109	204925	673	INFO	dfs.DataN	Receiving	E13	Receiving block blk_<*> src: /<*>:<*> dest: /<*>:<*>
16	81109	205035	28	INFO	dfs.FSNan	BLOCK* N	E7	BLOCK* NameSystem.allocateBlock: /<*>/part-<*>. blk_<*>
17	81109	205056	710	INFO	dfs.DataN	PacketRes	E10	PacketResponder <*> for block blk_<*> terminating
18	81109	205157	752	INFO	dfs.DataN	Received	E11	Received block blk_<*> of size <*> from /<*>

- Application log

LineId	Time	Compone	Pid	Content	EventId	EventTemplate
1	20171223-	Step_LSC	30002312	onStandSt	E42	onStandStepChanged <*>
2	20171223-	Step_LSC	30002312	onExtend:	E39	onExtend:<*> <*> <*> <*>
3	20171223-	Step_Stan	30002312	onReceive	E41	onReceive action: android.intent.action.SCREEN_ON
4	20171223-	Step_LSC	30002312	processHa	E43	processHandleBroadcastAction action:android.intent.action.SCREEN_ON
5	20171223-	Step_Stan	30002312	flush sens	E12	flush sensor data
6	20171223-	Step_SPU	30002312	getToday	E22	getTodayTotalDetailSteps = <*>##<*>##<*>##<*>##<*>
7	20171223-	Step_SPU	30002312	setToday1	E58	setTodayTotalDetailSteps=<*>
8	20171223-	Step_LSC	30002312	onStandSt	E42	onStandStepChanged <*>
9	20171223-	Step_ExtS	30002312	calculateC	E4	calculateCaloriesWithCache totalCalories=<*>
10	20171223-	Step_ExtS	30002312	calculateA	E3	calculateAltitudeWithCache totalAltitude=<*>
11	20171223-	Step_Stan	30002312	REPORT :	E47	REPORT : <*> <*> <*> <*>
12	20171223-	Step_LSC	30002312	onExtend:	E39	onExtend:<*> <*> <*> <*>
13	20171223-	Step_LSC	30002312	onStandSt	E42	onStandStepChanged <*>
14	20171223-	Step_LSC	30002312	onStandSt	E42	onStandStepChanged <*>
15	20171223-	Step_LSC	30002312	onExtend:	E39	onExtend:<*> <*> <*> <*>

- HPC log

LineId	LogId	Node	Compone	State	Time	Flag	Content	EventId	EventTemplate
1	134681	node-246	unix.hw	state_cha	1.08E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
2	350766	node-109	unix.hw	state_cha	1.08E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
3	344518	node-246	unix.hw	state_cha	1.08E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
4	344448	node-153	unix.hw	state_cha	1.08E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
5	366633	node-200	unix.hw	state_cha	1.09E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
6	366463	node-122	unix.hw	state_cha	1.09E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
7	438190	node-228	unix.hw	state_cha	1.1E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
8	225111	node-10	unix.hw	state_cha	1.12E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
9	360778	node-130	unix.hw	state_cha	1.14E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
10	401569	node-169	unix.hw	state_cha	1.14E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
11	401855	node-187	unix.hw	state_cha	1.14E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
12	460773	node-199	unix.hw	state_cha	1.15E+09	1	Compone	E13	Component State Change: Component <*> is in the unavailable state (HWID=<*>)
13	2568643	node-70	action	start	1.07E+09	1	clusterAdi	E8	clusterAddMember (command <*>)
14	2570772	node-124	action	start	1.07E+09	1	clusterAdi	E8	clusterAddMember (command <*>)
15	2571927	node-28	action	start	1.07E+09	1	risBoot (c	E36	risBoot (command <*>)
16	2572286	node-17	action	start	1.07E+09	1	bootGenv	E6	bootGenvmunix (command <*>)
17	2575909	node-162	action	start	1.07E+09	1	boot (con	E4	boot (command <*>)

- Linux log

LineId	Month	Date	Time	Level	Compone	PID	Content	EventId	EventTemplate
1	Jun	14	15:16:01	combo	sshd(pam	19939	authentic	E16	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*>
2	Jun	14	15:16:02	combo	sshd(pam	19937	check pas	E27	check pass; user unknown
3	Jun	14	15:16:02	combo	sshd(pam	19937	authentic	E16	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*>
4	Jun	15	2:04:59	combo	sshd(pam	20882	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
5	Jun	15	2:04:59	combo	sshd(pam	20884	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
6	Jun	15	2:04:59	combo	sshd(pam	20883	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
7	Jun	15	2:04:59	combo	sshd(pam	20885	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
8	Jun	15	2:04:59	combo	sshd(pam	20886	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
9	Jun	15	2:04:59	combo	sshd(pam	20892	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
10	Jun	15	2:04:59	combo	sshd(pam	20893	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
11	Jun	15	2:04:59	combo	sshd(pam	20896	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
12	Jun	15	2:04:59	combo	sshd(pam	20897	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
13	Jun	15	2:04:59	combo	sshd(pam	20898	authentic	E18	authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=<*> user=root
14	Jun	15	4:06:18	combo	su(pam_u	21416	session op	E102	session opened for user <*> by (uid=<*>)
15	Jun	15	4:06:19	combo	su(pam_u	21416	session cl	E101	session closed for user <*>
16	Jun	15	4:06:20	combo	logrotate		ALERT exi	E8	ALERT exited abnormally with [1]
17	Jun	15	4:12:42	combo	su(pam_u	22644	session op	E102	session opened for user <*> by (uid=<*>)
18	Jun	15	4:12:43	combo	su(pam_u	22644	session cl	E101	session closed for user <*>

- MAC log

LineId	Month	Date	Time	User	Compone	PID	Address	Content	EventId	EventTemplate
1	Jul	1	9:00:55	calvisitor-	kernel	0		IOThunde	E252	IOThunderboltSwitch<<*>>(<*>)::listenerCallback - Thunderbolt HPD packet for route = <*> port = <*> unplug = <*>
2	Jul	1	9:01:05	calvisitor-	com.apple	43		Thermal p	E323	Thermal pressure state: <*> Memory pressure state: <*>
3	Jul	1	9:01:06	calvisitor-	QQ	10018		FA||Url||	E216	FA||Url||taskID[<*>] dealloc
4	Jul	1	9:02:26	calvisitor-	kernel	0		ARPT: 620	E128	ARPT: <*>.<*>: AirPort_<*>::syncPowerState: WWEN[enabled]
5	Jul	1	9:02:26	authorMa	kernel	0		ARPT: 620	E124	ARPT: <*>.<*>: AirPort_<*>::platformWoWEnable: WWEN[disable]
6	Jul	1	9:03:11	calvisitor-	mDNSRes	91		mDNS_De	E260	mDNS_DeregisterInterface: Frequent transitions for interface awdl0 (<*>)
7	Jul	1	9:03:13	calvisitor-	kernel	0		ARPT: 620	E135	ARPT: <*>.<*>: IOPMPowerSource Information: onSleep, SleepType: Normal Sleep, 'ExternalConnected': Yes, 'TimeRemaining': <*>,
8	Jul	1	9:04:33	calvisitor-	kernel	0		ARPT: 620	E149	ARPT: <*>.<*>: wl0: wl_update_tcpkeep_seq: Original Seq: <*>, Ack: <*>, Win size: <*>
9	Jul	1	9:04:33	authorMa	kernel	0		ARPT: 620	E131	ARPT: <*>.<*>: ARPT: Wake Reason: Wake on Scan offload
10	Jul	1	9:04:37	authorMa	symptoms	215		__73-[Net	E50	__<*>-[NetworkAnalyticsEngine observeValueForKeyPath:ofObject:change:context:]_block_invoke unexpected switch value <*>
11	Jul	1	9:12:20	authorMa	kernel	0		IO80211A	E58	<*>::setAwdlAutoMode Resuming AWDL
12	Jul	1	9:12:21	calvisitor-	symptoms	215		__73-[Net	E50	__<*>-[NetworkAnalyticsEngine observeValueForKeyPath:ofObject:change:context:]_block_invoke unexpected switch value <*>
13	Jul	1	9:18:16	calvisitor-	kernel	0		ARPT: 620	E140	ARPT: <*>.<*>: wl0: MDNS: <*> SRV Recs, <*> TXT Recs
14	Jul	1	9:19:03	calvisitor-	kernel	0		AppleCam	E120	AppleCamIn::systemWakeCall - messageType = <*>
15	Jul	1	9:19:03	authorMa	configd	53		setting ho	E311	setting hostname to "<*>"
16	Jul	1	9:19:13	calvisitor-	com.apple	258		com.apple	E189	com.apple.<*>: scheduler_evaluate_activity told me to run this job; however, but the start time isn't for <*> seconds. Ignoring.
17	Jul	1	9:21:57	authorMa	corecaptu	31174		CCIORepo	E175	CCIOReporterFormatter::addRegistryChildToChannelDictionary streams <*>
18	Jul	1	9:21:58	calvisitor-	com.apple	25654		[09:21:58.:	E13	[<*>:<*><*>.<*>] <<<< CRABS >>>> crabsFlumeHostAvailable: [<*>] Byte flume reports host available again.
19	Jul	1	9:22:02	calvisitor-	com.apple	258		com.apple	E189	com.apple.<*>: scheduler_evaluate_activity told me to run this job; however, but the start time isn't for <*> seconds. Ignoring.
20	Jul	1	9:22:25	calvisitor-	kernel	0		IO80211A	E58	<*>::setAwdlAutoMode Resuming AWDL

- OpenSSH log

LineId	Date	Day	Time	Compone	Pid	Content	EventId	EventTemplate
1	Dec	10	6:55:46	LabSZ	24200	reverse m	E27	reverse mapping checking getaddrinfo for <*> [<*>] failed - POSSIBLE BREAK-IN ATTEMPT!
2	Dec	10	6:55:46	LabSZ	24200	Invalid us	E13	Invalid user <*> from <*>
3	Dec	10	6:55:46	LabSZ	24200	input_use	E12	input_userauth_request: invalid user <*> [preauth]
4	Dec	10	6:55:46	LabSZ	24200	pam_unix	E21	pam_unix(sshd:auth): check pass; user unknown
5	Dec	10	6:55:46	LabSZ	24200	pam_unix	E19	pam_unix(sshd:auth): authentication failure; logname= uid=<*> euid=<*> tty=ssh ruser= rhost=<*>
6	Dec	10	6:55:48	LabSZ	24200	Failed pas	E10	Failed password for invalid user <*> from <*> port <*> ssh2
7	Dec	10	6:55:48	LabSZ	24200	Connectic	E2	Connection closed by <*> [preauth]
8	Dec	10	7:02:47	LabSZ	24203	Connectic	E2	Connection closed by <*> [preauth]
9	Dec	10	7:07:38	LabSZ	24206	Invalid us	E13	Invalid user <*> from <*>
10	Dec	10	7:07:38	LabSZ	24206	input_use	E12	input_userauth_request: invalid user <*> [preauth]
11	Dec	10	7:07:38	LabSZ	24206	pam_unix	E21	pam_unix(sshd:auth): check pass; user unknown
12	Dec	10	7:07:38	LabSZ	24206	pam_unix	E19	pam_unix(sshd:auth): authentication failure; logname= uid=<*> euid=<*> tty=ssh ruser= rhost=<*>
13	Dec	10	7:07:45	LabSZ	24206	Failed pas	E10	Failed password for invalid user <*> from <*> port <*> ssh2
14	Dec	10	7:07:45	LabSZ	24206	Received	E24	Received disconnect from <*>: <*>: Bye Bye [preauth]
15	Dec	10	7:08:28	LabSZ	24208	reverse m	E27	reverse mapping checking getaddrinfo for <*> [<*>] failed - POSSIBLE BREAK-IN ATTEMPT!
16	Dec	10	7:08:28	LabSZ	24208	Invalid us	E13	Invalid user <*> from <*>
17	Dec	10	7:08:28	LabSZ	24208	input_use	E12	input_userauth_request: invalid user <*> [preauth]
18	Dec	10	7:08:28	LabSZ	24208	pam_unix	E21	pam_unix(sshd:auth): check pass; user unknown
19	Dec	10	7:08:28	LabSZ	24208	pam_unix	E19	pam_unix(sshd:auth): authentication failure; logname= uid=<*> euid=<*> tty=ssh ruser= rhost=<*>

- OpenStack log

LineId	Logrecord	Date	Time	Pid	Level	Compone	ADDR	Content	EventId	EventTemplate
1	nova-api.l	########	00:00.0	25746	INFO	nova.osap	req-38101	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
2	nova-api.l	########	00:00.3	25746	INFO	nova.osap	req-9bc36	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
3	nova-api.l	########	00:01.6	25746	INFO	nova.osap	req-55db2	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
4	nova-api.l	########	00:01.8	25746	INFO	nova.osap	req-2a3dc	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
5	nova-api.l	########	00:03.1	25746	INFO	nova.osap	req-939eb	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
6	nova-api.l	########	00:03.4	25746	INFO	nova.osap	req-b6a4f	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
7	nova-com	########	00:04.5	2931	INFO	nova.com	req-3ea40	[instance: <*>	E22	[instance: <*>] VM Started (Lifecycle Event)
8	nova-com	########	00:04.6	2931	INFO	nova.com	req-3ea40	[instance: <*>	E20	[instance: <*>] VM Paused (Lifecycle Event)
9	nova-com	########	00:04.7	2931	INFO	nova.com	req-3ea40	[instance: <*>	E7	[instance: <*>] During sync_power_state the instance has a pending task (spawning). Skip.
10	nova-api.l	########	00:04.8	25746	INFO	nova.osap	req-bbfc3	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
11	nova-api.l	########	00:05.1	25746	INFO	nova.osap	req-31826	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
12	nova-com	########	00:05.2	2931	INFO	nova.virt.l	req-addc1	image 067	E34	image <*> at (<*>): checking
13	nova-com	########	00:05.2	2931	INFO	nova.virt.l	req-addc1	image 067	E35	image <*> at (<*>): in use: on this node <*> local, <*> on other nodes sharing this instance storage
14	nova-com	########	00:05.4	2931	INFO	nova.virt.l	req-addc1	Active bas	E27	Active base files: <*>
15	nova-api.l	########	00:06.3	25746	INFO	nova.osap	req-7160b	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
16	nova-api.l	########	00:06.6	25746	INFO	nova.osap	req-e46f1	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
17	nova-api.l	########	00:07.9	25746	INFO	nova.osap	req-546e2	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
18	nova-api.l	########	00:08.1	25746	INFO	nova.osap	req-e2c35	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>
19	nova-api.l	########	00:09.4	25746	INFO	nova.osap	req-ce9c8	10.11.10.1	E25	<*> "GET <*>" status: <*> len: <*> time: <*>,<*>

- Proxifier

LineId	Time	Program	Content	EventId	EventTemplate
1	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
2	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
3	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
4	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
5	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
6	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
7	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
8	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
9	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
10	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
11	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
12	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
13	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
14	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
15	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS
16	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
17	10.30 16:4	chrome.e	proxy.cse	E8	<*> close, <*> bytes<*>sent, <*> bytes<*>received, lifetime <*>
18	10.30 16:4	chrome.e	proxy.cse	E2	<*>:<*> open through proxy <*>:<*> HTTPS

- Spark

LineId	Date	Time	Level	Compone	Content	EventId	EventTemplate
1	17/06/09	20:10:40	INFO	executor.(	Registere	E22	Registered signal handlers for [TERM, HUP, INT]
3	17/06/09	20:10:40	INFO	spark.Secu	Changing	E5	Changing view acls to: <*>
3	17/06/09	20:10:40	INFO	spark.Secu	Changing	E4	Changing modify acls to: <*>
4	17/06/09	20:10:40	INFO	spark.Secu	SecurityM	E26	SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(yarn, curi); users with modify permissions: Set(yarn, curi)
5	17/06/09	20:10:41	INFO	spark.Secu	Changing	E5	Changing view acls to: <*>
6	17/06/09	20:10:41	INFO	spark.Secu	Changing	E4	Changing modify acls to: <*>
7	17/06/09	20:10:41	INFO	spark.Secu	SecurityM	E26	SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(yarn, curi); users with modify permissions: Set(yarn, curi)
8	17/06/09	20:10:41	INFO	slf4j.Slf4jL	Slf4jLogge	E28	Slf4jLogger started
9	17/06/09	20:10:41	INFO	Remoting	Starting re	E31	Starting remoting
10	17/06/09	20:10:41	INFO	Remoting	Remoting	E23	Remoting started; listening on addresses :[akka.tcp://<*>]
11	17/06/09	20:10:41	INFO	util.Utils	Successfu	E34	Successfully started service 'sparkExecutorActorSystem' on port <*>.
12	17/06/09	20:10:41	INFO	storage.Di	Created lc	E7	Created local directory at <*>
13	17/06/09	20:10:41	INFO	storage.M	MemoryS	E18	MemoryStore started with capacity <*> GB
14	17/06/09	20:10:42	INFO	executor.(	Connectin	E6	Connecting to driver: spark://<*>
15	17/06/09	20:10:42	INFO	executor.(	Successfu	E32	Successfully registered with driver
16	17/06/09	20:10:42	INFO	executor.I	Starting e	E30	Starting executor ID <*> on host <*>
17	17/06/09	20:10:42	INFO	util.Utils	Successfu	E33	Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port <*>.
18	17/06/09	20:10:42	INFO	netty.Net	Server cre	E27	Server created on <*>
19	17/06/09	20:10:42	INFO	storage.Bl	Trying to r	E36	Trying to register BlockManager

- ThunderBird

LineId	Label	Timestam	Date	User	Month	Day	Time	Location	Compone	PID	Content	EventId	EventTemplate
1	-	1.13E+09	2005.11.09	dn228	Nov	9	12:01:01	dn228/dn	crond(pan	2915	session cl	E117	session closed for user root
2	-	1.13E+09	2005.11.09	dn228	Nov	9	12:01:01	dn228/dn	crond(pan	2915	session op	E118	session opened for user root by (uid=0)
3	-	1.13E+09	2005.11.09	dn228	Nov	9	12:01:01	dn228/dn	crond	2916	(root) CMI	E3	(root) CMD (run-parts /etc/cron.hourly)
4	-	1.13E+09	2005.11.09	dn261	Nov	9	12:01:01	dn261/dn	crond(pan	2907	session cl	E117	session closed for user root
5	-	1.13E+09	2005.11.09	dn261	Nov	9	12:01:01	dn261/dn	crond(pan	2907	session op	E118	session opened for user root by (uid=0)
6	-	1.13E+09	2005.11.09	dn261	Nov	9	12:01:01	dn261/dn	crond	2908	(root) CMI	E3	(root) CMD (run-parts /etc/cron.hourly)
7	-	1.13E+09	2005.11.09	dn3	Nov	9	12:01:01	dn3/dn3	crond(pan	2907	session cl	E117	session closed for user root
8	-	1.13E+09	2005.11.09	dn3	Nov	9	12:01:01	dn3/dn3	crond(pan	2907	session op	E118	session opened for user root by (uid=0)
9	-	1.13E+09	2005.11.09	dn3	Nov	9	12:01:01	dn3/dn3	crond	2908	(root) CMI	E3	(root) CMD (run-parts /etc/cron.hourly)
10	-	1.13E+09	2005.11.09	dn596	Nov	9	12:01:01	dn596/dn	crond(pan	2727	session cl	E117	session closed for user root
11	-	1.13E+09	2005.11.09	dn596	Nov	9	12:01:01	dn596/dn	crond(pan	2727	session op	E118	session opened for user root by (uid=0)
12	-	1.13E+09	2005.11.09	dn596	Nov	9	12:01:01	dn596/dn	crond	2728	(root) CMI	E3	(root) CMD (run-parts /etc/cron.hourly)
13	-	1.13E+09	2005.11.09	dn700	Nov	9	12:01:01	dn700/dn	crond(pan	2912	session cl	E117	session closed for user root
14	-	1.13E+09	2005.11.09	dn700	Nov	9	12:01:01	dn700/dn	crond(pan	2912	session op	E118	session opened for user root by (uid=0)
15	-	1.13E+09	2005.11.09	dn700	Nov	9	12:01:01	dn700/dn	crond	2913	(root) CMI	E3	(root) CMD (run-parts /etc/cron.hourly)
16	-	1.13E+09	2005.11.09	dn73	Nov	9	12:01:01	dn73/dn7	crond(pan	2917	session cl	E117	session closed for user root
17	-	1.13E+09	2005.11.09	dn73	Nov	9	12:01:01	dn73/dn7	crond(pan	2917	session op	E118	session opened for user root by (uid=0)
18	-	1.13E+09	2005.11.09	dn73	Nov	9	12:01:01	dn73/dn7	crond	2918	(root) CMI	E3	(root) CMD (run-parts /etc/cron.hourly)
19	-	1.13E+09	2005.11.09	dn731	Nov	9	12:01:01	dn731/dn	crond(pan	2916	session cl	E117	session closed for user root

- Windows

LineId	Date	Time	Level	Compone	Content	EventId	EventTemplate
1	########	4:30:30	Info	CBS	Loaded Se	E23	Loaded Servicing Stack <*> with Core: <*>\cbscore.dll
2	########	4:30:31	Info	CSI	00000001(	E13	<*>@<*>/<*>:<*><*>:<*>,<*> WcpInitialize (wcp.dll version <*>) called (stack @<*>)
3	########	4:30:31	Info	CSI	00000002(	E13	<*>@<*>/<*>:<*><*>:<*>,<*> WcpInitialize (wcp.dll version <*>) called (stack @<*>)
4	########	4:30:31	Info	CSI	00000003(	E13	<*>@<*>/<*>:<*><*>:<*>,<*> WcpInitialize (wcp.dll version <*>) called (stack @<*>)
5	########	4:30:31	Info	CBS	Ending Tru	E17	Ending TrustedInstaller initialization.
6	########	4:30:31	Info	CBS	Starting Th	E44	Starting the TrustedInstaller main loop.
7	########	4:30:31	Info	CBS	TrustedIn	E48	TrustedInstaller service starts successfully.
8	########	4:30:31	Info	CBS	SQM: Initi	E40	SQM: Initializing online with Windows opt-in: False
9	########	4:30:31	Info	CBS	SQM: Clea	E37	SQM: Cleaning up report files older than <*> days.
10	########	4:30:31	Info	CBS	SQM: Req	E42	SQM: Requesting upload of all unsent reports.
11	########	4:30:31	Info	CBS	SQM: Faile	E39	SQM: Failed to start upload with file pattern: <*>, flags: <*> [HRESULT = <*> - E_FAIL]
12	########	4:30:31	Info	CBS	SQM: Faile	E38	SQM: Failed to start standard sample upload. [HRESULT = <*> - E_FAIL]
13	########	4:30:31	Info	CBS	SQM: Que	E41	SQM: Queued <*> file(s) for upload with pattern: <*>, flags: <*>
14	########	4:30:31	Info	CBS	SQM: War	E43	SQM: Warning: Failed to upload all unsent reports. [HRESULT = <*> - E_FAIL]
15	########	4:30:31	Info	CBS	No startup	E25	No startup processing required, TrustedInstaller service was not set as autostart, or else a reboot is still pending.
16	########	4:30:31	Info	CBS	NonStart:	E26	NonStart: Checking to ensure startup processing was not required.
17	########	4:30:31	Info	CSI	00000004	E5	<*> IAdvancedInstallerAwareStore_ResolvePendingTransactions (call <*>) (flags = <*>, progress = <*>, phase = <*>, pdwDisposition = @<*>
18	########	4:30:31	Info	CSI	00000005	E2	<*> Creating NT transaction (seq <*>), objectname [<*>]"(null)"
19	########	4:30:31	Info	CSI	00000006	E1	<*> Created NT transaction (seq <*>) result <*>, handle @<*>

References

[1] J. H. Andrews: \Theory and practice of log file analysis." Technical Report 524, Department of Computer Science, University of Western Ontario, May 1998.

[2] J. H. Andrews: \Testing using log file analysis: tools, methods, and issues." Proc. 13 th IEEE International Conference on Automated Software Engineering, Oct. 1998, pp. 157-166.

[3] J. H. Andrews: \A Framework for Log File Analysis." http://citeseer.nj.nec.com/159829.html

[4] J. H. Andrews, Y. Zhang: \Broad-spectrum studies of log file analysis." International Conference on Software Engineering, pages 105-114, 2000

[5] J. H. Andrews: \Testing using Log File Analysis: Tools, Methods, and Issues." available at http://citeseer.nj.nec.com

[6] M. Guzdial, P. Santos, A. Badre, S. Hudson, M. Gray: \Analyzing and visualizing log files: A computational science of usability." Presented at HCI Consortium Workshop, 1994.

[7] M. J. Guzdial: \Deriving software usage patterns from log files." Georgia Institute of Technology. GVU Center Technical Report. Report #93-41. 1993.

[8] Tec-Ed, Inc.: \Assessing Web Site Usability from Server Log Files White Paper." http://citeseer.nj.nec.com/290488.html

Terms & Conditions

Interns shall be solely responsible for all its acts and omissions under this program. Interns will comply at all times with all applicable laws. Interns shall not use Cognizant's name, logo and trademark in any promotional materials or other communications with third parties without the prior written consent of Cognizant. Any materials used by interns in relation to program will not infringe the copyrights, trademarks, patents, trade secrets or other intellectual property rights, privacy or similar rights of any person or entity. Interns agrees not to post, draw, make, display any content that is threatening, libelous, obscene, defamatory, abusive, pornographic, or advocates/encourages any conduct that could constitute a criminal offence or give rise to any civil liability. Cognizant its associates' personal details including but not limited to name, address, contact number shall not be shared or forwarded to any third party, without prior written consent of Cognizant, its associates. All intellectual property provided by Cognizant as part of program shall be owned exclusively by Cognizant. Intern shall indemnify, defend and indemnify Cognizant its associates, officers, directors from and against any claims, demands, loss, damage, liability, causes of action, judgments, or costs and expenses of every nature (including attorney's fees and expenses) incurred by Cognizant based on any claim that any breach of terms and conditions of this program.