User-centric Visualization of Data Provenance

A thesis

Submitted in fulfilment

of the requirements for the degree

of

Master of Cyber Security

at

The University of Waikato

by

JEFFERY GARAE

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

Cyber Security Researchers Of Waikato (CROW)

Department of Computer Science

Hamilton, New Zealand

February 20, 2015

ABSTRACT

The need to understand and track files (and inherently, data) in cloud computing systems is in high demand. Over the past years, the use of logs and data representation using graphs have become the main method for tracking and relating information to the cloud users. While it is still in use, tracking and relating information with 'Data Provenance' (i.e. series of chronicles and the derivation history of data on meta-data) is the new trend for cloud users. However, there is still much room for improving representation of data activities in cloud systems for end-users.

In this thesis, we propose "UVisP (User-centric Visualization of Data Provenance with Gestalt)", a novel user-centric visualization technique for data provenance. This technique aims to facilitate the missing link between data movements in cloud computing environments and the end-users' uncertain queries over their files' security and life cycle within cloud systems.

The proof of concept for the UVisP technique integrates D3 (an open-source visualization API) with Gestalts' theory of perception to provide a range of user-centric visualizations. UVisP allows users to transform and visualize provenance (logs) with implicit prior knowledge of 'Gestalts' theory of perception.' We presented the initial development of the UVisP technique and our results show that the integration of Gestalt and the existence of '*perceptual key(s)*' in provenance visualization allows end-users to enhance their visualizing capabilities, extract useful knowledge and understand the visualizations better. This technique also enables end-users to develop certain methods and preferences when sighting different visualizations. For example, having the prior knowledge of Gestalt's theory of perception and integrated with the types of visualizations offers the user-centric experience when using different visualizations. We also present significant future work that will help profile new user-centric visualizations for cloud users.

ACKNOWLEDGEMENTS

Being part of the first and new Cyber Security Lab (CROW) at the University of Waikato, is an honour, and working towards achieving and becoming the first ever '*Master of Cyber Security*' student could not have been done so effectively and enjoyably without the help and encouragement of many individuals.

Firstly, I would like to extend my special thanks to my supervisor, Dr. Ryan Ko, for inspiring me with all those knowledgeable research methods, and guiding me during the course of this thesis. His research expertise have allowed me to enhance my knowledge and skills in the field of Cyber Security.

I am also grateful to everyone that has assisted me during the course of this thesis – especially, Alan Tan, Mark Will, Baden Delamore, Mohammad B Taha and the CROW team for the continuous discussions and help. All those hints, point-of-views, guides and weekly updates have helped me through the year of study.

I would also like to thank and acknowledge the D3.org team, for responding well to our D3 queries, especially M.Bostock for his contribution towards D3.js and allowing us to reuse his codes.

Finally, I would like to thank my wife - Camillia, my son - Caleb Garae Liu and family for their continuous support, patience and help throughout the course of this thesis.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

API	Application Program Interface
CSS	Cascading Style Sheets
CSV	Comma-separated values
D3	Data-Driven Documents
DB	Database
GUI	Graphical User Interface
G.S-V	Gestalt - Similarity
G.P-V	Gestalt – Proximity
G.Pr-V	Gestalt - Pragnanz
G.CF-V	Gestalt – Common Fate
HDP	Hortonworks Data Platform
HDFS	Hadoop Distribution File System
HTML	HyperText Markup Language
IT	Information Technology
JSON	JavaScript Object Notation
RSYSLOG	Rocket-fast System for Log Processing
SCP	Secure copy
SVG	Scalable Vector Graphics
PHP	A server-side scripting Language designed for web development but also used as a general purpose programming language
PM	Physical Machine
PROGGER	Provenance Logger

U-C	User-Centric
UVisP	User-Centric Visualization of Data Provenance with Gestalt
VM	Virtual Machine

CHAPTER 1 INTRODUCTION

1.1 MOTIVATION

The complex changes on networks raises the question of how to secure them. Before securing these networks and systems, one needs to "know" or "see" what is going on in these systems. A proposed solution to this is visualization. With the state-of-the-art 'data visualization' techniques are the means of providing a visible solution to knowing what is going on in these systems. It also helps users to actively identify useful patterns and gens within the data when picturing different visualizations [11].

In the area of security, data provenance is a burgeoning field for data security particularly for knowing what goes on with the data. Data provenance enables the ability of keeping track of file movements over the network and on the Internet. However in the area of provenance, visualization is not a well-studied topic. Most visualizations in this area of study are for exploratory purposes. Therefore the need to bring visualization into provenance at the end-users perspective is a high demand. Such visualizations depict the history of their data when needed.

The main objective of this research is to explore potential methods and techniques of visualizing provenance to benefit the end-users of the cloud network. Based on this research objective, evaluations on past and existing works on data and user-centric visualization will be carried out. This thesis presents a user-centric visualization technique (UVisP) centred on visualizing provenance. It incorporates the concepts of Gestalt theory of perception [29] to provide a user-centric approach. The aim of UVisP is to transforms data into visualization for end-users to identify useful patterns, gens, trends, and relationships by observing activities of interested files over the network in a given time. Such observations leads to new insights which will be discussed in Chapters 6, 7 and 8 of this thesis.

1.2 THESIS OBJECTIVES

Based on the main research objective, a set of sub objectives are identified. These are:

- To understand relevant user-centric theories and frameworks applied to visualization.
- To empower users to understand provenance elements and granularities of systems (application, virtual machine (VM), physical machine (PM), and cloud) by applying visualization theories
- To propose a novel technique to analyse existing visualization which empowers users to interpret their data's provenance elements and security throughout their data life cycle.

1.3 SCOPE

The focus of this thesis is on visualizing provenance at granularities. This enables end-users with an opportunity to view data over the local and cloud environment. Based on the time frame of this thesis, we will only look at visualizing provenance for the purpose of file tracking and movements:

- Within a local machine
- Across two VMs, and
- From a physical machine to a VM.

In addition, this thesis emphasises on the application of 'Gestalt Theory of perception.' These inclusion of Gestalt theory will create that user-centric relationship between the end-users and the UVisP interface. It will also provide the background knowledge to the end-users of how UVisP intends to deliver its visualizations to the end-users. Visualization patterns and organisations are some of the background knowledge inserted into end-users prior to visualizing with UVisP.

All datasets will be generated and collected using Progger (Provenance Logger) [5].

This thesis will not address visualization for:

- Tracking virtual-physical mapping, and vice versa.
- Detecting data leakage threats

Overall, this thesis will focus on the user-centric aspect of visualizing provenance to benefit end-users.

1.4 KEY CONTRIBUTIONS

The main contributions of this thesis are as follows:

- We propose a novel user-centric visualization technique for visualizing provenance. Our technique is based on an UVisP (User-centric Visualization of Data Provenance with Gestalt) technique for visualizing provenance at the users' level of understanding and knowledge.
- We implement our novel technique by integrating D3, an existing visualization API with the Gestalt theory of perception. To the best of our knowledge, this is the first time, such user-centric technique is proposed.
- We evaluate our methodology using random users of provenance visualization and found positive impacts on how end-users extract useful knowledge and information from our visualizations.
- We introduce and define "perceptual key." A core aspect to identifying useful patterns and information indicators in a user-centric provenance visualization.

1.5 THESIS OUTLINE

This thesis is organized as follows:

Chapter 2 covers the new terms, definitions and evaluates past and existing works in this area of study. For this it highlights existing visualization gaps and limitations to this field.

Chapter 3 describes the 'user-centric visualization' technique (proof of concept). It also provides the design, UVisP Flowchart, and components. And finally it provides the UVisP's user-centric (U-C) technique that will be used to obtain the final visualizations.

Chapter 4 provides a detail description of the proposed technique and how it is implemented. For this it describes the UVisP implementation phases and their distinctive tasks. It also describes the user-testing steps of execution and how the testing will be carried out.

Chapter 5 discusses the user-centric visualization methods and technique. It also analyses the findings from the datasets provided using a given visualization. This chapter also describes relationship between Gestalt and the UVisP visualizations.

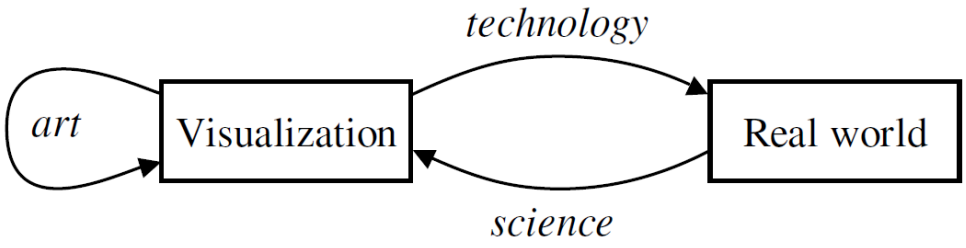
Chapter 6 concludes the thesis. For this it summaries all the methodology and techniques used to achieving user-centric visualizations. Furthermore, it states the future works.

CHAPTER 2 BACKGROUND AND RELATED WORK

This section begins by providing a background on visualization research and defining some commonly used terms in this thesis. It then evaluates past and existing works within the field of visualization, user centricity, data provenance and Gestalt theory of perception. Based on the evaluation, existing research gaps, limitations and findings are analysed and presented.

2.1 THE VISUALIZATION DISCOVERY PROCESS

Visualization is a common method for analysing, relating and understanding scientific datasets. Previous research show that visualization exists for several reasons: technology, art and for an empirical science [19]. Different visualizations target specific user groups for specific goals, needs, environment and time [12]. For example, most visualization gives the users the flexibility of customizing their displays and visual view in an understandable way [15] [10] [19]. Figure 2.1 states that visual views could be for the purposes of technological effectiveness and efficiency and it could also be for the purpose of artistically elegance and beauty. In a scientific point of view, generic laws are the users' view.



Src: http://peterahall.com/mapping/Van%20Wijk,%20J.pdf

Figure 2.1. Views on Visualization [19].

In order to acquire these user views, one must understand the visualization discovery process as presented in Figure 2.2.

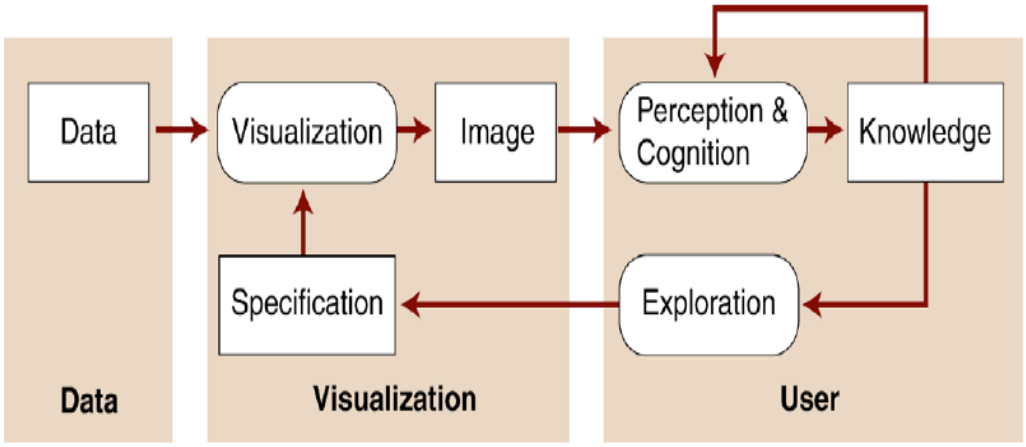


Figure 2.2. The visualization discovery process [10].

(*Data encompasses the range form a single bit, to time-varying 3D tensor fields, to multi-model data sources requiring alignment, registration, and fusion, and to non-spatial information sources integrating broad areas of human knowledge) [10].*

The 'visualization discovery process' in Figure 2.2 shows a simplified process of how raw data is being inserted into a visualization system then transformed into knowledge [10] [36]. Users have the flexibility to adjust specifications of the visualization according to how they perceive the final visualization output. The output relies on human perception and sense [10] and therefore producing specific informative visualizations.

Informative visualizations are valuable. For instance, in the field of data provenance, visualization can be a means of parsing and understanding a range of datasets especially large datasets [43]. It facilities a methodical way of not only understanding large data sets, but allowing a coherent view for the users of the visualization.

2.2 Definitions of Research Terms

Before we explore the landscape of user-centric visualization and related works, let us define a few terms commonly used in this thesis.

User-Centric

User-Centric (U-C) is defined in [6] as "contextual pieces of information as the user perceives them." In the context of this research, user-centric is a term given to an end user driven environment whereby the users possess the centre of attention on products and services, i.e. requirements, preferences and access controls on products and services varies according to the user hierarchies. For example, age groups and genders.

Visualization

The Oxford English dictionary defines visualisation as "the power of forming a mental picture or vision of something not actually present to the sight; a picture thus formed [77]." Visualization in this research is defined as an effective technique of creating imagery, diagrams and animations to draw insights on abstract and concrete ideas from data.

Data Visualization

Data visualization is a process of mapping data to visuals, i.e. data in, visual representation out.

Data Provenance

Data Provenance is defined in [41] as "a meta-data describing the derivation history of data." Data Provenance is defined in [77] as "a record of origin and evolution of data." Data Provenance in the context of this research is defined as "series of chronicles and the derivation history of data on meta-data".

End-User

End-user in [37] is defined as "a role that a software program hardware device is designed for." The term is based on the idea that the end goal of the software or hardware product is to be useful to the users. For this research, end-user is a non-technical personal who interacts with the hardware or software directly after it has been fully developed, marketed and installed. While end-users can mean companies

and institutions, it is also the person who keeps calling the 'IT Guys' with queries to why the product or service is not working correctly. In this research end-users and users are the same person.

Touched

'Touched' in the context of this thesis refers to any actions caused on a file either in the physical or cloud environment. It can mean:

- Viewing the file,
- Accessing the file,
- Moving the file from one location to another, and
- Having authorized access or rights to the files.

Unauthorized

'Unauthorized' users refers to any users of the physical or cloud environment that are not allowed to view or 'touch' (see above) certain files.

Dataset

Dataset in this research refers to the data being collected by Progger.

Perceptual Key

'Perceptual key' is defined as the act of recognizing the very first key-identifier, eye-catching, common and recurring pattern(s) in a given visualization.

2.3 Evaluation of Related Work

As shown previously in Figure 2.2, users are the active component of the visualization discovery process. They process and transform raw data into knowledge from the visualizations.

A vast amount of work concentrating on visualization and security specifically targeting IT expects who wishes to analyse data and produce reports out from the findings. However, less work in the field of "visualizing data provenance" for security purposes to address the end-users needs. Visualizing data provenance can help on to understand the life cycle of his/her data and observe potential security breaches.

There are several types of related works: Firstly, related works regarding visualization in general and visualization of provenance. Secondly, related works regarding user-centric visualization. Thirdly, related works regarding the Gestalt theory of perception, and finally, related works regarding how Progger works and presents information to the users.

During the research we also explored the storage and processing of logs for visualization, and the relationship between Gestalt and the visualization APIs. These will be discussed in Chapters 3 and 4.

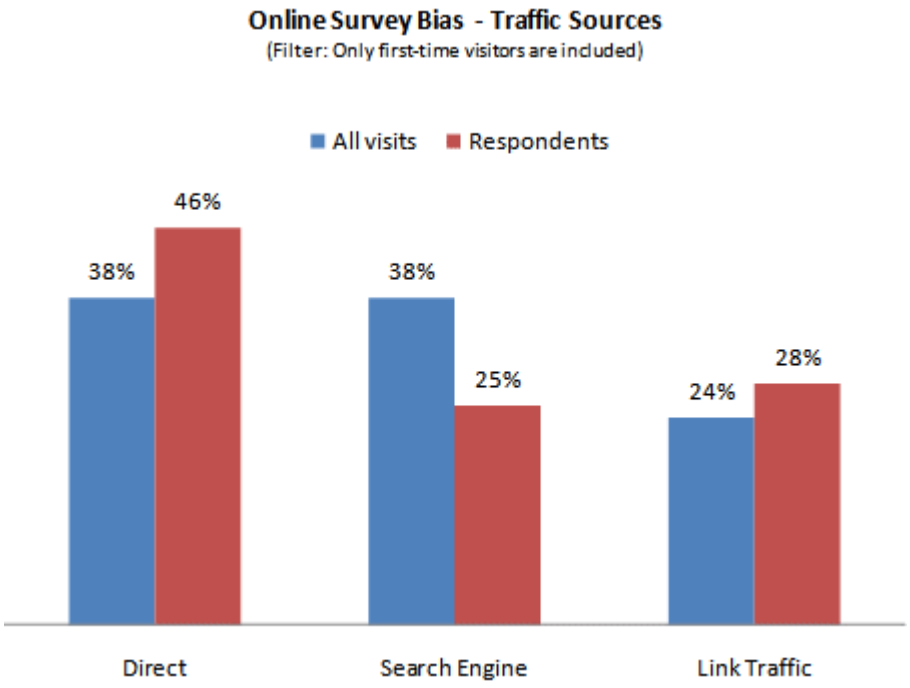
2.3.1 Visualization in General

We will now review some contemporary exploratory visualization techniques. We begin with setting the foundation of visualization by addressing the general use of line/bar graphs, pie charts, donut charts and heat maps to explore and analyse data through visualization. It has been the preferable solution for most researchers in the past and up till today [51] [67] [69].

These types of visualization have paved a way forward to new and intractable visualizations which will be discussed later in this chapter. For example, Figure 2.3 shows a bar graph of traffic resources based on an online survey [68]. It measures all visits to the online survey bias in percentage form and delivers the results in a form of a bar graph for users to observe.

The use of bar graph, line graphs and other popular graphs has been around in the research fields for centuries, and centered towards research purposes. Their

existence came about as a better way to display data in a visual representation than using tables [27]. However, to date, the presence of visualization in the research fields has attracted researchers to adopt graphs, pie-charts, and other common graphs into the visualizations to address user interactions and a means of providing exploratory information.



Src: http://www.theartofwebanalytics.com/

Figure 2.3. An example of a bar graph to illustrate an online survey bias – Traffic Sources [68].

2.3.1.1 Tag Cloud Visualization

The use of a 'tag cloud' for visualization is another example which came around to replace the use of list. This type of visualization is simple and easy to understand however, it is not user-centric in a sense that it does not produce analytic outputs apart from identifying the 'order of importance' and/or 'giving a count on the number of words appearing the most number of time.' Moreover, this type of visualization is still inconclusive within the research fields since there is still no prove on how effective it is. [67] [75]. Figure 2.4 shows an example of a tag cloud visualization which shows a visualization of data visualization and its subgroups.

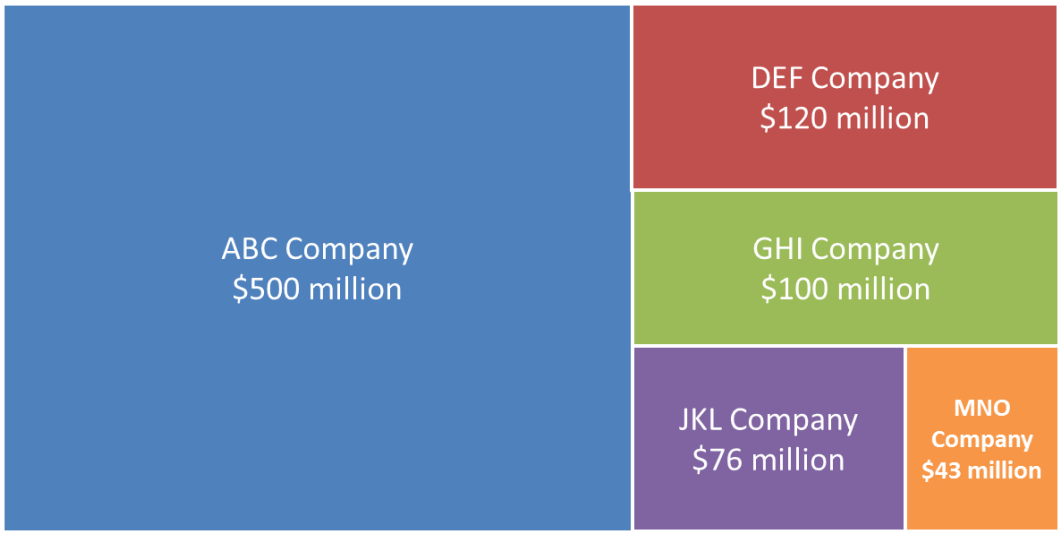
Src: http://www.neoformix.com/archive.html

Figure 2.4. An example of a tag cloud visualization of Data visualization subgroups [51].

2.3.1.2 TreeMaps

'Treemaps' or 'Dynamic Squarified Treemaps' by Bhulai *et al.* [75], is another type of visualization introduced to solve the representation issues tied to the lists and tag clouds. The size of the squares in the dynamic squarified treemaps represents the magnitude or importance of an item. One can argue that this is user-centric and that it provides a reasonable amount of information to the users. However, as data grows, scalability can become an issue with this type of visualization because of proliferation of tiny and large datasets requiring human-supervised analysis [67] [69]. For example, the ratio in size, of the largest square in the 'dynamic squarified treemap' to the smallest square can become small to a point where users of the visualization cannot read and analyse the treemap visualization. Figure 2.5 illustrates an example of a 'Dynamic Squarified Treemap' visualization based on company income. As shown in Figure 2.5, large datasets are visible. However take a new company, 'X Company' for example, has only $1 million for its income. To visualize 'X Company' along with all other existing company will be a user-centric issue, since the ratio of the largest company to the 'X Company' is so big that

visualizing it on the existing Dynamic Squarified Treemap' will be difficult in terms of visibility.



Src: http://www.thinkoutsidetheslide.com/wp-content/uploads/2013/09/treemap.png

Figure 2.5. An example of a Dynamic Squarified Treemap [74].

2.3.1.3 Circular Pattern Visualization

The 'Circular Pattern' representation visualization is another form that addresses large datasets. The so-called "AnalyticaR ([Unpublished work] -a visualization technique for data exploratory)" helps users to view data with a reasonable amount of space without much effort [5]. Similarly, in [18], visualization of large datasets are represented using circle segments for 'data exploratory' purposes. The basic idea behind representing large datasets using circle segments is to display all data dimensions as segments of the circle, i.e. data is viewed in a reasonable amount of space without much effort.

Other techniques aimed at visualizing very large amount of data is using 'recursive patterns' [62]. This technique is specifically for multidimensional data and it is built using generic recursive scheme. Due to its ability to build visualization using generic recursive schemes, it requires less resources, time, and provides better multidimensional data visualization. However, with tiny or small datasets, this technique is not a feasible user-centric solution for visualization.

However, with the existing examples of visualization stated in Figure 2.6 and Figure 2.8, the emphases are exploring and analysing data for better understanding. There is less focus on who the targeted audiences are. With different levels of visual evaluations by targeted audiences, end-users may struggle to understand such visualizations. Visualizations targeted for end-users should be user-friendly and the presentation of information has to be at their degree of understanding, i.e. the time taken for an end-user to see and process the visualization to obtain knowledge should be in seconds.

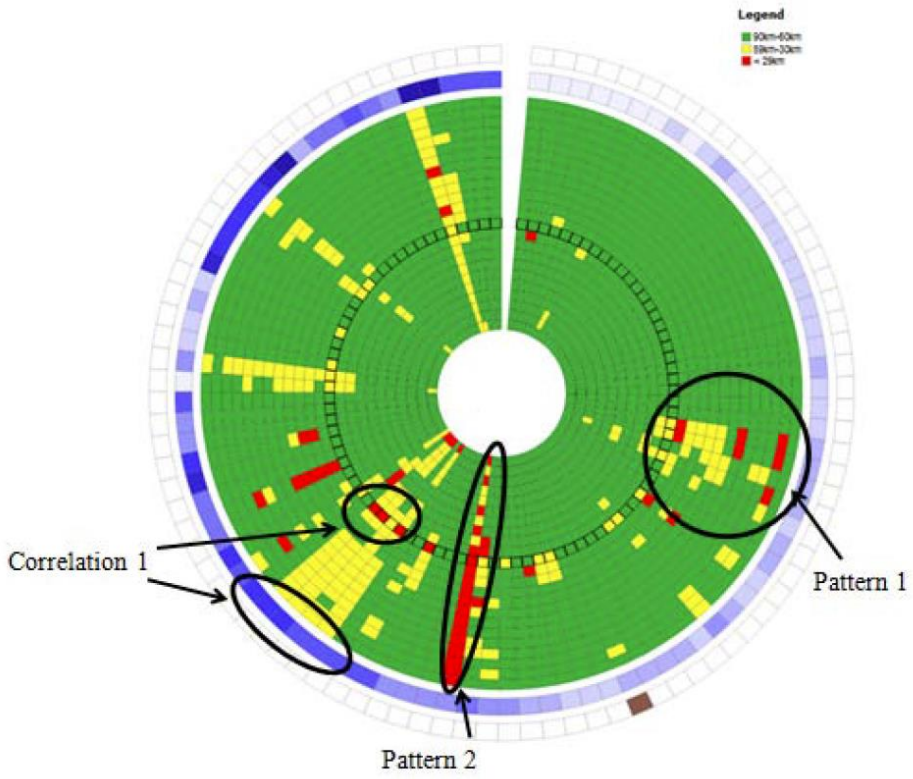
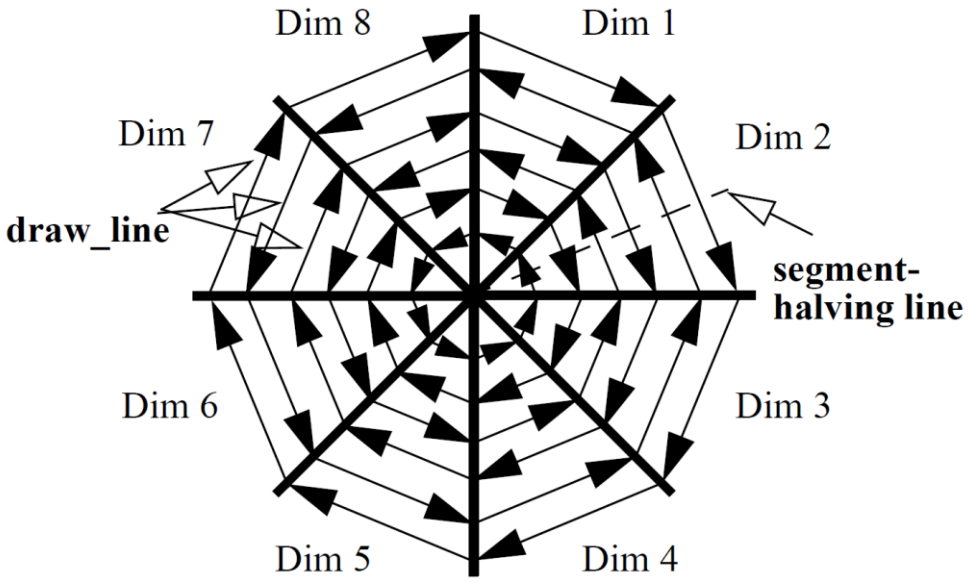
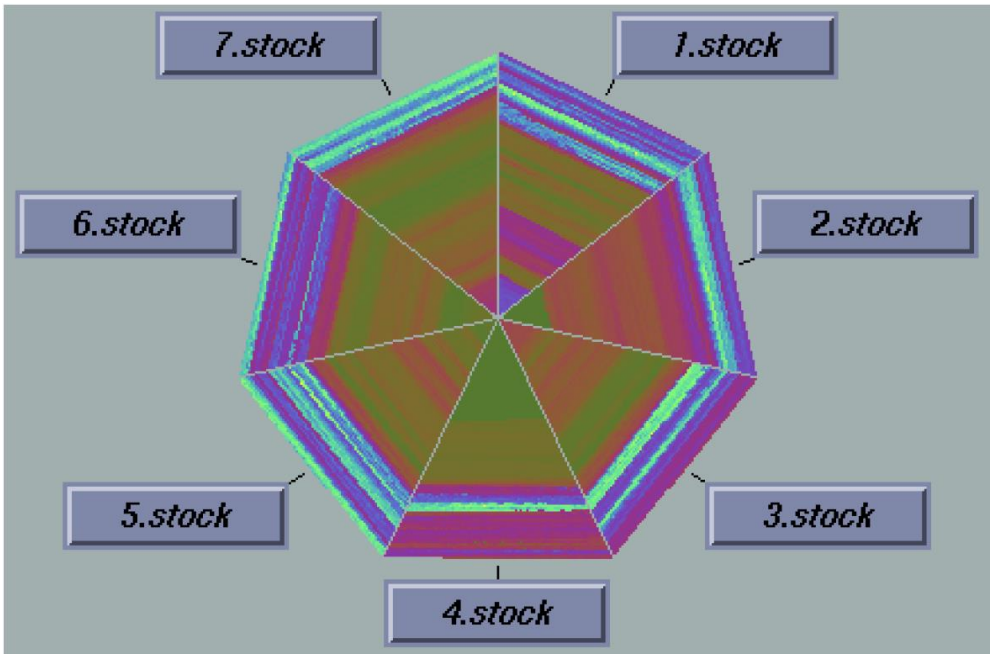


Figure 2.6. An example of some patterns and possible correlation between rainfall intensity and traffic speed identified from the AnalyticaR representation [5].



Src:http://kops.uni-konstanz.de/bitstream/handle/123456789/5943/Circle_Segments.pdf?sequence=1

Figure 2.7. A 'Circle Segments' Technique for 8-dimensional Data. Users can control the arrangement of the dimensions especially when visualizing large datasets using multiple dimensions [62].



Src:http://kops.uni-konstanz.de/bitstream/handle/123456789/5943/Circle_Segments.pdf?sequence=1

Figure 2.8. An example of visualizing 7-dimensional Data with the 'Circle Segments' Technique. Data used for this visualization is based on a stock market dataset [62].

2.3.2 DATA PROVENANCE

In order to visualize provenance, we need to understand data provenance. Therefore in this section, we will begin by understanding "*what is provenance*?" We will also look at the 'uses' and the 'issues' of data provenance which are highlighted by previous researches.

The notion of "provenance" originates from *"the art and archiving worlds, where it refers to information about the creation, chain of custody, modifications or influences pertaining to an artefact"* [59].

Data Provenance in the context of this thesis is defined as "**series of chronicles and the derivation history of data on meta-data**". Often scientific domains specifically use different forms of provenance for different purposes [13] [14]. Data provenance has gained an increase in interest recently due to its contribution in unique desiderata introduced by the distributed data in grids [13] [14].

Other facts which make provenance an interesting matter to consider and work with is the usefulness of provenance in certain domains and the fact that they are linked to the granularity at which it is collected [13] [14]. While provenance is proven to be interesting and important with graphing operations and its representations, the security of provenance can be an issue [23] [44] [64] [65] [66]. For example, copying and transforming data on and over the web or even in local machines has made it difficult to determine the origins of a piece of data [23].

Therefore the use visualization tools are a means of securing and tracking provenance when tracking file movements. On that note, existing visualization of provenance are for analysis and exploration purposes, which means the targeted audiences are technical people who seek methods to understand large datasets and provenance better. However the need to capitalize on data provenance towards the end-users advantage, is yet to be addressed. On that note, introducing "user-centric" visualizations to link data provenance and the end-users together is a prominent visualization technique that will result in understanding data provenance better. We will look into this after explaining about visualization of provenance in the next section.

2.3.3 VISUALIZATION OF PROVENANCE

As visualization in general is a popular means of data representation, visualizing provenance is still in its beginning years, whereby researchers are gradually changing from the current graphing (use of graphs) techniques, circular patterns and moving onto applying other existing visualization techniques. This is due to the rapid study and use of 'data mining' and analysis of large datasets.

Past and existing visualization of provenance are considered partial and contains a lot of information. For example, in the paper "Visualizing Large Scale Scientific Data Provenance [18]", the visualization approach is based on large datasets for exploratory purposes. Although there are features of user-centric visualization within this visualization, there are still no visibility with the fonts used in the writings. This makes it not user-centric.

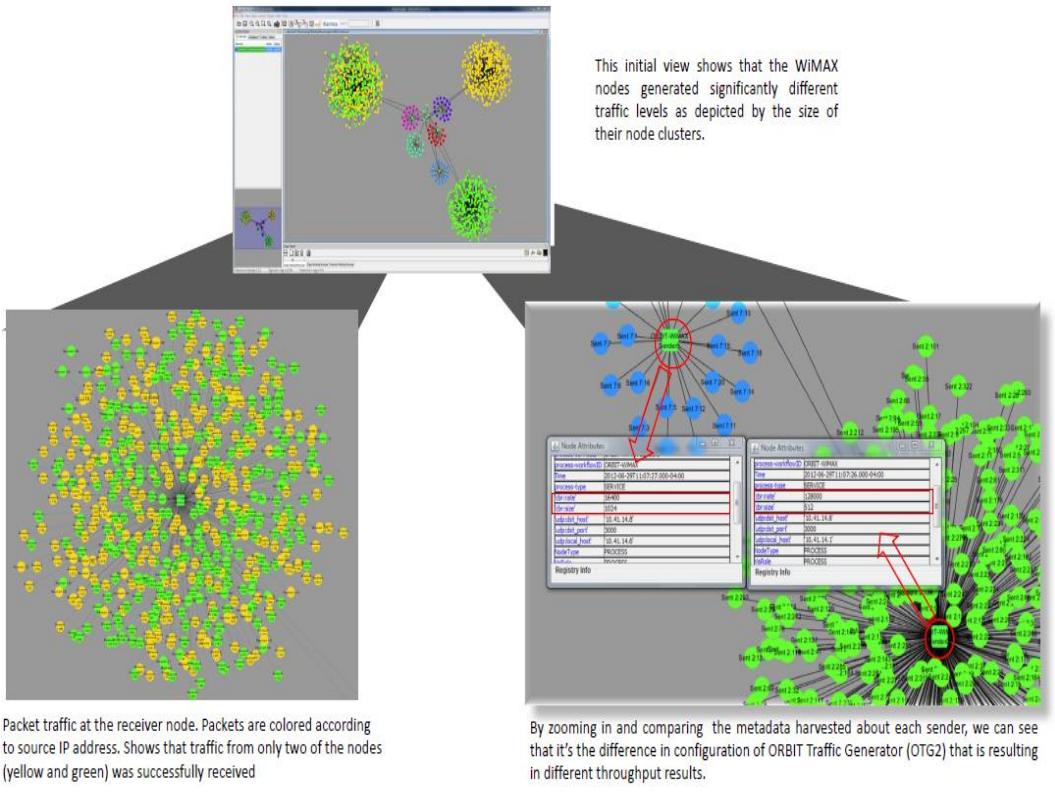


Figure 2.9. An example of an exploratory Visualization [1].

Another example of visualization of provenance is discussed in the paper "Visualization of network data provenance" [78]. Chen *et al.* aided their study on

large provenance datasets for exploratory and explanatory purposes [78]. This is another example of user-centric provenance visualization showing the chain of history of datasets.

This visualization uses the "layout" technique to improve the researchers' understanding of large datasets. The important components here in this the visualization are; 'the nature of provenance data' and the 'user requirements'. For example, Chen et al, showed (Figure 2.10) the provenance visualization displaying the relationship between sea-ice data products [78]. Here we see that the nature of the provenance dictates the outcome of the visualization. However, the user's requirement is an important input to producing this visualization as well. This provides the degree to how much user-centricity it is.

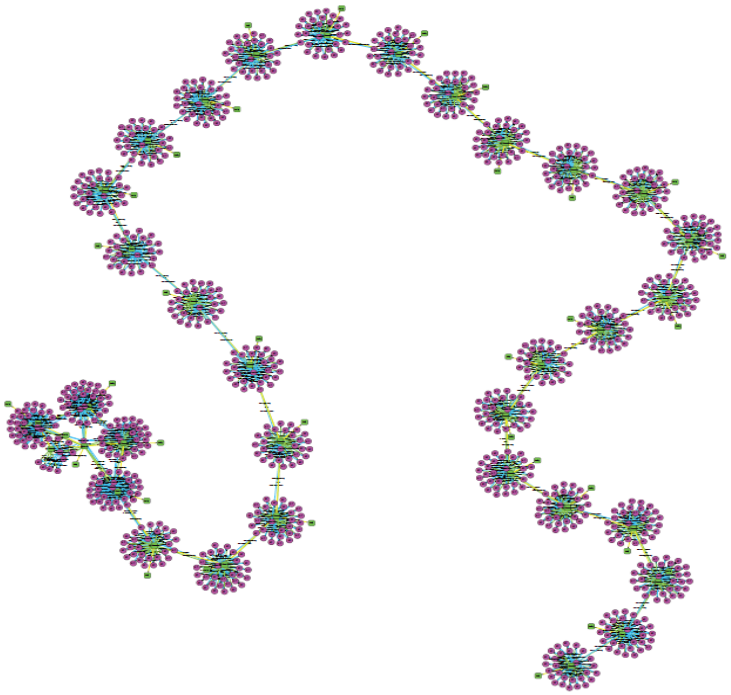


Figure 12.10. An example of Visualizing Provenance to show the history of relationships between daily products [78].

2.3.4 User-Centricity

As mentioned in the earlier sections, the connection between data provenance and end-users is a user-centric visualization technique. End-users contribute remarkably towards these techniques. However, users have remarkable perceptual capabilities that tend to be misjudged in current visualization designs and tools [71]. Occasionally, this is due to lack of analysing the users well in terms of [81]:

- Who are the users?
- What do the users want?
- What tasks can be performed using these visualizations?
- What are the users' working environments?

The need to empower web applications and security tools to become user-centric plays a huge role in allowing users to easily collaborate with applications and security tools. Therefore the developer has to consider not only the surface level of user interfaces but also the complex structure of users, tasks and the different functions that the system is built for.

This thesis stresses while existing visualization researches and tools are built for analytic purposes, they are not built for end-users [42]. For a standard end-user to be able to use these visualization tools to obtain satisfactory solutions their queries requires time and further training.

2.3.5 Gestalt Principle and User-centricity

Another part of our related works is to describe and evaluate the "Gestalt Theory of Perception" and investigate whether the application of Gestalt's theory into visualizations would make visualization user-centric. We will begin with making a comparison between two entities or observations.

Munzner *et al*. stated that "*to extend visualization's utility, we must integrate visualization with other techniques and tools from other disciples*" [52] and that user perceptual and cognitive capacities are mostly permanent and not subject to Moore's Law [52]. Therefore, based on humans' own perceptual and cognitive capacities, user views, the approach and understanding towards visualization often remains the same.

While human perceptual and cognitive capacities contribute to interpreting existence, science and visualization are useful when relating datasets to users so they can understand and relay better to different visual representations. In the area of design, Gestalt theory has succeeded on playing an important role in making visualization or visual data representation becoming more user-centric. The Gestalt principles established by Max Wertheimer, Wolfgang Kölher and Kurt koffka [29]

[55] states the key ideas behind Gestalt theory and the principles of the Gestalt perception model [2] [7] [8] [25] [45] [53].

While Gestalt theory has its advantages, it does possess its challenges and limitations. These challenges are closely related to the perceptual grouping and figure-ground organization. For example, there is a need for further analysis on the factors that are common in both the grouping and figure organization. There is also a need to fully distinguish between perceptual grouping and figure-ground organization [2]. Such challenges and limitations can affect visualizations when trying to apply both the perceptual grouping and figure-ground organization into one visualization. In other words, how can a visualization express and display either theory into one visualization?

Apart from challenges and limitations, there are other people who opposed Gestalts Work. People like I.P.Pavlo, who strongly provide objections to the Gestalt psychology issues of causality, methodology and on the problem of mind and body [54]. However these objections have not weakened the influence of Gestalt psychology.

Nevertheless, the Gestalt theory of Perception still dominates the research field with its wide application and use to date. The Gestalt perception model [47] states that "*the whole is other than the sum of the parts*." The original principles as listed in the paper *"Laws of Organization in Perceptual Forms"* by Max Wertheimer in 1923 stated the following [47]:

- Proximity
- Good figure
- Common Fate
- Similarity

We will now elaborate on these principles.

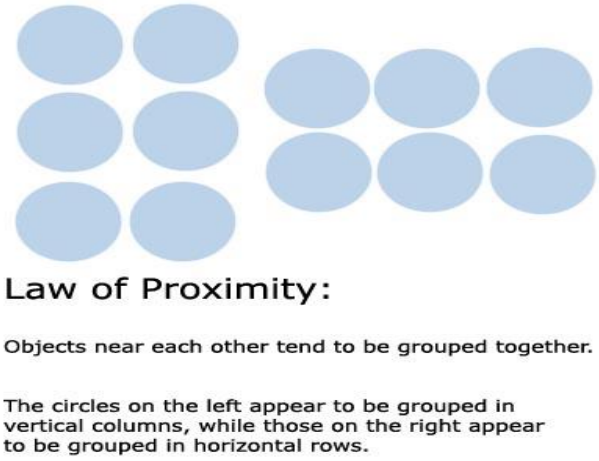
2.3.5.1 Proximity

The Proximity Law states that [32]:

- *"Items or points placed near each other appear to be a group."*
  - *Max Wertheimer*

According to the Law of "Proximity", observers naturally and mentally consolidate nearby elements into groups to form a coherent object, because they assume that closely spaced elements are related or associated and those further apart are unrelated [31] [32][46][53].

The purpose of this is to allow observers to draw up conclusions from visual representations according to their own judgement by allowing Proximity elements to induce the mind to perceive a collective or totality. This technique can be applied into visualization to allow users of the visualization to perceive a collective or totality when confronting visualizations of such natures.



Src: http://psychology.about.com/od/sensationandperception/ss/gestaltlaws_4.htm#step-heading

Figure 2.11. An example of the Gestalt Law of Proximity (Gestalt Theory) [46].

2.3.5.2 Pragnanz (Good Figure or Closure)

The Law of Pragnanz states that [45]:

- *"A stimulus will be organized into as good figure as possible."*
- *"Identifies an organizational tendency – a way in which the human brain decides that things go together"*
    - *Kohler Wolfgang*

The key idea here is that good figure delivers a simple design or a symmetrical layout [32] [45] [53]. The Law of Pragnanz was developed by Kohler (Gestalt psychologists) [31] [32] [45]. Pragnanz is a term in German that mean 'clarity' or

'good form'. This is why the Law of Pragnanz is often known as "Good Figure" or "Closure". The Law of Good Figure uses the Law of "Similarity" for a complete organizational tendency [32].

Src: http://www.intropsych.com/ch04_senses/laws_of_pragnanz.html

Figure 2.12. An illustration of the Law of Pragnanz – Good Figure, Closure (Gestalt Theory) [45].

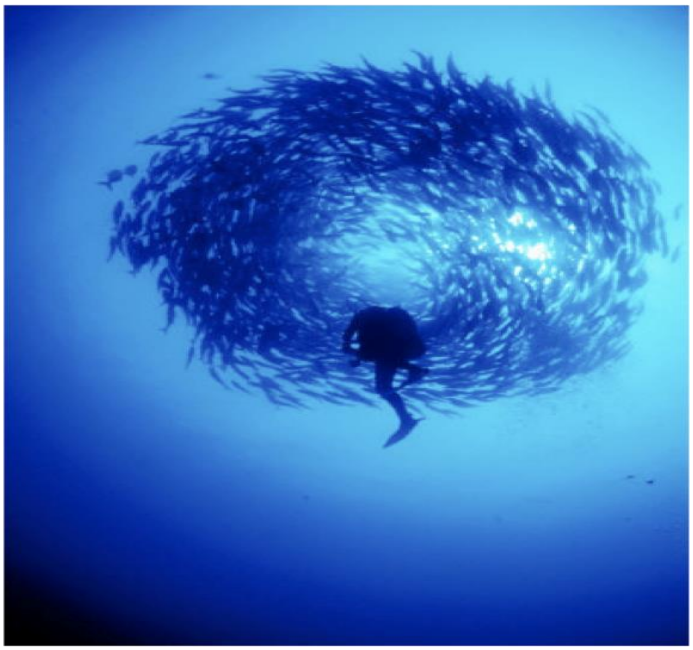
In Figure 2.12, the human mind has the ability to perceive and form that good figure of a dog sniffing the ground as shown in the red coloured circle. This mental and natural technique of forming images makes such visualization attractive to end-users therefore creating the whole experience a user-centric and subjective experience.

2.3.5.3 Common Fate

The Law of "Common Fate" states that [31]:

- *"Humans tend to perceive elements moving in the same direction as being more related than elements that are stationary or that move in the different direction"*

This law applies to stationary and motion objects. However the objects moving, for example in "Figure 2.13" below, fishes swimming in the same circular direction are clearly related to each other, by forming one unit [53]. This is the perceptual law of "Common Fate."



Src: http://students.aub.edu.lb/~awy01/psychology/organizationalperception.html

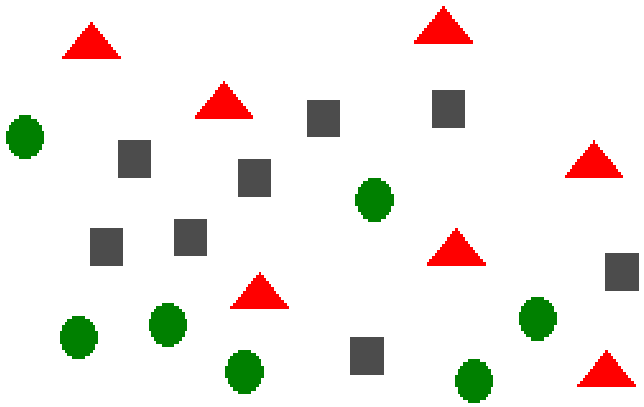
Figure 2.13. An illustration of the Law of Common Fate (Gestalt Theory) [53].

Applying the law of Common Fate to data visualization will allow users to perceive and draw out hidden information. This is particularly useful for applications into security event visualization. For example a group of moving data points, represented using bubbles, can alert end-users that these points represent the same file activity and could be related, whereas points stationary or scattered far apart illustrates that data points represent different and unrelated file activity.

2.3.5.4 Similarity

The Law of Similarity states that *"similar objects are set to be counted together as the same group."* The purpose of this is to attract the observers' attention [31] [32] [53] [72].

The adoption of the Law of Similarity to data visualization contributes to a vast end-user understanding towards scientific and technical data analysis. At the time of writing, the Gestalt Law of Similarity has been used within visualizations but researches have not clearly identified it and have not acknowledge its existence. Therefore the need to apply the Gestalt Law of Similarity to visualizing data provenance is highly recommended.



Src: http://architectingusability.com/2011/05/26/using-the-gestalt-laws-of-perception-in-ui-design/

Figure 22.14. An illustration of the Law of Similarity (Gestalt Theory) [72].

However in 1999, Stephan Palmer [47] [63] further added a variety of laws and principles onto the list of Gestalt principles. This is due to the fact that "Similarity" has all forms [19] [47] [76]. For example, Proximity can be Similarity in location; and Common Fate can also be Similarity in time; and others as well. The principle of "Continuation", "Simplicity", "Familiarity", and "Connectedness" have been added to the list and now commonly used by researchers [2] [8] [31] [32] [46] [72] [76].

2.3.6 PROGGER DATASET

Having to understand the Gestalt theory of perception and its application, we now take a look at the datasets which will be used in this thesis. All datasets are generated from Progger (short for "Provenance Logger") [58]. The datasets are the raw data for the visualization for this visualization technique. It is the provenance which we will later visualize to understand, track and observe data throughout this thesis.

This section also describes the current status of how Progger datasets are presented to the user and the components of Progger log. Moreover the correlation between the datasets and the application of Gestalt theory of perception are yet to be discussed.

```
progger log format.txt
//Log Formats
#define
LOG_OPEN(type,user,pid,ppid,audit,paudit,pname,filename,path,flags,mode,
fd) printk(KERN_INFO
"%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s,%u,%u,%lu\n",PROGGER_ID,type,user,pid,
ppid,audit,paudit,pname,filename,path,flags,mode,fd)
#define LOG_CLOSE(type,user,pid,ppid,audit,paudit,fd) printk(KERN_INFO
"%s%d,%s,%lu,%lu,%lu,%lu,%u\n",PROGGER_ID,type,user,pid,ppid,audit,paudi
t,fd)
#define LOG_S_CLOSE(type,user,pid,ppid,audit,paudit,fd) printk(KERN_INFO
"%s%d,%s,%lu,%lu,%lu,%lu,%us\n",PROGGER_ID,type,user,pid,ppid,audit,paud
it,fd)
#define
LOG_RENAME(type,user,pid,ppid,audit,paudit,pname,oldfile,newfile,path)
printk(KERN_INFO
"%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s,%s\n",PROGGER_ID,type,user,pid,ppid,au
dit,paudit,pname,oldfile,newfile,path)
#define LOG_UNLINK(type,user,pid,ppid,audit,paudit,pname,filename,path)
printk(KERN_INFO
"%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s\n",PROGGER_ID,type,user,pid,ppid,audit
,paudit,pname,filename,path)
#define
```

Figure 2.15. A captured snapshot of parts of the Progger log format.

We begin this section by observing existing snapshots of the Progger log for a better visual understanding. Figures 2.15, 2.16 and 2.17 shows the format of the log, the Progger system call lists, and how Progger records and presents its data to users. Currently the Progger dataset is presented in a log format and has no visual

representation. It contains a lot of kernel based information collected from capturing system calls in the Linux kernel module. For example, kernel version, type of system-call, user, processor-id, filename, file-paths and other required details.

As observed in the Figure 2.15, we see that the Progger log format contains all the syscalls. Each syscall types contain attributes that correspond to the type of syscall. For example, a syscall for "open" is "LOG_OPEN." The attributes associated with the syscall open, are; 'type'. 'user', 'pid', 'ppid' all the way down to attribute 'fd'. These attributes are assigned to store values related to the "LOG_OPEN" syscall. The second part to the LOG_OPEN call type, is a line "printk" function. This executes all prints to the corresponding attributes stated above. There are about 32 syscall types, and all have different attributes, types and values. Figure 2.16 defines the system calls and identifies them with an integer.

```
                              progger syscall list.txt
#define SYSCALL_OPEN                  0
#define SYSCALL_UNLINK                1
#define SYSCALL_WRITE                 2
#define SYSCALL_CREAT                 3
#define SYSCALL_MOVE                  4
#define SYSCALL_CLOSE                 5
#define SYSCALL_READ                  6
#define SYSCALL_S_CONNECT             7
#define SYSCALL_S_SENDTO              8
#define SYSCALL_UNLINKAT              9
#define SYSCALL_MKDIR                 10
#define SYSCALL_RMDIR                 11
#define SYSCALL_SYMLINK               12
#define SYSCALL_LINK                  13
#define SYSCALL_LINKAT                14
#define SYSCALL_CHOWN                 15
#define SYSCALL_FCHOWN                16
#define SYSCALL_LCHOWN                17
#define SYSCALL_FCHOWNAT              18
#define SYSCALL_CHMOD                 19
#define SYSCALL_FCHMOD                20
#define SYSCALL_FCHMODAT              21
#define SYSCALL_S_SENDMSG             22
#define SYSCALL_S_ACCEPT              23
#define SYSCALL_S_SOCKET              24
#define SYSCALL_SENDFILE              25
#define SYSCALL_S_RECVFROM            26
#define SYSCALL_S_RECVMSG             27
#define SYSCALL_DUP2                  28
#define SYSCALL_PIPE                  29
#define SYSCALL_PIPE2                 30
#define SYSCALL_DUP                   31
```

Figure 2.16. A captured snapshot of Progger log syscalls.

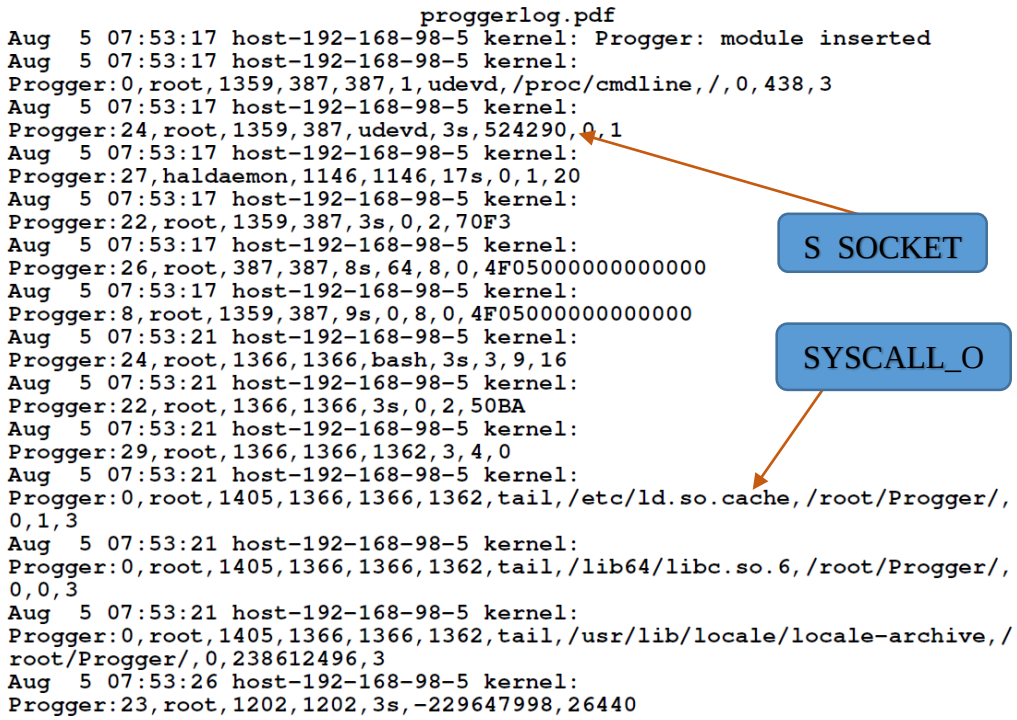


Figure 2.17. A Progger raw data sample generated and displayed in a log file format.

Observing Figure 2.17, we see a captured system call from Progger. As indicated in the Figure 2.17, a "Progger: 24, root, etc..." call was executed as shown in Figure 2.17. "Progger: 24, root, …," is identified as "Syscall_S_SOCKET" call. "Progger: 0, root, …, is equivalent to a "Syscall_OPEN" call. This call is executed when a file, directory is opened.

At this stage, this thesis acknowledges the development of Progger being at its initial state and that it has no end-user visualization platform. However, its ability to track files and operations makes it interesting to use.

The current log representation (without visualization) can be regarded as 'too much' information for an end-user especially while trying to understand and carry out such user actions as 'tracking' his/her files. Therefore the need to build a visualization tool alongside Progger is in high demand. This is to provide the same output information as the current log but in a visual representation view that any end-user could go to when tracking files.

Overall, previous researches have produced user-centric visualization tools but the targeted audiences are directed towards the technical and research personals and such visualization are purely for exploratory and analytical purposes. End-users

have little access to web tools that are available for visualizing data provenance. Finally the need to apply Gestalt laws in a graphic design perspective alongside the scientific visual representation does have an effect on changing the current idea of using visualization for exploratory and analytical purposes to an end-user demand. This visualization application data provenance is a ‘missing link’ or factor between end-users and existing visual representations (Data Provenance Visualization).

CHAPTER 3 ANALYSING DIFFERENT DATABASE STORAGE

Progger logs provides the data as we have seen in the earlier sections of this thesis. Yet, all these data needs to be stored in a database and with data collected from Progger increasing as machines are added to the network, a suitable database is required for data recording and storage purposes. This fast data increase is due to how the kernel handles system calls.

3.1 COMPARING AND UNDERSTANDING DIFFERENT DATABASES

For this thesis we had analysed several database management systems, such as MySQL, CouchDB, MongoDB, and Hadoop (HBase and Hive). All types of database have advantages and disadvantages such as processing speed, data capacity, and compatibility with designated applications. However, choosing the right database system for this thesis is very important. The reason for this choose is based on several aspects of how valuable the data is. Such reasons are:

- Scalability of data size
- Availability of the data when needed
- Accessibility of data when needed
- Security of data storage and handling.
- Importance of Data content

We will now provide an analysis for the five different database systems from both SQL and NoSQL schemes. Table 3.1 provides the details of these various database systems.

DATABASE STORAGE ANALYSIS					
	CouchDB	MySQL	MongoDB	HBase	Hive
Affordable	√	√	√	√	√
Consistency	√	√	√	√	√
Compression	√	√		√	√
Fault-tolerance	√			√	√
Normalize		√			√
Map/Reduce	√		√	√	√
Scalable	√		√	√	√
Real-Time Read/Write Access	√	√	√	√	√
High Performance	√		√	√	√
Replication	√		√	√	√
Open Source	√		√	√	√
User-centric		√		√	√
Database Model	Document Store	Relational DBMS	Document Store	Wide Column Stored	Relational DBMS

Table 3.1. A comparison between the database systems [3] [50] [60] [61].

Table 3.1 shows the comparisons between MySQL and the NoSQL databases. We see that MySQL performs faster with small and medium sizes of datasets, and not big datasets. With small and medium sizes of data, there is reliability and good table, column, and row management which allows less duplication of records, conflicts and invalid transactions. However, with MySQL, data scalability, single server performance and strict schema design are the issue when the data size grows [3].

Comparing that to the NoSQL databases, we notice that, performance and scalability is not an issue. However, it is just other features that change. For example, CouchDB performs the best when it comes to "peer-to-peer" replication, either in a

local network or even over the cloud [3] [61]. MongoDB has weaker replication but offers full consistency, high performance and allows extensive uses of memory-mapped files, i.e. read/write – through memory caching [3] [60]. Finally, Hadoop – Hive and HBase offers all the above features mentioned for NoSQL databases but has a unique feature. It is user-friendly. It is pre-packaged in a virtual Image environment ready for use. It offers a web interface access, and it is in a "plug and play" state, with such features such as:

- Ambari – Dashboard
- Hive - Database System
- HBase – Database system
- Pig – Scripting language

With the above analysis, this thesis chooses to implement Hadoop-Hive over the other database systems.

3.2 Processing Large Datasets

In this section of the thesis, we will provide two database installation modules. These databases are MySQL and Hadoop – Hive databases. We begin by stating the required components of the two setups, and their functions in the database storage systems. Secondly, we will provide the proposed Hadoop – Apache Hive database system design. Since this section of the thesis covers the design of processing of large datasets, we will provide further analysis to the database storage setups, speed, and performance in Chapter 7 – Evaluation, of this thesis. We will begin with MySQL and later provide information for the Hadoop – Hive databases.

3.2.1 Setup1 - MySQL Database and Storage

MySQL database due to its popularity became a choice for the storage of Progger logs. The installation of MySQL is on a windows environment, and it has a windows web development tool that is installed to manage the overall process and operations. There are several options to install MySQL in a windows environment such as XAMP and WampServer. In this thesis, we have implemented WampServer (64BITS & PHP 5.5) 2.5 to manage MySQL and other related services like Apache.

3.2.1.1 WampServer

The WampServer is a windows web development environment for Apache, MySQL, and PHP [79]. It allows the user to create web applications with PHP, Apache and MySQL database on windows. In addition, it has another component called "PhpMyadmin" which gives access to the user to manage the databases. Overall, WampServer is a windows web development environment that has the functionalities of a server, and manages Apache, PHP and MySQL.

In order for Apache, PHP and MySQL to work, one has to download and install WampServer 2.5 from the website: http://www.wampserver.com/en/ . Upon completion of download, the installation process should begin on your windows Machine. The installation process consists of installing Apache, PHP and PhpMyAdmin.

3.2.1.2 PhpMyAdmin

PhpMyAdmin is a software tool that provides a web access/interface to MySQL and allows the user to manage to your database. All database are creating using phpMyAdmin web access. To access the phpMyAdmin page, the user has to access it using the localhost address: http://localhost/phpmyadmin/ .

In order to start using this service and start storing data, the user has to create a new database, tables and load the data into the Database. Figure 3.1 shows the "phpMyAdmin" interface with a database created (proggerdb) and a list of tables which contains the Progger data. Because, the Progger data is big, "views" are created in the database to handle faster performance. (*Refer to Appendices F.1 to further understand the configuration of phpMyAdmin.*)

The main features of phpMyAdmin are creating the databases and tables to handle data. PhpMyAdmin also offers 'view' and 'queries' to handle data processing and performance purposes. Which processing of data can be done within 'phpMyAdmin' a connection is required to connect the phpMyAdmin database to the front-end API. This can be done using several methods, however for this thesis a 'php script' is written to facilitate the connection between the database and the front-end API.

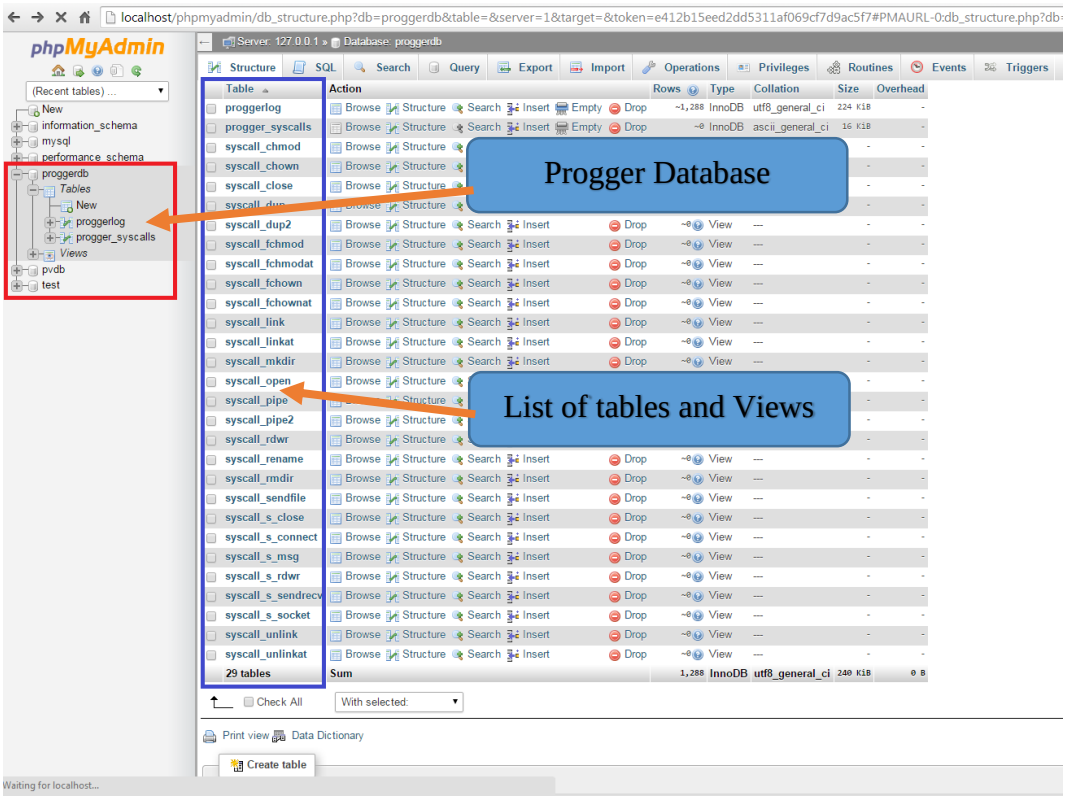


Figure 3.1. A snapshot of phpmyAdmin database system.

3.3 Setup 2 - Apache Hadoop - Database and Storage

The other setup we have modelled is Hadoop – Hive/HBase framework. The advantage of implementing Hadoop over MySQL is, all features are wrapped into one virtual machine image.

3.3.1 Hortonworks Sandbox 2.1

The Hadoop environment uses the foundational platform of Hortonworks Sandbox 2.1, i.e. a single node implementation of the Hortonworks Data Platform (HDP). It is wrapped as a virtual machine to allow fast and easy experimentation with the Hortonworks Data Platform (HDP) [34]. It can be easily imported into a virtualization environment for quick implementation.

3.3.2 Why Hadoop and Hive/HBase?

The components of Hadoop are:

- Hadoop Distribution File System (HDFS)
- MapReduce

The advantages of implementing Hadoop over the other evaluated Databases is that Hadoop is affordable, reliable and provides scalability. It also contains chain of tools intended to solve big data problems. It has the ability to process big amount of data irrespective of the data structure and provides data motion to numerous applications. The Hadoop components acts as the basis of the Hadoop cluster and manages how datasets are stored in the Hadoop cluster. It allows a user to store, process, and analyse large volumes of data [34].

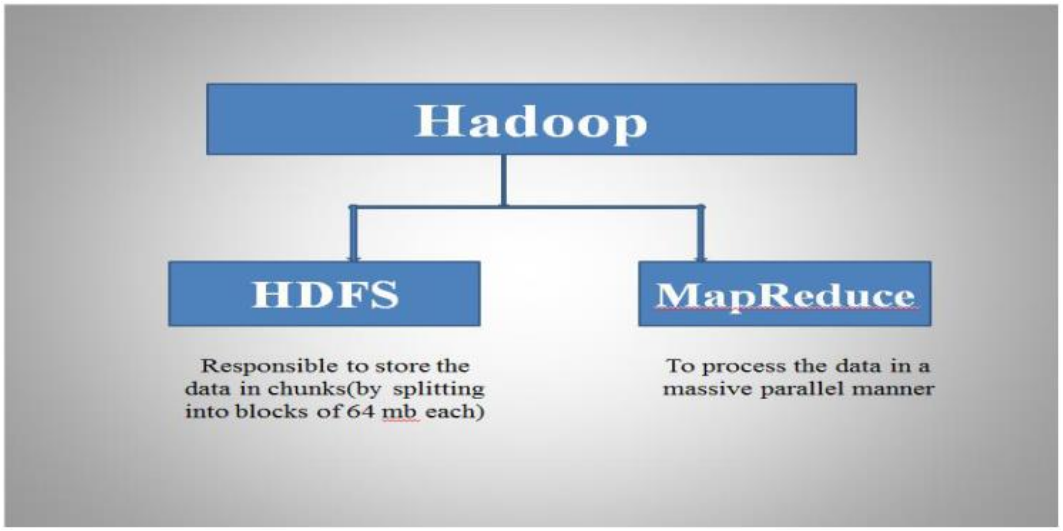


Figure 3.2. A Hadoop structure with its core components [33] [34].

Data storage is the important aspect and core function for any Database systems. The HDFS in Hadoop is responsible for storing data in chunks (by splitting into blocks of 64mb each). This allows Hadoop to have a better advantage over the other databases previously mentioned. We now look at the Hadoop modules that handle the data storage tasks and briefly describe their functions.

3.4 Components of Hadoop used for Processing Large Datasets

The Hadoop modules that are used for this thesis are:

3.4.1 Ambari

The Hadoop Ambari is web-based front-end that acts as the management and monitoring interface for the whole Hadoop framework [80]. It provides are dashboard to view and access the modules. It links up all the modules together such as Hive, Pig, HBase, HCatalog, Zookeeper, Oozie, and Sqooq. Another useful feature for Ambari is, it provides support for Hadoop HDFS and Hadoop MapReduce. For the case of this thesis, much of our concentration are on Hive, Hbase, Pig and Hcatalog.

3.4.2 Hive

Hive is a module of the Hortonworks Data Platform (HDP) which provides a SQL-like interface purposely to store data in Apache Hadoop [80]. It shares similar tasks to HBase, however has its own advantages. For example, Hive is considered user-friendly and familiar to users who are SQL-like for querying data.one can conclude that Hive is the user-centric option compared to the HBase.

3.4.3 HBase

HBase, like Ambari, is part of the Hadoop distributed database that maintains a structured data storage for large table. It is a module based on top of the HDFS. It is an open-source, distributed and column-oriented store modelled after Google' Bigtable [3] [80].

3.4.4 Pig

Pig which works alongside HBase, is scripting language with the primary focus on dataflow. It runs on Hadoop and uses MapReduce and the HDFS. It is a high level data-flow language for parallel computation and it is design for executing long series of data operations [80]. Pig has the following features [10]:

- Extensible
- Easy to program
- Self-optimizing

3.5 HADOOP – HIVE DATABASE DESIGN

Now that we have understand all the Hadoop components and their functions, let's look at the proposed Database design. Figure 3.3 shows the Hadoop- Hive database design.

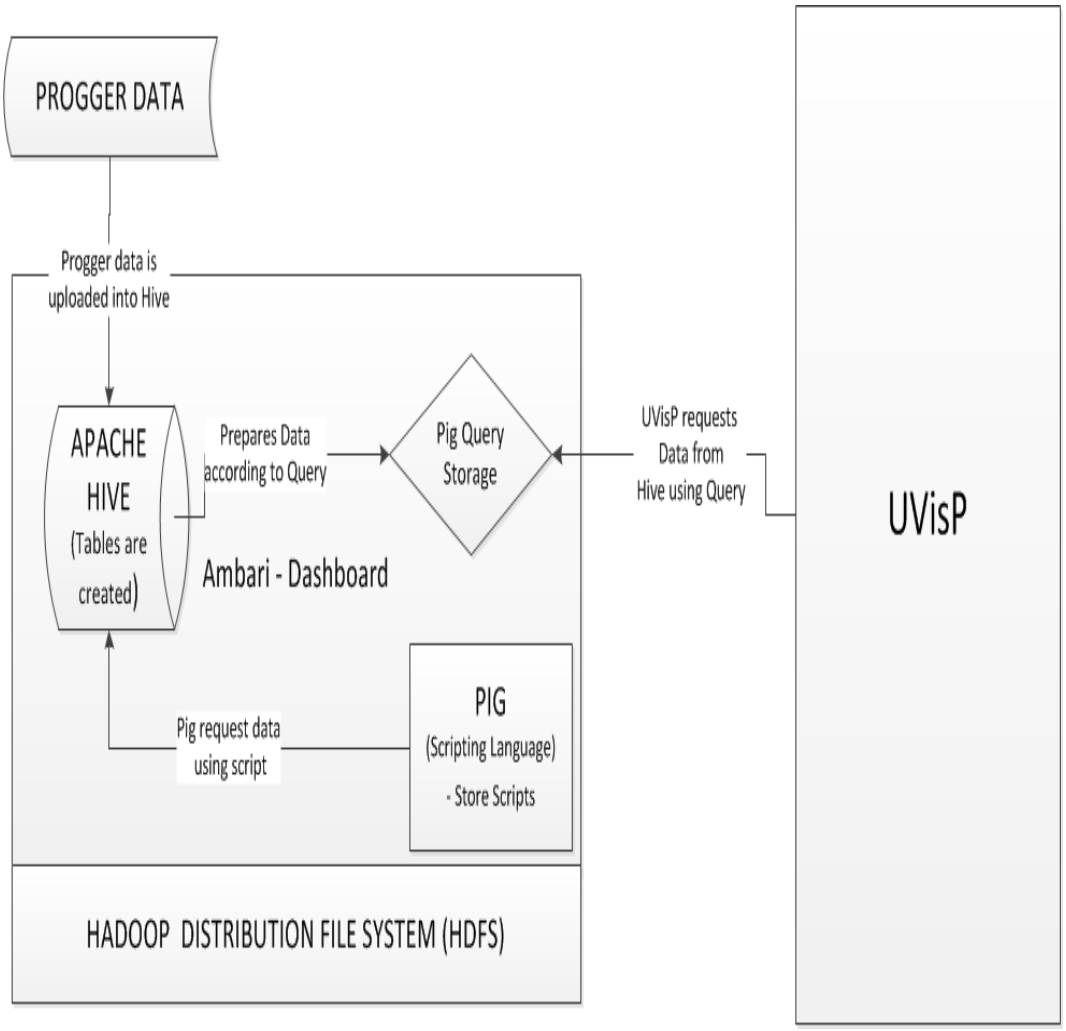


Figure 3.3. The Hadoop – Hive database design.

With such main features (user-centric), Hadoop became the choice for this thesis in terms of database and storage. All database, tables, queries and scripts are created using Hadoop and managed by Hadoop via Hortonworks 2.2sandbox. All databases are created in Hive and are channelled using Pig and Php scripts to the front-end for processing and visualization. (*Refer to Appendix F.2 for the related Hadoop installations files and snapshots*)

While analysing and testing out the two database installation options (MySQL and Hadoop – Hive), we notice that Hadoop performs faster and better than MySQL in many ways. For example, the overall installation and setup of Hadoop is easier and faster than the MySQL installation process. Performance is the main pitfall of MySQL. For example, the time it retrieve data, write data, and outputs it for visualization is slower than Hadoop. The size of the data is another factor that affects the performance of MySQL, however it does not affect Hadoop since Hadoop is built purposely for large datasets. Finally Hadoop –Hive provides a user-centric approach that is easy for anyone with little knowledge on database systems to implement.

CHAPTER 4 D3 AND GESTALT INTEGRATION DESIGN

4.1 D3.JS

As previously discussed, Hadoop - Hive is needed for data storage. However, visualizing the provenance of the data stored in Hive requires a Visualization API. In this thesis we chose to use D3.js for the visualizations development. We begin this section of the thesis by understanding D3.

D3 (short for Data-Driven Documents) is a JavaScript library for manipulating documents founded on data [21]. It helps developers to bind arbitrary data to a Document Object Model (DOM) and finally applying data-driven transformations and transitions to the document to acquire visualization [21]. To use D3.js for the 'front-end' of the web application, there are two options:

- download D3 (d3.zip) as a package and link to your 'front-end' web application
- directly link to the latest online release using the script below:

```
<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
```

Figure 3 Figure 4.1. A note on how to get D3.js and links [21].

Why D3.js?

D3.js is selected due to its capabilities and features that allows developers to build any data visualization framework according to their design and wants. Below are the main powerful features of D3 [17]:

- Easy mapping of data to HTML elements and vice versa
- Flexibility of render elements, either HTML or SVG.
- Simple API:
    - Configuring layouts
    - Configuring styling
- The transition API supports animations.

- Ability to build data trees according to how users visualize the output.

With such features, D3.js attracts developers to design and develop visualizations. It also allows developers to re-use codes and add specific functions to suit the content used. Overall, D3.js allows users to interact with the front-end application making it a user-centric application.

4.2 UVISP'S USER-CENTRIC TECHNIQUE

With the foundation knowledge of D3.js and how it works, we now take a look at an important key phases of the UVisP design. This thesis refers to the key phases as the 'UVisP's U-C methodology.' The methodology consists of (in an orderly manner):

1. Prior-knowledge of the Gestalt theory of perception
2. Processing of the raw data (Progger log)
    - Type, User, Filename, FD, etc.
3. Building data-input format (.json & .csv)
4. Visualization type & animation (D3 – Library)
5. Colour Theory
    - Which colour selection presents each visualization
    - How many colours to use?
    - Applying the colour wheel/circle (a tool for combining colours)
6. Gestalt - & - D3 Association Visualization Approach

With this U-C methodology in place, users of the UVisP technique can adopt this visualization technique to help them extract useful information and knowledge from any given visualizations. However, in order to use this visualization technique well, the user must have a full understanding of the U-C methodology presented above. Therefore we will take further describe each step of the methodology in detail for better understanding.

Referring to 'Step 1' of the U-C methodology, we consider looking at the meaning of 'prior-knowledge of the Gestalt theory of perception.' In this step, a user of the

UVisP has to understand what the Gestalt theory of perception is. This can be done via a user-guide of the UVisP technique and or 'help' modules on the internet that are based on the Gestalt theory of perception. A fair understanding of the Gestalt Laws / Principles is the minimum requirement for the 'Step 1'of this U-C methodology.

'Step 2' of the U-C methodology states that, 'processing of the raw data (Progger log)'. In this step, a user with the preferred visualization in mind, should have some knowledge of what type of data is being process for the visualization. For example, tracking or visualizing the data movement requires certain data types. This knowledge will provide the user with some clues to how the visualization will look like. In any visualizations, like any other art portraits, clues and prior knowledge are the key sources of building that image and final knowledge/story from that particular visualization.

Processing of the raw data plays a very important role for 'Step 3'of the U-C methodology. The processed raw data has to be in a form that can easily be build and converted into a '.json' or '.csv' format. This step allows the UVisP to produce better user-centric visualizations. In other words, the visualization do tell a story and are viewed over and over again by the users of the UVisP technique.

Since we now understand the very first basic parts of the U-C methodology (Step 1 – 3), 'Step 4 (Visualization type & animation (D3 – Library))' and 'Step 5' are important, since they direct the way the users of the UVisP would see the visualizations. In other words, 'Steps 4 and 5' creates the connection to 'Step 1' again in order for whole methodology to focus only on the output of the visualization.

Once all 'Steps 1 – 5' are achieved, 'Step 6' integrates the whole process of the U-C methodology back to the human perception where the user has the power of perceiving useful information and knowledge from the visualizations.

4.3 D3 VISUALIZATION AND GESTALT INTEGRATION DESIGN

This section describes a model design of the Gestalt application and the user's visualization experience when interacting with UVisP. The emphasis is that humans

are natural at "perceptual mapping (a technique)" and often their perception are permanent [34]. This is the approach using Gestalt theory.

4.3.1 Types of Visualizations

Using the Gestalt theory approach, we describe how the D3 visualizations and Gestalt would integrate together in the UVisP technique. Table 4.1 shows the types of visualizations. Each type of visualization has a 'visualization code' assign to it. For example, '*Vis_GP1*' is assigned to a '`Create/Open a .txt file, write to the .txt file, and later save the file`' type visualization.

Types of Visualizations	
Visualization Code	Visualization Scenario
Vis_GP1	*Create/Open a .txt file, write to the .txt file, and later save the file.*
Vis_GP2	*SCP a .txt file from one folder to another folder in the same local machine.*
Vis_GP3	*SCP a .txt file from one VM to another VM in the same network.*
Vis_GP4	*Open .txt file, read & write/change content of .txt file, save .txt file and close .txt file.*
Vis_GP5	*Move .txt file from one folder to the other folder in the same local machine.*
Vis_GP6	*Delete .txt file from folder.*

Table 4.1 Types of Visualizations.

4.3.2 Common Grounds in Visualization Type and Gestalt Laws.

Table 4.1 shows the type of visualizations. The two columns represents two different parts to the Progger logs. In Table 4.1, the "Visualization Code" is added to act as a form of ID or tagging. The "Visualization Scenario" column contains the different Progger scenarios, which purposely to produce provenance visualizations. The visualizations are then added to the different Gestalt laws/principles to compare and analyse for common grounds (union). Figure 4.2 shows the comparison and analysis for common grounds.

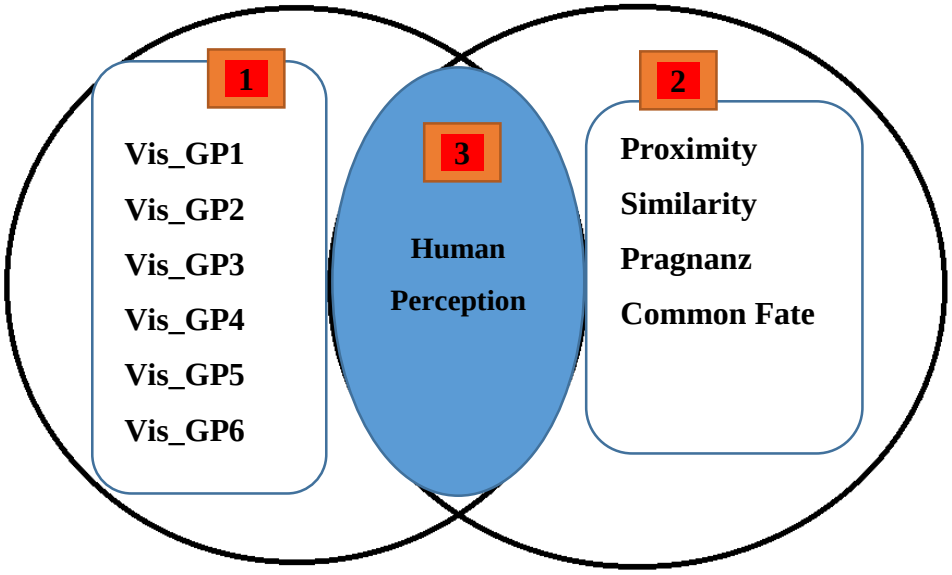


Figure 4.2. The UVisP D3 – Gestalt visualization technique.

The idea of using Figure 4.2 is to summaries the idea behind integrating the visualization scenarios and the Gestalt's theory of perception. This is shown whereby the 'union (3)' integrates 'visualization type (1)' and 'Gestalt theory (2)' together. This union is the 'human perception'. Therefore human perception in visualizations has a big impact on the presented knowledge to the users. The knowledge gained when visualizing using the UVisP technique are directed towards the provenance of data.

With that knowledge in hand, a model design of how a user (human) would interact with the UVisP technique with the aim of extracting useful information from the visualizations is shown below:

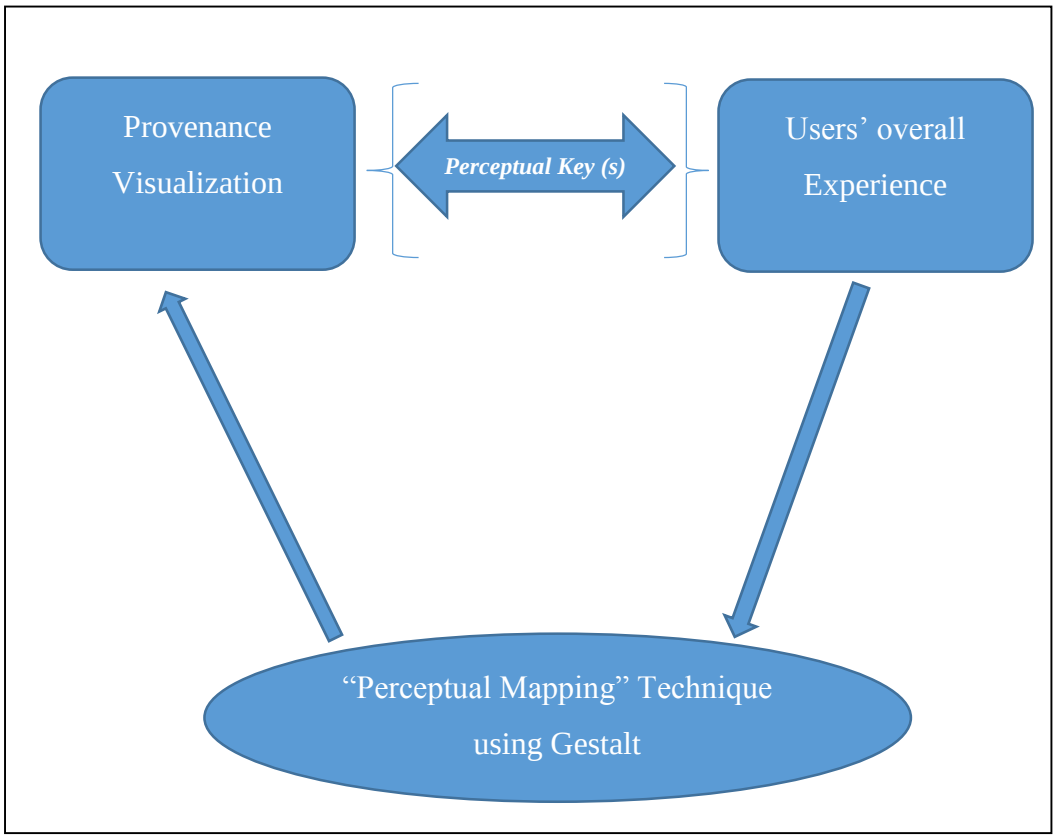


Figure 4.3. A D3 – Gestalt Visualization Model of the user perceptual experience.

4.3.3 The use of the Perceptual Key in Visualizations

The model design in Figure 4.3 states that a "`perceptual key`" in humans plays an important role in the UVisP technique, i.e. the user depends on it to obtain user-driven visualization experience when using UVisP. '`Perceptual key(s)`' in this thesis is defined as; the act of recognizing the very first key-identifier, eye-catching, common and recurring pattern(s) in a given visualization that are sourced out by the users of UVisP technique. For example, in Figure 4.4, there are several perceptual keys such as the dogs 'head', 'body' and or 'legs.' The order at which the perceptual key(s) appear depends on the users of the UVisP technique.

In addition, the "perceptual mapping" technique is applied as well, by the user when confronting different visualizations of UVisP technique [4] [57]. 'Perceptual

mapping' technique is the ability for the users to visualize objects by forming virtual arrangements (virtual images) the moment the user visualizes a visual representation. This technique helps users to build the knowledge faster, retrieve and extract useful knowledge from the visual representation.

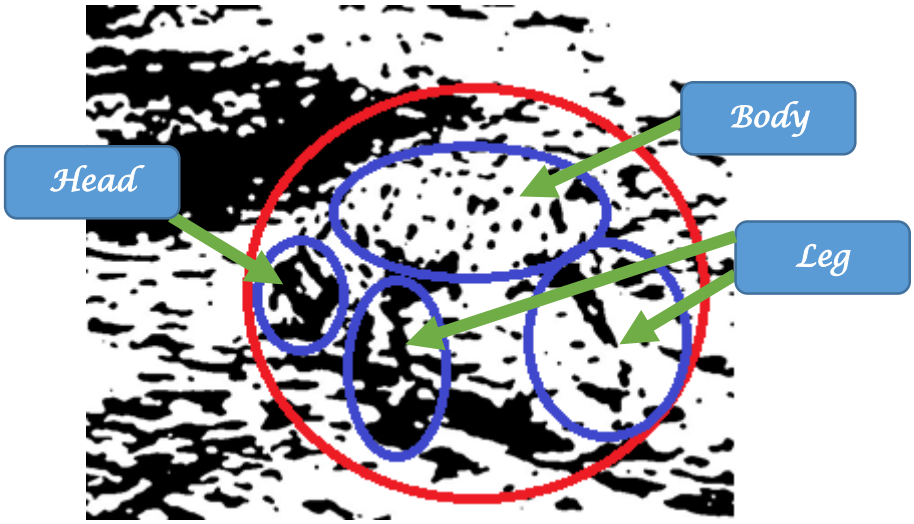


Figure 4.4. A perceptual key (s) illustrated in the Pragnanz visualization of the dog [45]. All possible 'perceptual key (s) are shown using the blue circles/ovals.

4.3.4 User Queries /Survey

Another important aspect to the "D3 - Gestalt visualization Model" involves manually identifying user queries when using the cloud services. This has been done by carrying out a user survey (questionnaire) to categorize queries from most important to least important. We begin by providing a step-by-step method of how the user survey was carried out.

Step 1: Develop/brain-storm up to 100 user queries when using a cloud service.

Step 2: Categorize all user queries into common groups. For example, "file creation" and "file movement"

Step 3: Narrow down the user queries to the top 20 user queries. For this step, selecting the top 20 queries goes in line with what the users of the cloud consider most important when tracking and monitoring data activities.

Step 4: Using the top 20 user queries, carryout a survey/questionnaire by asking random users of the cloud to select their top 3 user queries based on the importance of data stored in the cloud. The top 3 selected queries are for the survey.

Step 5: Based on the survey/ questionnaire, the top 3 user queries are associated with the Gestalt theory for visualization.

Table 4.2 shows the user survey on the first 20 user queries (*Refer to Appendix E for full survey findings*):

Type of Visualization	Goal of Visualization	D3 Visualization representation	Gestalt Approach					
			similarity	uniformility	Closure	continuation	Proximity	figure
Open file actions base on different PID	statistics on File access or modification	Treemap / Buble chart	√					
File created (open), read, write and close	User Report	Force directed Graph/ treemap /heatmap		√		√	√	
copying of a file and saving it to new directory	keeping track of the file movement/activities	Collapsible Tree layoutTree / Buble	√	√				
moving a file from one directory to another (SCP)	keeping track of the file movement/activities/in case of leakage	Froce directed Graph	√					
renaming a file	statistics, tracking of	Treemap / Buble chart	√					
rename directory	statistics, tracking of	treemap / buble chart	√					
Open file -> Read -> Write ->save in new directory	File Process and tracking	Treemap / Buble chart / heatmap / heatmap	√	√	√			√
delete file action	statistics to show which possible types of files	buble chart	√					
tracking multiple CRWC actions	percieve visualization for new paterns of processes	Collapsible Force layout /heatmap	√		√		√	√
identifying Socket processes	Statistics of whats happening behind the seen	Treemap / Buble chart /heatmap	√			√		
moving file from VM to PM (file = any type of file)	tracking of Data movement for leakage	Treemap / Buble chart	√				√	
Single create, read, close file in VM then move to PM	File Tracking	Collapsible Tree layoutTree / Buble		√	√			√
multiple Data CRWC file then	tracking for leakage	Streamgraph / heatmap		√	√			√
File created (open), read, write and close in VM	statistics purposes	buble chart / heat map	√		√	√	√	√
File created (open), read, write and close in PM	statistics purposes	buble chart / heat map	√		√	√	√	√
Create, read, write file	Statistics purposes	zoomable Treemap	√				√	
sendMSG action in the kernal	statistics	Treemap / Buble chart	√	√				
recvMSG ation in the kernal	statistics purposes and also identify who the msg is	Donut chats with labels	√	√				
Change of file permission	statistics and possibly know why permision has been	bubble charts / donut chat/	√					
remove file from location	statistics	Treemap / Buble chart	√					

Table 4.2. A D3 – Gestalt Model of association.

The aim of this user survey is to identify:

- types of visualization (Based on Progger system call actions)

- goal of visualization
- type of D3 visual representation
- which Gestalt law approach will match the type and D3 visualization

With the aims in place, a justifiable method is used to design and obtain the 'types of visualizations', 'goals of visualizations' and link them up with the Gestalt approach. This method not only has to work for the end visualization, but also to link up all components such as the:

- Progger scenarios
- Reasons for the scenarios
- What type of visualization model is suitable for this scenarios
- Gestalt approach proposed for each scenarios.

Figure 4.2 shows that to link all components together requires two processes:

Process 1:

Identify and analysing the existing D3 visualization templates and manually linking them up with how the Progger scenarios are executed. For example, a '`D3 Treemap`' visualization can be linked to the Progger scenario that allows the user to "*create, read, write and finally save/close*" a file. The choice of selecting a `D3 Treemap` to visualize this scenario is based on the formation of data structure (.json or .csv).

Process 2:

Once Process 1 is complete, the type of Gestalt approach will be added to the visualization based on the behaviour of the data when executing the scenario from Progger. The behaviour of the Progger data can only be understood if it is analysed against time, user involved, and the required files needed and visualized.

The findings should provide a justifiable indication to which D3 visualization representation will associate with the Progger scenarios.

Finally this section of the thesis covers the most sensitive aspect of the visualization technique proposed. The overall structure, flow-functions and output is tailored toward user-centricity therefore it targets simplicity for both the users and also the UVisP operations.

CHAPTER 5 UVISP DESIGN AND METHODOLOGY

This section presents the UVisP skeleton, process view and describes the UVisP technique with its various components. The components consist of the Storage and database, programming language used to implement the UVisP technique and the incorporation of D3.js with the concept of the Gestalt principles to produce user-centric (user driven) visualizations.

5.1 UVISP SKELETON VIEW

In this section a skeleton view of the web-front-end is presented. It covers the core components of the technique.

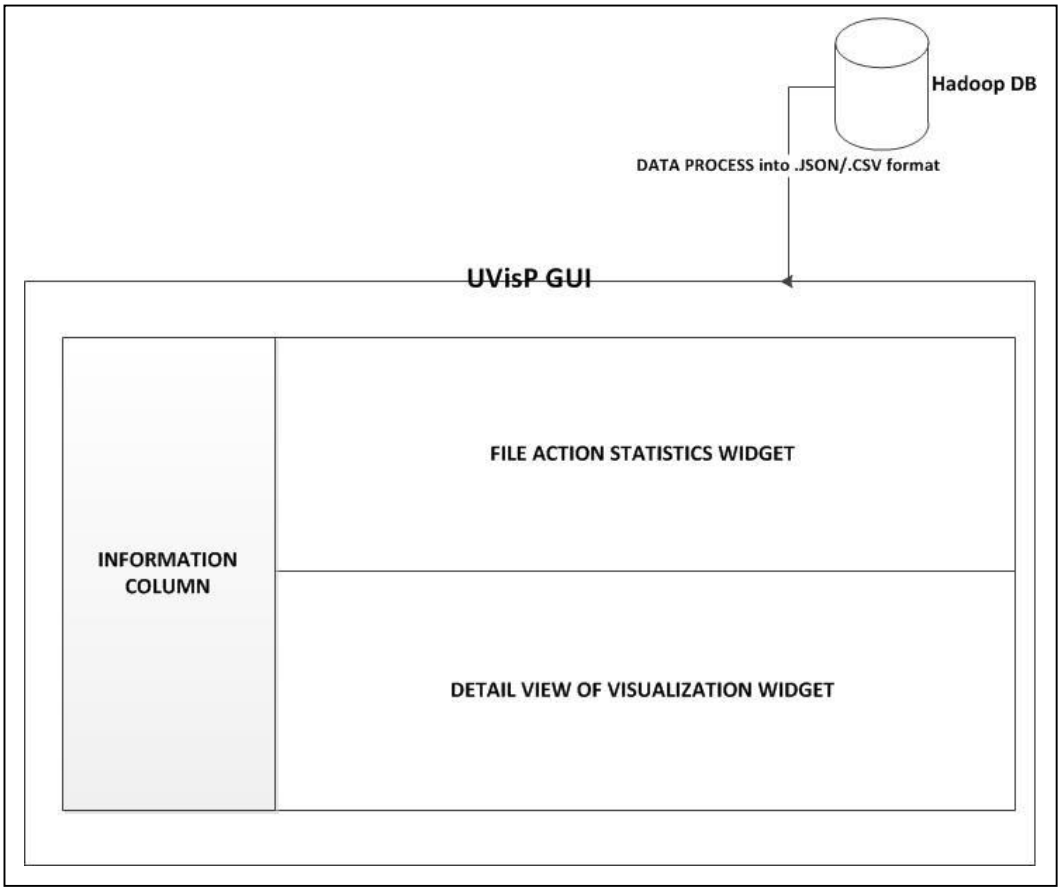


Figure 5.1. A skeleton view of the Visualization Design.

Figure 5.1 displays a simple User interface. The aim of this GUI is to be simple (less clicks required), and user intractable but most importantly informative. Users

are given a page to work with where they can toggle their view and observe two separate visualization outputs.

5.2 UVISP FLOWCHART

This section describes the UVisP flowchart beginning from uploading the Progger dataset until the UVisP application outputs the requested visualization.

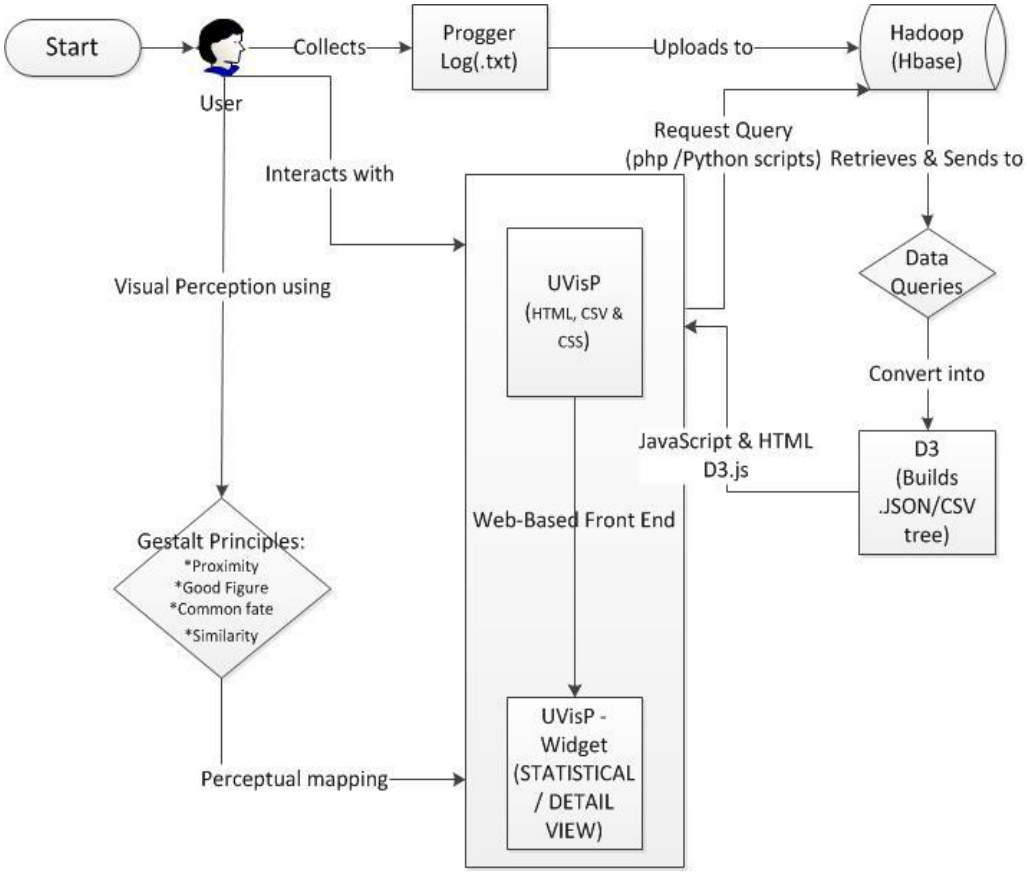


Figure 5.2. A Flowchart diagram of UVisP.

The main components to the UVisP flowchart are:

- User
- Hadoop (DB)
- Web Interface
- D3.js

- Gestalt Principles
- Progger Logs (Datasets)

Users with prior knowledge of the Gestalt Principles, will interact with the web interface to visualize provenance. The emphasis of the flowchart heavily focus on the front-end of the visualization technique. This means, the visualization output must be 'user-centric' and must have the Gestalt Theory of perception presence.

5.2.1 User's Specification and Functions

The user in this thesis are the end-users. They are the reason why this UVisP technique is developed. They are the users of this technique and often we refer to them as the controllers of this technique. Revising the objective of this thesis, we see that, users are empowered by this novel technique to interpret their data's provenance elements and security throughout their data life cycle.

Figure 5.2 shows the user has three roles. Firstly, the user collects and uploads the Progger logs into Hadoop (Hbase). Secondly the user interacts with the UVisP front-end (web interface). In this phase, the user usually provides an input criteria for what type of visualization he/she intends to visualize. For example, setting the timeframe and processing the datatype are some of the input criteria. Finally based on the UVisP visualization output, the user uses the prior Gestalt knowledge to interpret the visualizations and gain knowledge from the visualizations.

5.2.2 Hadoop Storage Infrastructure

The Hadoop infrastructure acts like the storage for this novel technique. Figure 5.2 shows that Hadoop receives the data from the user and stores it inside the Hbase. Hadoop does the processing of the data before sending it to D3 to build the visualizations. Most processing are done using queries within the Hadoop systems. At times, the user through the UVisP system, sends requests using a query to the Hadoop database on what type of data they required for visualization.

5.2.3 UVisP Web Interface

The purpose of this UVisP web interface is to provide the front-end widget for the visualization. It is the point of contact for the user, where all information are presented to the user in the form of visualization. All visualizations are visualized from using the web interface.

5.2.4 D3.js

The D3 (D3.js) is the visualization API. Figure 5.2 shows that D3 has two core functions. First, it receives data from the Database (Hive) by means of queries and converts them into basically two formats, .json and .csv formats. Secondly, since D3.js is written in JavaScript, it is easy for D3 to communicate to the web interface using JavaScript and HTML. D3 is more less the main component of the UVisP where, it has the power to dictate how the visualization should look like when visualized at the front end interface.

5.2.5 Gestalt Principles

The Gestalt Principles as shown in Figure 5.2 plays an important role in the UVisP flow of operations. It acts as an enabler or an instruction-set in the users' mind that helps the user to analyse and visualize the UVisP visualizations at the faster rate. Others can refer to the Gestalt Principles are a set of predefined markers in the UVisP visualizations that enables the users of the UVisP technique to use, gain, extract useful information faster from the UVisP technique.

Users of the UVisP technique have to develop and gain that prior Gestalt knowledge of perception and technically store it in their minds. When users confront the UVisP technique, that prior Gestalt knowledge is transform into the visualization output to help the users to retrieval useful knowledge from the visualization. It links the user to the UVisP technique together to achieve user-centric visualizations.

5.2.6 Progger Logs

Progger logs are the source of data provenance as shown in the Figure 5.2. The logs are generated using Progger. For this thesis, the logs are generated based on the top

user queries identified in section 4.3.4 of this thesis. Additional random scenarios are generated for visualization as well. All scenarios are derived according to data movements and behaviours. The Progger logs is in a form of a log file are stored in the Progger application.

In this thesis, Progger is a third party application which generates the Progger logs. The logs are uploaded into the database system (Hadoop – Hive) for further processing and use by the UVisP technique. These Progger logs have a special function in the UVisP technique. They are the types of visualizations which the UVisP uses to build visualizations, i.e. the Progger logs are transformed into visualization for better understanding.

CHAPTER 6 UVISP IMPLEMENTATION

This section describes the UVisP implementation. This includes the data collection method (Progger), loading and converting the data into JavaScript Object Notation (.json) or comma-separated values (.csv) format [82] [83] [84]. We also provide description of the visualization framework and providing a visual representation to the end-user. We will briefly describe the visualization outputs in the chapter. However all visualization samples are analysed and discussed in Chapter 7 – Evaluation. Most parts of source codes of all implementation can be found in the 'Appendix C' section.

6.1 DATA COLLECTION PROCESS

In sections 6.1 and 6.2, a brief description of the data collection and loading method is outlined as it has already been discussed earlier in Chapter 3. Data collection are from two separate sources. These are:

- The data from the Progger log(s)
- Data collected by carrying out a user-study/questionnaire.

The Progger logs are collected in the form of a text file (progger.log) and the user-study/ questionnaire data is collected in a form of excel sheet.

6.1.1 Progger Data

The dataset used in this thesis are in a log format as shown in appendices D.1 (Progger log format) and appendices D.3 (Progger raw data sample). The Progger log format contains all the system calls that are executed in the kernel. The logs are in a text file format and it contains around 32 different system calls. All system calls have various attributes. For example the "Syscall_Open" has a number of attributes present such as 'Type', 'user', 'PID (processor ID)' and other attributes as well. This system call attributes makes up the content of the Progger logs.

The data used for this thesis is collected by Progger. In order to collect random sets of data, different scenarios are developed to generate various data provenance or file movements. Table 4.1 in Chapter 4 of this thesis shows several different scenarios. The reason for selecting only this number of scenarios is due to the data sizes, and the time allocated to process, analyse and visualize these data scenarios.

All scenarios generated were saved as separate Progger log files (progger.log) as shown in 'Table 4.2'. This allows the user to visualize different visualization. The time frame (duration) of each separate Progger scenarios are based in 'seconds'.

6.1.2 Data Collected from User Study (questionnaire)

Another set of data is the data collected by running a user study/ questionnaire to find out the three top priority user queries when using/storing data in a certain cloud storage service. This data is carried out by using a set of at least one hundred (100) different user questions.

The set is then narrowed down to twenty three priority questions. A set of three top priority user queries are selected using a tally method on paper. Finally these three priority user queries and the different laws of Gestalt theory are associated according to scenarios provided to the users when conducting the study/questionnaire. (*Please refer back to previous Chapter 4, 'Table 4.2' for further explanation*)

6.2 LOADING DATA INTO DATABASE

A database is first created and named. For this thesis, the database name is "ProggerDB". The Progger datasets (progger.log) loaded into the Hadoop (Hive) Database. Each log files generated from the different scenarios are uploaded separately as unique tables of the Progger Database.

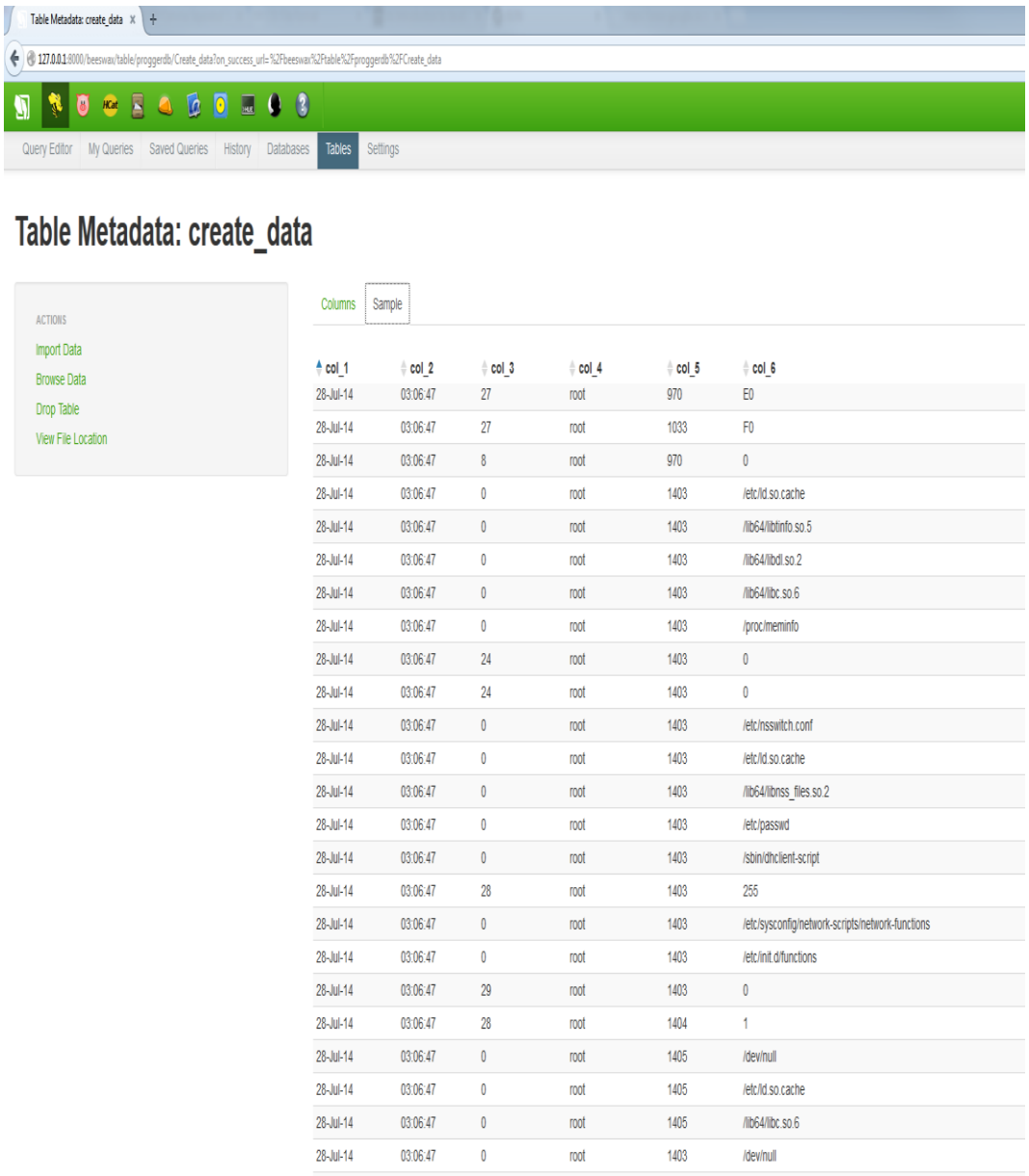


Figure 6.1. A snapshot of Hadoop – Hive Database / table with uploaded data.

Figure 6.1 shows the content of the table "create_data" in a "ProggerDB" database. The dataset has been uploaded into Hadoop – Hive using the upload feature. All queries are executed using these data for the required visualization.

6.3 Converting Raw Progger Data to .json and .csv Format

In order for D3.js to process the data (Progger dataset) from Hadoop the data has to be converted into two formats:

- .json format
- .csv format

D3.js has the features to build and convert datasets into .json and .csv formats according to certain D3 visualization types. Figure 6.2 shows how D3 converts the flat data into a tree format then process it for visualization. There are several phases of carrying out the 'conversion of data into tree'. These phases are as follows:

- Load the external data (either .json/.csv)
- Create node map
- Create the tree array (containing nothing at this stage)
- Identify and add a parent node into the map
- Check and create child array if it does not exist
- Add nodes into the child array.

```
// load the external Progger data from Hive into a .csv file
d3.csv("http://localhost/CirclebarVis/ps.csv", function(error,
data) {

  // ******* Convert flat data into a tree *******
  // create a name: node map
  var dataMap = data.reduce(function(map, node) {
    map[node.name] = node;
    return map;
  }, {});

  // create the tree array
  var treeData = [];
  data.forEach(function(node) {
    // add to parent (Start Node)
    var parent = dataMap[node.parent];
    if (parent) {
      // create child array if it doesn't exist
      (parent.children || (parent.children = []))
        // add node to child array
        .push(node);
    } else {
      // check if parent is null or missing
      treeData.push(node);
    }
  });

  root = treeData[0];
  root.x0 = height / 2;
  root.y0 = 0;

// update root and tree
  update(root);
});
```

Figure 6.2. D3 source code to convert flat data into a tree.

While the above phases are being carried out, there are two other processes happening. These are:

- While building the tree, the is a continues check back to the parent node (root)
- There is always the tree being build, update to the tree, and saving the tree.

6.3.1 .json File Format Sample Output

D3 processes the data and stores them as .json file formats before using them. The reason for using .json file format is because it is a lightweight data-interchange format and it is user-centric, meaning it is easy for people to read and understand. Another reason why D3 chooses the .json file format is because machines and APIs can also parse and generate data. A .json file sample built by D3 and used by UVisP is shown in Figure 6.1. The .json file is easy to understand and it is easy to be read by users. This is because the content and format of the .json file is in a form of a tree and has groupings in them. For example, all data can be grouped by users or 'PID'.


```
{
 "user": "Progger",
 "children": [
  {
   "user": "Syscalls",
   "children": [
    {
     "user": "read",
     "children": [
      {"user": "START", "PID": 8888},
               {"user": "alice", "PID": 1352},
      {"user": "alice", "PID": 1352},
      {"user": "alice", "PID": 1352},
      {"user": "alice", "PID": 1352}
     ]
    },
    {
     "user": "Open",
     "children": [
      {"user": "alice", "PID": 1146},
      {"user": "alice", "PID": 1146},
      {"user": "alice", "PID": 1146},
      {"user": "alice", "PID": 1146},
      {"user": "alice", "PID": 1146}
     ]
    },
    {
     "user": "postfix",
     "children": [
      {"user": "postfix", "PID": 1287}
     ]
```


Figure 6.3. A snapshot of a .json file format from D3.

Finally the .json file structure is important to this thesis because it is structured in a way that allows user-centric capabilities. For example, a treemap like structure already is user-centric without visualizing it. Adding the visualization makes it even more user-centric.

6.3.2 .csv File Format Sample Output

Besides Json, another option is to process the data and store it as a .csv file format. A .csv file convert for D3 processing is shown below:

name	PID	PPID	Type	Filename	User	time	date
0		0	0	/start	nul	03:06:38	28-Jul-14
1	0	1354	27	B0	root	03:06:38	28-Jul-14
2	0	1354	0	/proc/cmc	root	03:06:38	28-Jul-14
3	2	387	22	9.00E+10	root	03:06:38	28-Jul-14
4	2	1354	0	/proc/cmc	root	03:06:38	28-Jul-14
5	2	1354	24	0	root	03:06:38	28-Jul-14
6	2	1146	27	20	haldaemo	03:06:38	28-Jul-14
7	2	1354	22	7.00E+04	root	03:06:38	28-Jul-14
8	2	387	26	0	root	03:06:38	28-Jul-14
9	2	1354	8	0	root	03:06:42	28-Jul-14
10	2	1378	24	9	root	03:06:42	28-Jul-14
11	2	1378	22	C099	root	03:06:42	28-Jul-14
12	2	1378	29	0	root	03:06:42	28-Jul-14
13	2	1402	0	/etc/ld.so	root	03:06:42	28-Jul-14
14	2	1402	0	/lib64/libc	root	03:06:42	28-Jul-14
15	2	1402	0	/usr/lib/lc	root	03:06:47	28-Jul-14
16	2	970	8	0	root	03:06:47	28-Jul-14
17	2	1033	27	F0	root	03:06:47	28-Jul-14
18	2	970	8	56797376	root	03:06:47	28-Jul-14
19	2	970	27	E0	root	03:06:47	28-Jul-14
20	2	1033	27	F0	root	03:06:47	28-Jul-14
21	2	970	8	0	root	03:06:47	28-Jul-14
22	2	1403	0	/etc/ld.so	root	03:06:47	28-Jul-14
23	2	1403	0	/lib64/libt	root	03:06:47	28-Jul-14
24	2	1403	0	/lib64/libc	root	03:06:47	28-Jul-14
25	2	1403	0	/lib64/libc	root	03:06:47	28-Jul-14
26	2	1403	0	/proc/mei	root	03:06:47	28-Jul-14
27	2	1403	24	0	root	03:06:47	28-Jul-14
28	2	1403	24	0	root	03:06:47	28-Jul-14

Figure 6.4. A snapshot of a .csv file format from D3.

Just as the .json file format, .csv offers similar features and capabilities. Based on Figure 6.4, the .csv data file shows certain Progger attributes such as the:

- Name (manually processed unique Identifier)
- PID (Process ID)
- PPID (Parent Process ID)
- Syscall Type
- Filename
- User
- Time
- Data

With the .csv format, data is simply read into the visualization API as well. The logs in Figure 6.4 basically shows the events associated with the user "root". For example, when 'name = 7', the user 'root' executes a 'syscall_S_SENDMSG' call. All syscalls are linked together for the purpose of executing a data scenario such as "creating a new file."

6.4 BUILDING VISUALIZATION USING D3.JS

As discussed in the technique design section (Chapter 5), the UVisP visualizations are built using D3.js. The UVisP is founded on top of HTML, CSS and/or SVG. All data requested by the UVisP front-end will processed, converted into '.json' and '.csv' format. For example a data required for a D3-treemap visualization sends a request for a 'progger.json' from the Progger database.

The advantage of implementing the UVisP visualizations using D3.js is because of the rich D3 library that stores varieties of functions, and codes that can handle all sorts of visualization. For example, mathematical formulas available in D3 makes it easy to accommodate algorithms, therefore allowing developers to produce animated and interactive visualizations.

6.5 UVISP VISUALIZATION SAMPLES

The UVisP Visualization samples are created in a way that are highly influenced by both the graphical design visualization approach and scientific visualization approach. For example, from a graphical design perspective, the application of the 'fundamentals of design' plays a vital role in obtaining a great visualization output. On the other hand, the scientific visualization approach provides the method and prove of why such visualization appears to be in certain forms. The idea of

associating both approaches together is to obtain the best results from both approaches and create user-centric visualization outputs. Below are some samples of visualization.

6.5.1 UVisP Data Statistical (Similarity & Grouping) Visualization

The visualization in Figure 6.5 shows the statistics of how many system call actions are executed using certain file operations. And to be precise, how many times the users' files (data) has been 'touched' and accessed.

The idea of implementing the visualization in a circular approach has two major reasons. The first reason being user-centric in a sense that it has more details in less visual space. The second reason is the inclusion of the 'Gestalt Theory of perception.' Below is a snapshot of the visualization.

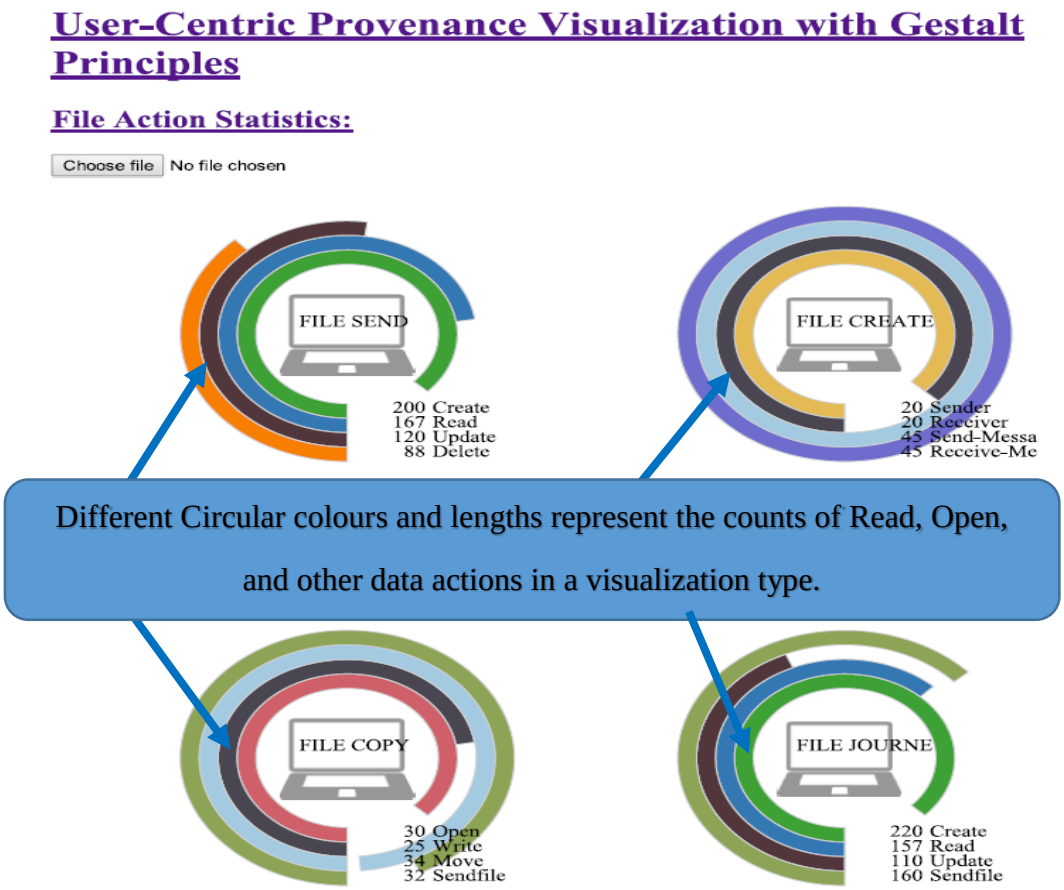


Figure 6.5. An example of the UVisP Visualization showing 'File Action Statistics' and applying the Gestalt law of Similarity, Good Figure and Simplicity.

For this visualization in Figure 6.3, obtaining such visualization requires either a .csv data file or a flat .json data file. Both types of data files are processed by D3.js and the main attributes that the UVisP is aiming for while parsing the data file are:

- Syscall type
- User
- Time
- File name

These main attributes are regarded to as the source of important information, i.e. information containing certain users, files or even the time the event happened. Although the final output only displays the 'syscall' data statistics in the visualization, the other mentioned attributes are required for this visualization to obtain its results. (Note: Refer to Appendix C.1 - Visualization Vis_Type_1 source codes)

6.5.2 UVisP Data Identification (Proximity) Visualization

This visualization sample in Figure 6.6 shows how a user can observe and track files using a D3 `spacetree` visualization approach. It uses a 'scp' scenario created in Progger. It captures all the system call actions that were involved with this action and displays them using visualization. 'Time' and 'syscall' actions are the main identifier for this scenario.

The application of Gestalt Law of Similarity and Proximity is added to this visualization sample to produce that user-centric experience. This enables end-users to be able to interact with the UVisP visualization sample in Figure 6.6 by identifying and selecting interested file to visualize. A 'mouse-over' action on the file will highlight the interested file and shade it with a 'green' colour. Different branches of the '`spacetree`' map can be closed with a 'selection' option from the mouse click. This is to allow users with less detail and better visualization visual area for better user analysis. Below is a snapshot of the visualization, extracted from parts of the Progger log of a 'scp' action of a file from one VM to another VM. (Note: Refer to Appendix C.2 - Visualization Sample 2 source codes)

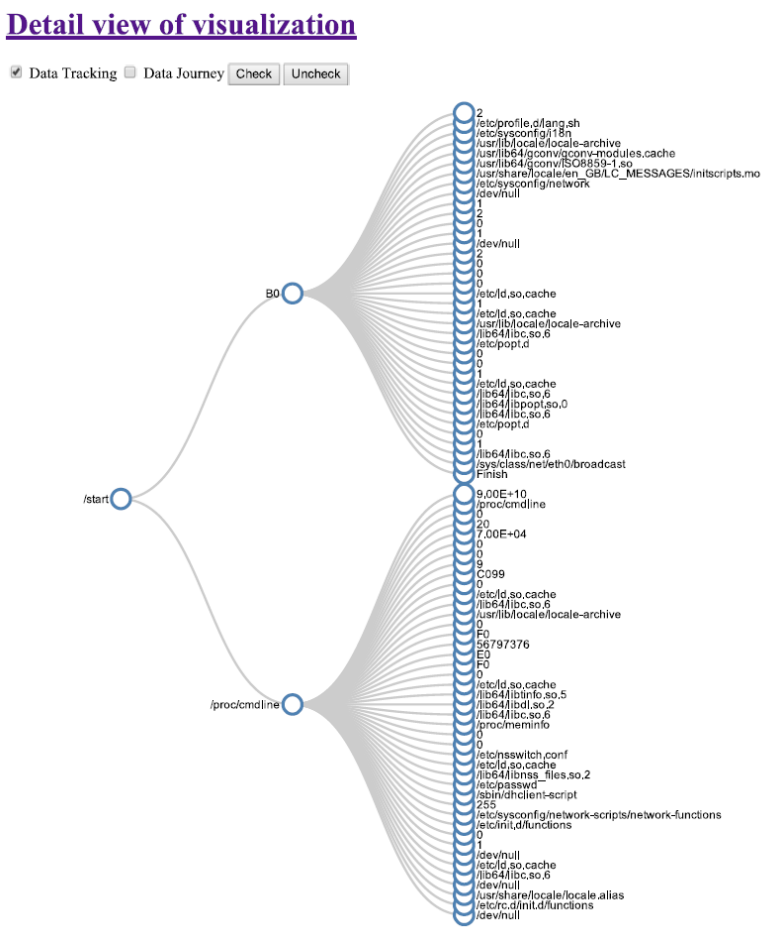


Figure 6.6. An example of the UVisP Visualization showing 'file tracking' and applying the Gestalt law of Similarity and Simplicity.

6.5.3 UVisP Data Tracking (Similarity & Pragnanz) Visualization

UVisP sample in Figure 6.7, is another form of UVisP visualization sample of the visualization in Figure 6.6, however it shows the end-user (person interacting with UVisP) how Progger carries out the logging method of data tracking.

Shaded data nodes together from the UVisP visualizations indicates that file operations are linked together or depended on each other to complete a file operation. It also indicates that most of the kernel activity regarding a certain data scenario or data execution relies on certain Progger syscalls. To an end-user, such visualization allows the user to group the data nodes together and over a certain period of visualizing the same scenario or similar scenarios, the end-user can easily identify the scenario and what data process or action has happened to that particular dataset.

Shaded data nodes far from the main group of spaded data nodes shows that, the file operation are either not related or are just requested once and not multiple times as compared to the others. The closer the shaded data nodes are the more related, linked and depended to each other. Figure 6.7 is a snapshot of the described UVisP visualization sample.

Figure 6.7. UVisP Visualization sample showing file action relationships and applying the Gestalt Law of Similarity, Continuity and Good Figure.

For this visualization, processing the data requires a .csv data file. Once the .csv data file is built, D3.js calls the data file and builds the visualization. (Note: Refer to Appendix C.3 - Visualization 'Vis_Type_3' source codes)

6.5.4 UVisP Data Tracking and Journey (Continuity) Visualization

For the Figure 6.8 visualization sample, we explore the movement of the file according to time, and progger-syscall type. This type of visualization keeps an eye on mainly the “Create”, “Read”, “Write”, “Update”, “Close”, and “Delete” progger-syscalls. However here are other important syscalls that are important but not mentioned are those that handles sending and receiving messages and files.

In order to implement such visualization in Figure 6.8, processing of the data has to be in a .json data file and is important due to several reasons:

- A possibility of tampering with provenance itself
- What visualization details is judged and regarded as user-centric
- Visualization output
- Visualization space relative to user-centricity.

(Note: Refer to Appendix C.4 - Visualization Sample 4 source codes)

Figure 6.8. An example of a 'file-tracking' Visualization from UVisP. It shows how the 'Fileopen.txt' is created.

Overall, processing the data is important because it will dictates the outcome of all the different visualizations discussed in section 6.5.1 through to section 6.5.4. While implementing this technique, the key aspects are as follows:

- Processing data for the visualization is important
- Building the right .json / .csv contributed to the final visualization output. Therefore it is very important that all data added are correct.
- Visualizing different scenarios have challenges to them and linking them to the Gestalt principles can be quiet challenging. This is because most visualization can combine several Laws/principles into one.

CHAPTER 7 EVALUATION

7.1 NOVEL TECHNIQUE FOR VISUALIZING PROVENANCE

In this section, we describe the novel technique used in this thesis, its findings and challenges encountered when carrying out the thesis. We begin with analysing the "D3-Gestalt Model of Association" method used in "Table 3.1". We will then discuss the challenges encountered while carrying out this thesis. Finally, we analyse the visualization findings using the sample visualizations.

7.2 THE D3 – GESTALT MODEL OF ASSOCIATION

The purpose of creating and consolidating all information regarding the Gestalt laws and the different types of Progger scenario is to create a link between the two entities and see what is common in them. The common factors between the two entities provides the user-centric portion of each visualizations in the UVisP technique.

Based on the idea stated above, we carried out the observations as shown in 'Table 4.1.' It indicates that most types of visualization which are highly applicable with applying Gestalt laws /principles are the process calls that combines several system calls together. For example, a Progger operation to create, open, write, and close a file is linked to the 'Gestalt approach' of Similarity, Closure, Continuation, and Proximity. Another example is the 'scp operation' of a file from one VM to another VM uses the Similarity and Proximity approach of Gestalt. However, multiple 'scp operations' from VM to VMs are justifiable with the Similarity, Closure, continuation, and Pragnanz approach of Gestalt.

Note: *'Type of Visualization' in 'Table 4.1' is equivalent to 'System call (syscalls) actions/operation in Progger.*

Type of Visualization	Goal of Visualization	D3 Visualization representation	Gestalt Approach					
			similarity	uniformility	Closure	continuation	Proximity	figure
Open file actions base on different PID	statistics on File access or modification	Treemap / Bubble chart	√					
File created (open), read, write and close	User Report	Force directed Graph/ treemap /heatmap		√		√	√	
copying of a file and saving it to new directory	keeping track of the file movement/activities	Collapsible Tree layoutTree / Bubble	√	√				
moving a file from one directory to another (SCP)	keeping track of the file movement/activities/in case of leakage	Froce directed Graph	√					
renaming a file	statistics, tracking of	Treemap / Bubble chart	√					
rename directory	statistics, tracking of	treemap / bubble chart	√					
Open file -> Read -> Write ->save in new directory	File Process and tracking	Treemap / Bubble chart / heatmap / heatmap	√	√	√			√

Table 7.1. A subsection of Table 4.1, showing the relationship between the Gestalt approach and the types of visualization.

7.3 RESEARCH CHALLENGES

Throughout the course of this thesis, we had encountered challenges. These challenges are as follows:

- **Supporting the full data life cycle (Provenance) with visualization**
  - Data acquisition
    - The ability to constantly carry out data logging and visualizing it was a challenge. In other words, being able to evaluate a full data life cycle and visualize it was a challenge for us.
  - Integration
    - The ability for us to incorporate all data life cycle (Provenance) into one visualization was a challenge. Most visualization portrays parts of the data life cycle and not all.
  - Analysis
    - Analysing the full data life cycle (Provenance) with visualization was a challenge due to several reasons. The first being, the amount of space needed for a user-centric

visualization was not possible. Secondly, analysing multiple datasets and its visualization was time consuming. However, the visualization output provided by UVisP had to be static and covers only parts of the data life cycle and not all.

- Dissemination
  - The ability to spread information using visualization was a challenge as well for us. All visualizations are developed with the idea of targeting the end-users, and most information extracted from the given visualizations were tailored towards end-users.

- **Understanding how end-users perceive patterns (U-C)**

While carrying out this research, an important aspect for this novel technique to fully function, is to "understand the targeted audiences" and how these audiences (in this thesis, audiences are the end-users) perceive patterns and information. Different range of visualization, images are put out to test, and feedbacks are collected.

Based on that, this thesis provided some end-user guidelines when developing this UVisP visualization technique. These are:

- Visualization view must be of reasonable size.
- Visualization must be user-intractable
- Visualization must contain less clicks if required inputs
- Visualization must be simple and attractive, i.e. captures the viewer's eyes and mind.
- Visualization must contain reasonable colours (common colours)

- **Understanding Colour Theory relationships to users, Products and environment**

Based on the end-users requirements, another challenge which we encountered in this thesis is the ability to decide and allocate which colours, colour categories, and how many sets of colours would full represent each visualizations.

One main factor affecting this colour allocation, is the ratio or visualization nodes and actions to the different colours available. Another factor is the types of colours that are by research or theories identified as user-centric colours. Up till today, there has still been the debate of which colours, colour categories are considered "user-centric". Therefore using a colour in any visualizations does have an effect on viewers, especially how they perceive the visualization and draw out useful information.

- **Visualization Outputs**

Due to other contributing aspects faced when building and analysing this novel technique, there has been a lot of views on how the visualization outputs should look like and whether their appearance are considered user-centric or not. With much reviews of past works, we have come to adopt the graphics design's "Fundamentals of Design" approach [65]. The fundamentals of design has six steps. These are as follows:

1. Element of design
2. Principal of design
3. Typography (layout)
4. Semiotic
5. Design Theory (Gestalt Theory)
6. Colour Theory (relationships between Colours)

However, acknowledging that the fundamental of design exist and contribute to the visualization implementation, the type and size of data is another factor that plays a big impact on the visualization outputs. With all those details in hand, not all parts of the design fundamental stand true for

the visualization outputs. Below are some of the visualization outputs that were put to test during the UVisP development. These visualizations are just samples and are implemented to address certain parts of the UVisP technique such as the ‘time’, ‘user’ and ‘file’ factors.

Sample 1: Similarity Visualization

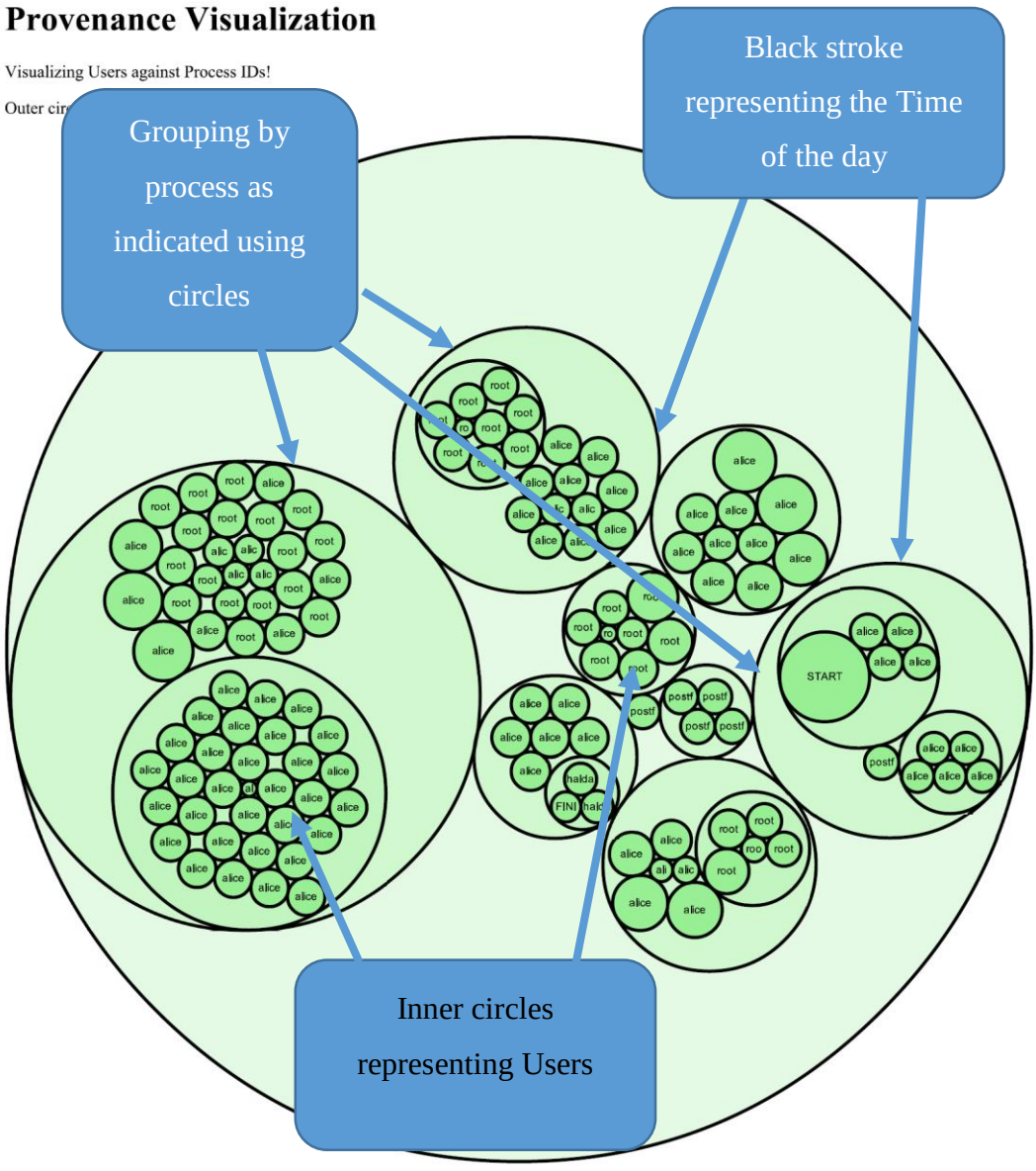


Figure 7.1. An example of an UVisP visualization during the development process.

In Figure 7.1, the visualization aims to address three attributes of provenance. These are:

- User
- Process involved
- Time (what time of the day, did the activity occurred)

As portrayed by the visualization in Figure 7.1, we see that the ‘black-coloured’ stroke outside the circles indicate the time factor of this activity. All processes are

identified by the grouping of the circles. The choice of shading the circles green means, all processes, activities have gone well, and no failure with provenance logging. Although it does contains the mentioned attributes such as user-centric characteristics, and the Gestalt Similarity Law, the viewer (end-user) has very little knowledge drawn out from that visualization. This is due to lack of visibility in visualization and should display more useful information. Apart from understanding the different user processes as illustrated using circles in Figure 7.1, 'time' is the only attribute displayed in the visualization.

Sample 2: Proximity Visualization

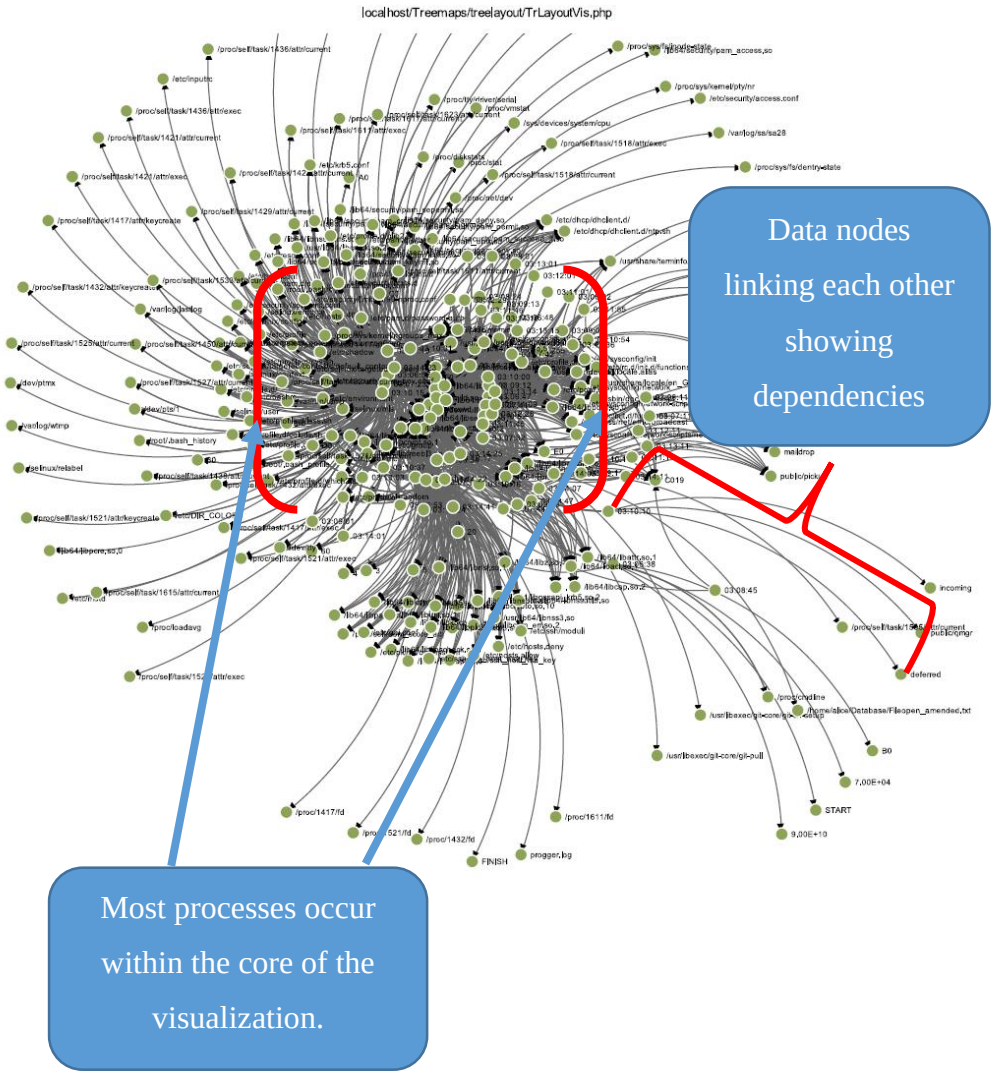


Figure 7.2. An example of an UVisP Pragnanz visualization during the development process.

Using Figure 7.1 Similarity visualization as the prime visualization sample, we produced the Proximity visualization shown in Figure 7.2. This visualization was designed to accommodate all the data from a complete provenance. The purpose

of this is to see how all data nodes (identified using the green coloured nodes) in Figure 7.2, are linked and related together. Patterns and visual-representations of the Gestalt Theory of perception are watched out for.

Having to analyse the Proximity visualization in Figure 7.2, we are able to visualize proximity in the visualization, i.e. nodes appearing together, tend to be in a group. By clicking on any nodes with multiple links, one can visualize the related nodes which are linked together.

We conclude that the Figure 7.2 Proximity visualization has both the user-centric and the Gestalt components in it, however, data congestion in the visualization is the negative factor of this type of visualization.

Sample 3: Common Fate Visualization

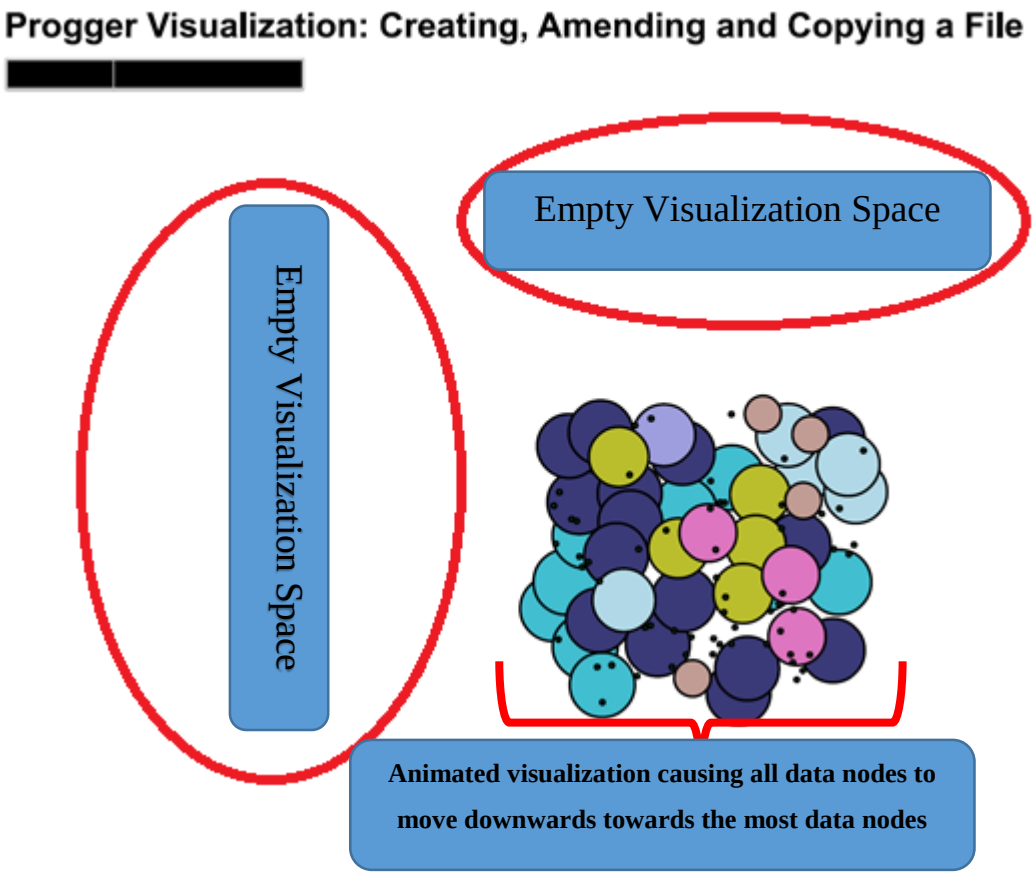


Figure 7.3. An example of an UVisP Common Fate visualization during the development process.

Since the Proximity visualization in Figure 7.2 gathered for all the data of the provenance, we opt to visualize selected portions of the data and not all in a given time. With the available features of D3.js, we added the animation features to the 'Common Fate' visualization to see if any related nodes in the dataset provided would show relations by moving/attracted towards each other. With the moving and

attracting motion capabilities of the animated D3 library, we tried to visualize parts of the dataset and observe for any new characteristics within the visualization. Figure 7.3 visualization shows that the ratio of similar system calls to the others make a difference in the movement of the data nodes. For example, if there are 20 “Syscall_Open” data nodes, 5 “Syscall_Creat” data nodes, and 3 “Syscall_Unlink” data nodes, then the final motion and force of attraction will sway towards the “Syscall_Open” since it has a total of 20 data nodes in the visualization. This motion behaviour does affect the final arrangement of the visualization as seen in Figure 7.3. The empty visualization spaces which are indicated with ‘red’ oval colours as shown in Figure 7.3 appeared due to several reasons, i.e. the coordinates of the final visualization space, and the force-layout from the data nodes in the visualization.

- **Understanding how D3 works and its library**

  Another challenge we have encountered is the visualization API used for this types of visualization which is D3.js. The choice for this thesis are:

  - D3.js (& Library)
  - Javascript
  - HTML
  - SVG
  - CSS

  Due to the timeframe allocated for this thesis, and constrains on having to learn and understand D3.js and its library was a challenge. D3.js has its own ways, methods of adding, rendering and calling functions into the program. However, the ability to re-use the D3 library and codes provided the way out to obtaining these visualizations. Apart from the challenge, D3.js was the choice for this thesis due to its features and capabilities that is provides for visualizations. It also has the characteristics of incorporating itself with the Gestalt theory of perception, which makes it a good choice of API for our thesis.

- **Justifying the links between Gestalt and D3 Visualization**

The choice of using D3 and Gestalt, brings a challenge that needs to be further analysed. The need for an algorithm is yet to be achieved and would provide a theoretical answer to this D3 – Gestalt novel Visualization technique. At the current thesis status, practical methods of integrating D3 and the Gestalt theory are the practical explanation of this novel technique, i.e. a user-centric novel visualization technique.

- **Data retrieving speed for visualization.**

One of the biggest challenge was to find a method to retrieve data fast from the Database and process them fast enough for the front-end application would display them. Due to the data type, the data growth per seconds and storage method, the time in which a set of data is called and converted into either '.json' or '.csv' format was long enough to be noticed when using the UVisP technique. Future works will be carried out on this to improve the speed and also to provide a "data processor" or filter to sort, process, and arrange data into relevant file type format ready for call and execution.

Overall, constructing these visualizations for different scenarios has its challenges. Processing the data and storing it in a database is a challenge since the raw Progger data requires manual processing and analysis before creating scripts to write them into the database of even using both MySQL queries and Pig scripts. Another challenge that we have encountered with these visualization is due to the type of data provided and the size of the data, it is difficult to produce a visualization that will display all required information for the end-users to visualize. With most cases of the visualization, only portions of the data is being used to visualize. On that note, having to charge which information (data attribute) to display and which not to display is a challenge. Finally, with these challenges in place, the design of the UVisP technique has been continually tweaked to the point where we are able to produce better and improved visualizations for the end-users to visualize and extract useful provenance knowledge from them.

7.4 ADVANTAGES OF PROOF OF CONCEPT

It is the visualization challenges that brings out the quality of any technique. For this thesis, the UVisP technique can now be considered user-centric and easy to collaborate with, i.e. less click to visualize different provenance visualizations. It provides an easy-to-understand web interface that sums up all information drawn from the Progger dataset (progger.log) into a one page visualization output for users to observe and understand relatively.

In addition to the web interface being simple and easy to use, UVisP offers a complete transformation of raw data into understandable visualizations that is attractive to the end-users when using the UVisP technique. The attractive features of the UVisP are:

- Less space needed for a user-centric visual representation.
- Less clicks to visualize provenance, means easy to use.
- The inclusion of Gestalt into UVisP makes it user-centric, i.e. end-user perception triggers the users to extract knowledgeable information instantly.
- Provides the perceptual key(s) in the visualizations that allows the end-users not only to understand the visualizations better but also perceive the visualizations in their own ways, giving them the sense of satisfaction when using the UVisP technique.

7.5 VISUALIZATION ANALYSIS

This section of the evaluation presents the different types of visualization produced for this thesis, what they represent and a brief analysis on the different visualizations.

7.5.1 The Gestalt Proximity Visualization

Associating the existing visualization with the Gestalt law of Proximity, we see the ability to visualize the treemap diagram and being able to "mouse-over" the selected nodes provides a brief summary of what the file is. This makes it user-centric and intractable. In Figure 7.1, executing a "mouse-over" action from the mouse over the nodes will highlight the nodes in green and provide a summary details of that particular node in the visualization. The end-user has the option to toggle between any files (nodes) to observe and analyse the targeted node. The ability to close a

treemap branch in this visualization, allows better analysis on selected nodes. All nodes in this visualization represent a syscall action.

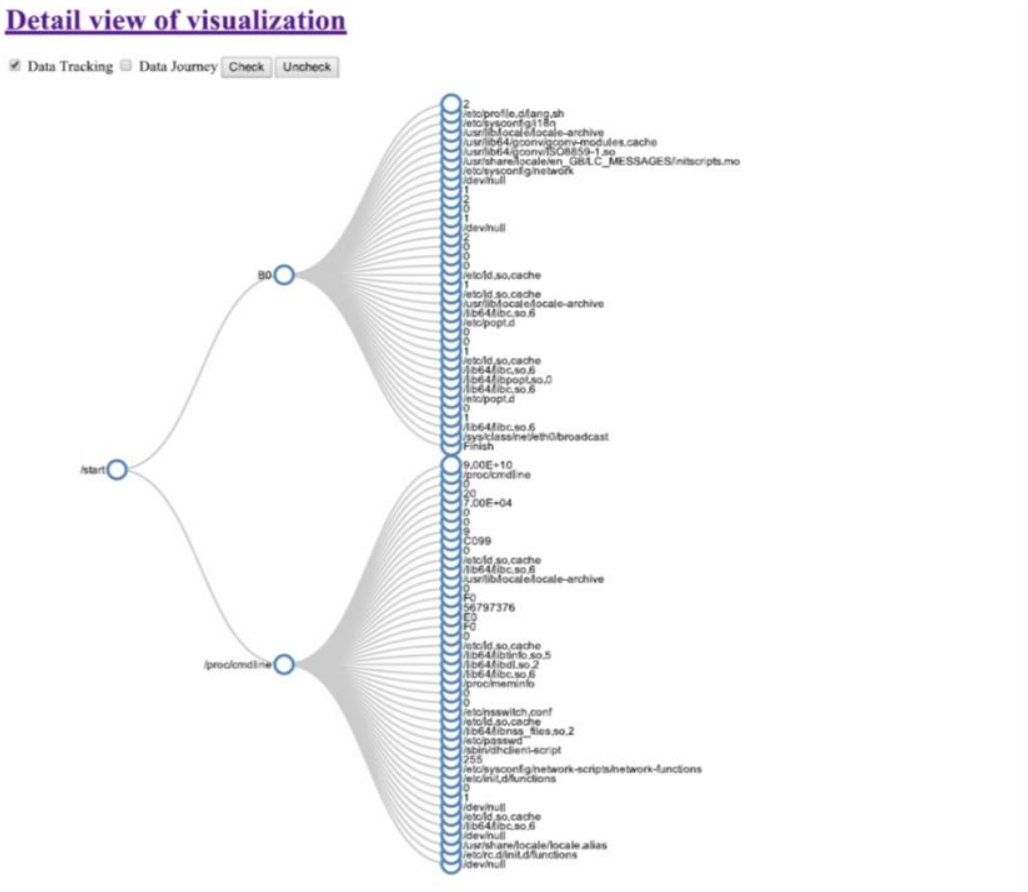


Figure 7.4. An example of the tracking the data of selected choice.

7.5.2 The Gestalt Simi-&-Pragnanz Visualization

This visualization is based on the *"File-Create", "File-Amend" and "File – Save/Close"* scenario in the Progger utility. The time required for Progger to execute this actions is in seconds. In this visualization we can observe that, all nodes in blue colour represents all syscall executed during the operation of the Progger scenario as mentioned above.

Having prior knowledge of the Gestalt theory of perception, the users can visualize and try to group the nodes together or even form patterns. In Figure 7.5, the end-users can have the option of grouping the files as indicated using the Red-coloured squares. Others can just visualize the spiral pattern and the congested nodes within

the centre of visualization. This also indicate that this files are related and are connected or require one another for a complete execution of tasks.

In Figure 7.6, a red and brown coloured circles are drawn over the visualization to identify the spiral pattern that was form in this visualization. Files identified using the Green boxes are the actual files (.txt). The order at which the ".txt" file are created and used are as shown in Figure 7.6 with coloured blue squares. While analysing the visualization, we notice that the outer spiral (Brown in colour sets of nodes) are the first to be executed. Then the red coloured spiral nodes are called and executed, and slowly the nodes in the centre are called and executed. Based on this observations, we argue that, the Gestalt Law of "Similarity" and "Pragnanz" is applied by the end-user to visualize Figure 7.5 and Figure 7.6.

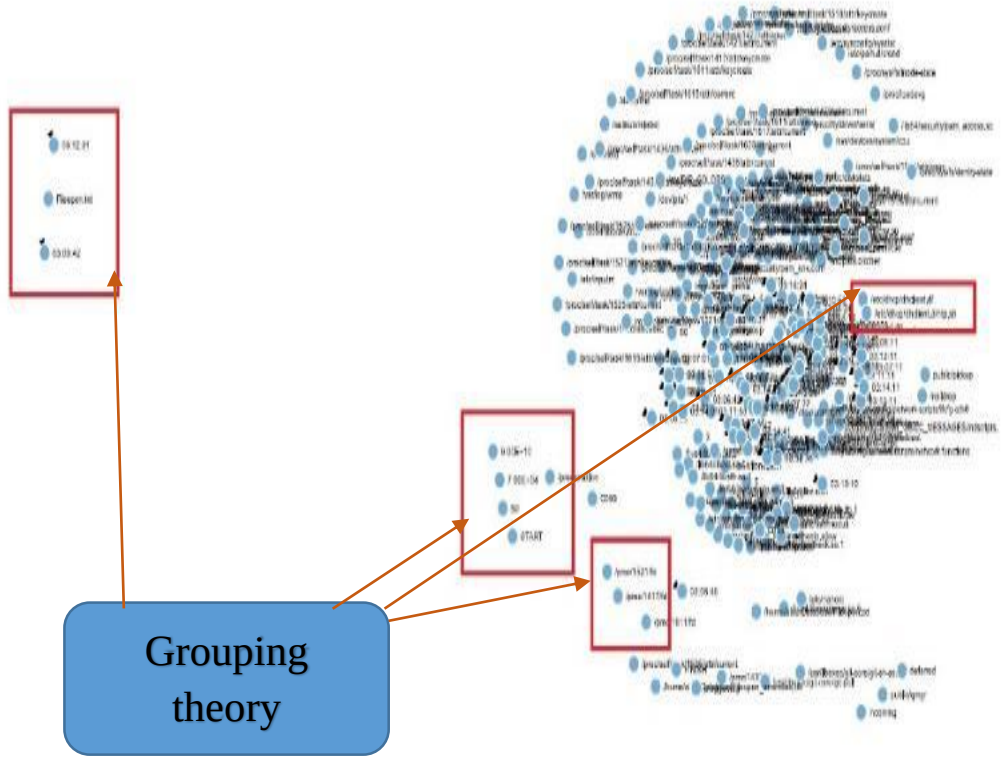


Figure 7.5. Visualization with Similarity and Pragnanz.

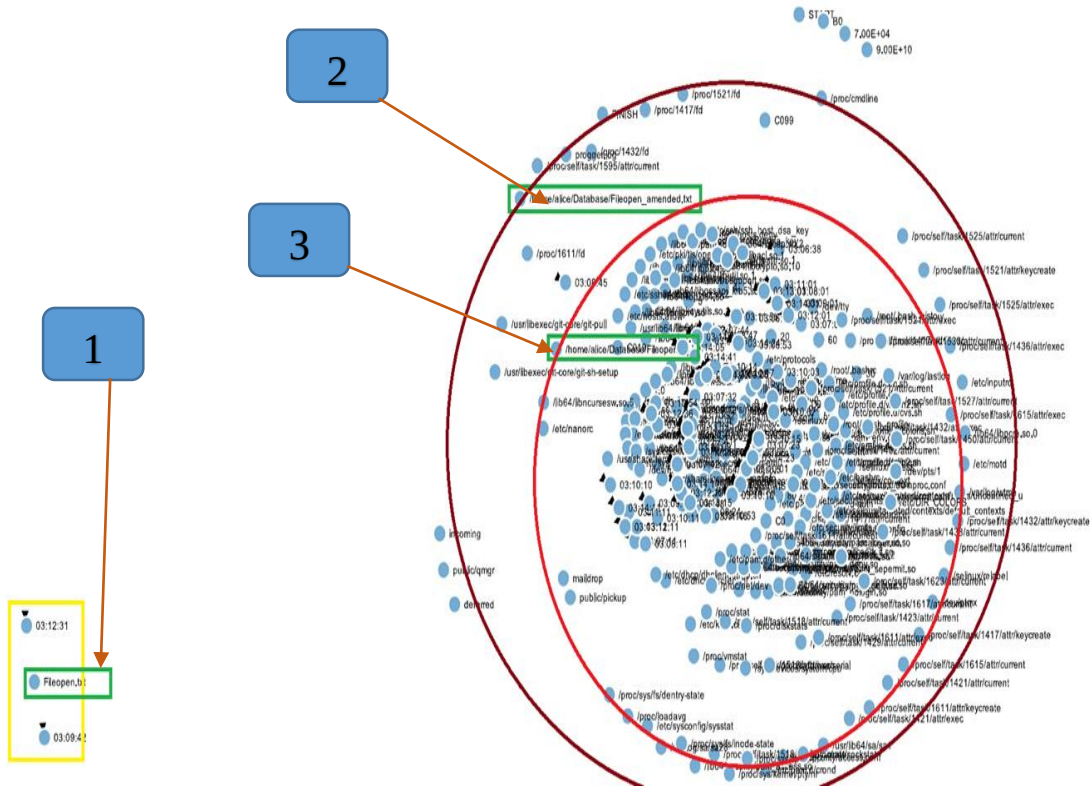


Figure 7.6. This visualization shows the movement of files. Tracking of files are indicated in this visualization which makes the visualization.

7.5.3 The Gestalt Continuity Visualization

This visualization is based on the *"File-Create", "File-Amend" and "File – Save/Close"* scenario in the Progger utility. The time required for Progger to execute this actions is in seconds. Due to this dataset being very large, only parts of the datasets are visualized. The dataset had to be processed before called into the visualization utility for visualization outputs.

In this visualization, we can identify patterns as the processing of the file is executed. Referring to Figure 7.7, we can observe that there are certain kernel files that are repeatedly called whenever a document file is created or open. For example, in Figure 7.7, the *"Fileopen.txt"* is created just after a repeated execution of kernel files such as *"56797376"* and also several *"/etc/passwd"* are called and executed as well. Moving on to observing the "syscall_UPDATE" which executes the file amendments (changes done on "FileOpen.txt"), we also observe similar repeated files being called before the update call was called for changes. Therefore we argue that, visualizing this provenance, does indicate patterns and in terms of the Gestalt

theory of perception, the Law of "Continuity" and "Similarity" is executed here by the end-user to observe the visualization patterns.

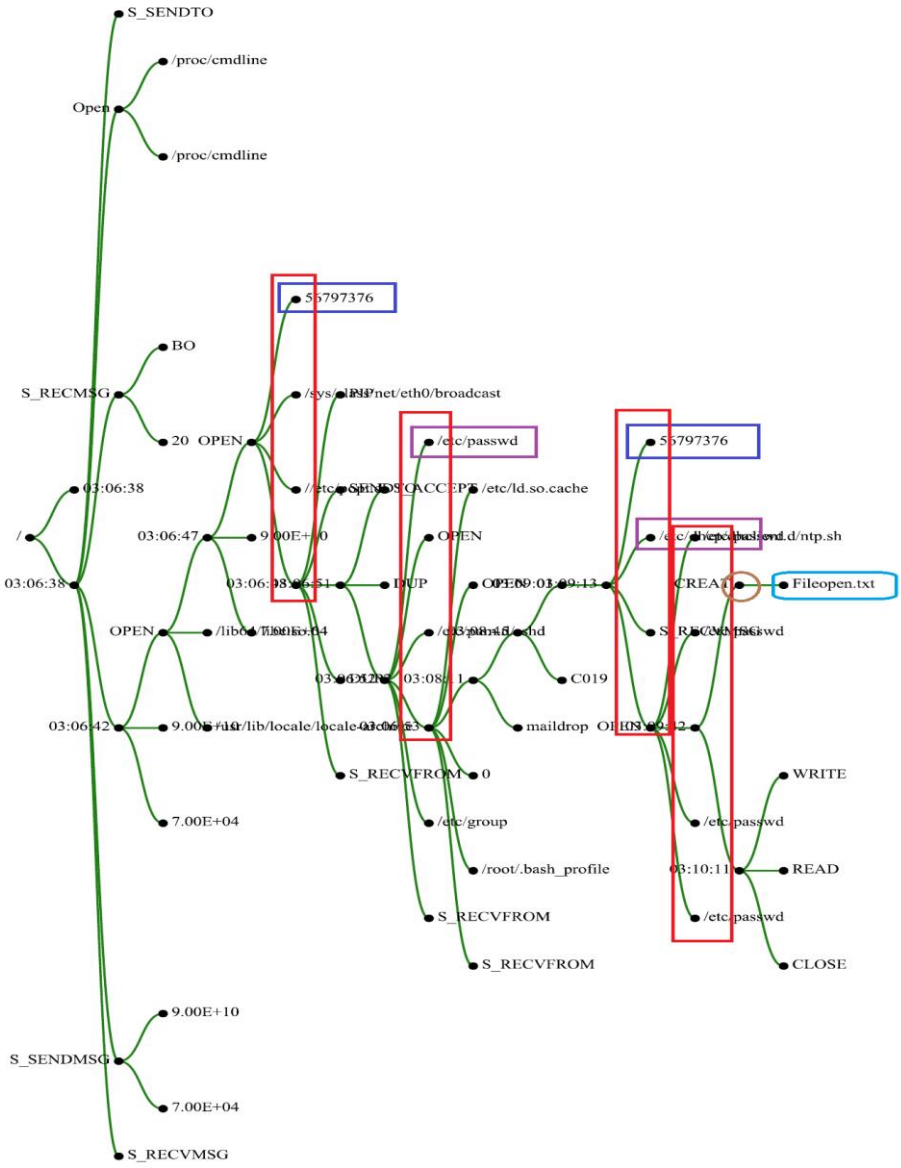


Figure 7.7. A Gestalt Continuity Visualization for the "File-Create" Scenario.

7.6. UVISP USER TEST

For this section we have carried out some testing with the UVisP technique to see how it performs. Our results are summarized and presented in Table 7.2.

Visualization Type	Informative (scale of 1 – 5) 5 is most informative	Easy/hard to understand scale of 1 – 5) 5 is hardest	Gestalt (G.S-V, G.P-V, G.Pr-V, G.CF-V)	Average Time of visualizing (sec)
Vis_Type_1	4	2	G.S-V, G.CF-V	< 60
Vis_Type_2	4	1	G.P-V	120
Vis_Type_3	5	1	G.Pr-V	< 60
Vis_Type_4	2	3	G.S-V	180
Vis_Type_5	4	3	G.S-V, G.P-V, G.Pr-V, G.CF-V	180
Vis_Type_6	3	3	G.S-V, G.Pr-V, G.CF-V	120

Table 7.2. User Visualization test findings and summary.

7.6.1 General Observations and Findings

While carrying out the visualization test with the UVisP technique, we have made general observations on how users approach this UVisP technique. We also observed how they respond to the visualization samples with respect to such factors:

- How informative the visualizations are?
- The level of difficulties in understanding the visualizations
- How well the Gestalt prior knowledge is understood by the users, and how well it is applied to the sample visualizations.
- Finally the time factor, i.e. how long it takes a user to observe, select, understand useful information from the visualizations.

7.6.1.1 UVisP User Test Phases

However before we describe the factors mentioned above, we will provide how the test is been carried out. The user test is carried out in three phases. Phase 1 provides the users with the background information of the current problem with visualizations. This covers showing the current log (provenance) or raw data, and explaining the aim of this thesis.

Phase 2 provides the Gestalts theory of perception (Gestalt Laws / Principles). In this phases users are provided with the Gestalt Laws and visualizations illustrating these Laws. The aim of this phase is to allow users to fully develop skills to identify different visualizations and extract useful information and knowledge from them.

Phase 3 provides the users with the different sample visualization with the aim of analysing the user's response to the visualizations. Users are tested and observed carefully on how fast they understand the given visualization and what are the key areas in the visualizations that help them understand the visualizations faster.

In the final phase of this user test is to see which Gestalts Law of perception is being associated with the given visualizations.

The sample of this user test is found in Appendix E.6.

7.6.1.2 Visualization vs Information Level / Understanding

While users engage in the visualizations, to see how informative the visualizations are, we've noticed that, users who understand the Gestalts theory of perception well knew what they are looking for and have a higher chance of understanding the visualizations better. Another behaviour that was observed from the users is, understanding what the purpose of the 'perceptual key' in visualizations. With that skill and knowledge in them, visualizations are easily identified. Finally, having both the 'prior knowledge of 'Gestalts theory of perception' and the perceptual key (s) present, the visualizations are identified (identifying the visualization takes less time) easily and comfortably.

7.6.1.3 Visualizations vs Prior Knowledge Gestalt's Theory of Perception

Understanding Gestalt's theory of perception has both advantages and disadvantages. The advantages is all theories presented allows users to relate better and faster to the sample provenance visualizations. However the down side of having prior knowledge of Gestalt's theory of perception is, users who find it difficult to spot the differences between the visualizations do not have the understanding of provenance visualization.

7.6.1.4 Visualization vs Time

During the user test, the 'time' taken from observing the users when trying to understand different visualizations is very important. 'Time' in this thesis is regarded as a positive indicator for promising results. This proved correct. Findings shows that most users took between 60 seconds to 180 seconds to understand and draw conclusion from the given visualization.

7.6.1.5 User Study Analysis of Findings

Based on the findings from 'Table 7.2' and the end-users, the common user-centric trends are:

- Visualizations that contain more than one colour is better and works well by attracting the users to take time and study the visualizations.
- Visualizations with known and common features are easier to identify. For example, terms like 'open' and 'create' helped the users to understand Vis_Type_1 and Vis_Type_3 better.

7.6.1.4 User Study Analysis on Feedback

While carrying out the 'UVisP user Questionnaire and feedback' there are quite a number of interesting feedbacks. Below are the feedbacks from users who took the visualization user study (*Refer to Appendices 4 for the related Visualizations (Vis_Type_1, and etc,*) :

Positive Feedback:

- Vis_Type_2 and Vis_Type_3 are the most informative and easy to understand. Visualization features in these two visualization samples shows that they are categorized into "Common Fate (G.CF-V)" and Pragnanz (G.Pr-V)
- Vis_Type_1 has a lot of information and can identify the activities in the tree diagram. However, it is not as easy as Vis_Type_2 and Vis_Type_3.
- Overall, Vis_Type_3 has captured a lot of the users of the UVisP technique with an easy to understand and very informative by users of this knowledge.

- Using multiple colours for the visualization in general, captures the users of the UVisP technique.

Negative Feedback:

- Vis_Type_6 is does not mean anything to the users and not much informative. The colours helped identified the processes, however, the numbers do not mean anything at this stage.
- The application of Gestalt at this stage, is presented in a indistinct manner in Vis_Type_1, Vis_Type_3 and Vis_Type_4. More prior knowledge is required to fully understand the application of Gestalt into the UVisP visualizations.
- Other feedbacks are focus on the colours applied. A handful of users preferred to pick their own colours if they are given a chance to use the Technique. In other words, users have their own colour preferences.
- Overall, not all visualizations are regarded as user-centric to all users of the UVisP technique. This is due to how much prior knowledge of Gestalt they acquire and also whether they are categorized as end-users or not.

▪

CHAPTER 8 CONCLUSION

8.1 UVISP – A NOVEL VISUALIZATION TECHNIQUE

Throughout the thesis, we aimed to find and propose techniques that will transform provenance in the form of raw data (logs) into a final output that end-users could easily understand. Visualization is proposed solution. With that in mind, we tried to understand relevant user-centric theories and frameworks applied to visualization. Based on that we have analysed several visualization theories and have selected "Gestalts' theory of perception." The reason behind this choice is that Gestalts' theory of perception shares the user-centric behaviours and approaches that empowers the users to relate and understand varieties of visualizations better.

In addition to these visualization theories such as Gestalts theory of perception, visualizing provenance elements and granularities of systems are to the users' advantage.

This thesis presents UVisP (User-centric Visualization of Data Provenance with Gestalt), a novel user-centric visualization technique to analyse existing visualization which empowers users to interpret their data's provenance elements and security throughout their data life cycle. The inclusion of the Gestalt theory of perception to the technique, provides a range of visualization that allows the end-users to visualize and interact with an aim to obtain useful information such as tracking data provenance.

Visualizing data provenance in general is popular and a lot of research has been done in the field (Recall Chapters 2.3.2 and 2.3.3). Existing provenance visualization are mainly in the form of logs and high level visualizations that targets technical people for the purpose of exploratory purposed. However, analysis shows there are less user-centric visualization techniques for visualizing provenance. The need for such visualization will enable users to extend their knowledge and empower the users to change and adapt to new visualizations.

In order to offer good and user-centric visualizations, factors such as the infrastructure of the entire visualization framework has to be at its best. Therefore this thesis had chosen D3.js as the visualization API, Hadoop – Hive as the database

and storage repository. Our choice for selecting both D3.js and Hadoop are basically because of performance, scalability, affordability and overall, the user-centric characteristics.

The integration of Gestalts' theory of perception into the UVisP technique allows users to develop prior knowledge of the Gestalts' theory for the purpose of mentally comparing and using them as bench-marks to the visualizations offered by the UVisP technique. This UVisP technique proves to work well for users who have the prior Gestalts theory of perception knowledge and identifying 'perceptual key (s).' However, for the Gestalt theory of perception and the perceptual key (s) to fully function, there has to be ways of integrating the Gestalt Laws / principles with the types of visualization. For the case of this thesis, the types of visualizations are the scenarios generated from Progger.

The types of visualization are the core aspects of UVisP technique since it is extracted from Progger as the source of provenance. Based on the types of visualizations, the UVisP technique uses D3 to link them up based on the structures if the existing D3 visualization templates (such as Treemaps) with the Gestalt theory of perception. The process of linking them up is referred to as the UVisP's user-centric (U-C) methodology. The UVisP technique with its methodology provides the procedure and steps on who to obtain the final user-centric visualizations.

While developing the UVisP technique, users have to test to see if it has positive feedbacks from it. In this thesis, we have carried out two user surveys. The first survey is to identify the types of visualization and link them with the corresponding Gestalts Laws. This will link the type of D3 visualization. Another survey is the 'user test' survey which is purposely to identify how much relation between the sample D3 visualizations and the users understanding from the visualization. Users are asked to identify key parts of the visualizations and explain what type of visualization category it would fit in with respect to the Gestalt Laws. For example, Vis_Type_1 is a 'Proximity' sample visualization, where users are required to identify files from such visualization.

With UVisP technique in place, end-users also have the opportunity to interact with different log files in the form of Visualization. These logs represents the type of

visualizations, which are then transformed and integrated with the Gestalt theory of perception to produce a range of user-centric visualizations.

Finally this thesis provides a user-centric visualization technique that offers “less clicks” front-end to visualize provenance. Results from the UVisP technique indicated that Vis_Type_1, Vis_Type_2, and Vis_Type_3 are considered user-centric. Therefore we conclude that, the novel technique is user-centric in a sense that the visualizations presented are easy to understand, informative and does indicate the Gestalt’s Laws of ‘Proximity’, ‘Similarity’, ‘Pragnanz’ and ‘Common Fate.' It is our hope that this work will form the basis of several new research directions which will be discuss in the following section.

8.2 Future Work

Conducting this research revealed the following potential further research areas:

- **Provenance Database System:**

  The need to store data into a well-managed Database system is required. This will allow fast queries requests from the front-end visualization technique. For example, such database systems as ‘CouchDB’ and ‘Hadoop’ should be further studied for better implementation.

- **Processing time:**

  A Data processor to speed up the time between processing the data and loading it into D3 for visualization. For example caching and distributed algorithms can be implemented.

- **A Middle-man Processor / filter:**

  The purpose to have a middle-man processor/ filter between the front-end interface and the storage database, is to sort out and process the datasets (Provenance) without tampering with the dataset. The process data is then channelled into the front end for visualization.

REFERENCES

[1] P. Chen and B. Plale, "Abstract: Visualizing Large Scale Scientific Data Provenance," in *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:*, 2012, pp. 1385–1386.

[2] J. Wagemans, J. H. Elder, M. Kubovy, S. E. Palmer, M. A. Peterson, M. Singh, and R. von der Heydt, "A century of Gestalt psychology in visual perception: I. Perceptual grouping and figure–ground organization.," *Psychological Bulletin*, vol. 138, no. 6, pp. 1172–1217, 2012.

[3] B. G. Tudorica and C. Bucur, "A comparison between several no sql databases with comments and notes," 22:41:34 UTC.

[4] J. R. Hauser and F. S. Koppelman, "Alternative Perceptual Mapping Techniques: Relative Accuracy and Usefulness," *Journal of Marketing Research*, vol. 16, no. 4, pp. 495–506, Nov. 1979.

[5] Y. S. Tan, E. Leonardi, B. Ma, R. Li, M. Kirchberg, G. L. K. Khoon, R. K. L. Ko, B. S. Lee, and T. H. G. Guang, "AnalyticaR: A visualization Technique for Visual Data Exploration," in *Infovis*.

[6] D. Cram, B. Fuchs, Y. Prie, and A. Mille, "An approach to user-centric context-aware assistance based on interaction traces," University of Lyon 42 - France, Mar. 2008.

[7] A. Rutledge, "Andy Rutledge - Gestalt Principles of Perception - 4: Common Fate," 13-Jul-2009. [Online]. Available: http://www.andyrutledge.com/common-fate.php. [Accessed: 08-Jan-2015].

[8] S. Vezzani, B. F. M. Marino, and E. Giora, "An early history of the Gestalt factors of organisation," *Perception*, vol. 41, no. 2, pp. 148–167, 2012.

[9] R. C. Taylor, "An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics," *BMC Bioinformatics*, vol. 11, no. Suppl 12, p. S1, Dec. 2010.

[10] "Apache Pig," *Hortonworks*. [Online]. Available: http://hortonworks.com/hadoop/pig/. [Accessed: 01-Feb-2015].

[11] R. Marty, *Applied Security Visualization*. Addison-Wesley Professional, 2008.

[12] I. Wassink, O. Kulyk, B. van Dijk, G. van der Veer, and P. van der Vet, "Applying a User-centered Approach to Interactive Visualisation Design," in *Trends in Interactive Visualization*, R. Liere, T. Adriaansen, and E. Zudilova-Seinstra, Eds. Springer London, 2009, pp. 175–199.

[13] Y. L. Simmhan, B. Plale, and D. Gannon, "A Survey of Data Provenance in e-Science," *SIGMOD Rec.*, vol. 34, no. 3, pp. 31–36, Sep. 2005.

[14] Y. Simmhan, B. Plale, and D. Gannon, "A Survey of Data Provenance Techniques. Technical Report IUB-CS-TR618."

[15] A. Komlodi, P. Rheingans, U. Ayachit, J. R. Goodall, and A. Joshi, "A user-centered look at glyph-based security visualization," in *IEEE Workshop on Visualization for Computer Security, 2005. (VizSEC 05)*, 2005, pp. 21–28.

[16] "Binding data — Scott Murray — alignedleft," *Interactive Data Visualization with d3 tutorials*. [Online]. Available: http://alignedleft.com/tutorials/d3/binding-data. [Accessed: 24-Dec-2014].

[17] P. Podila, "Building a tree diagram in D3.js - Pixel-in-Gene," 20-Jul-2011. [Online]. Available: http://blog.pixelingene.com/2011/07/building-a-tree-diagram-in-d3-js/. [Accessed: 10-Jan-2015].

[18] M. Ankerst, D. A. Keim, and H.-P. Kriegel, "Circle Segments : A Technique for Visually Exploring Large Multidimensional Data Sets," 1996.

[19] C. D. Green, "Classics in the History of Psychology -- Wertheimer (1923)." [Online]. Available: http://psychclassics.asu.edu/Wertheimer/Forms/forms.htm. [Accessed: 04-Jan-2015].

[20] K. Minarik, "CouchDB – A Database for the Web," 24-Sep-2010.

[21] M. Bostock, "D3.js - Data-Driven Documents." [Online]. Available: http://d3js.org/. [Accessed: 24-Dec-2014].

[22] O. Curé, R. Hecht, C. L. Duc, and M. Lamolle, "Data Integration over NoSQL Stores Using Access Path Based Mappings," in *Database and Expert Systems Applications*, A. Hameurlain, S. W. Liddle, K.-D. Schewe, and X. Zhou, Eds. Springer Berlin Heidelberg, 2011, pp. 481–495.

[23] P. Buneman, S. Khanna, and W. C. Tan, "Data Provenance: Some Basic Issues," in *Proceedings of the 20th Conference on Foundations of Software Technology and Theoretical Computer Science*, London, UK, UK, 2000, pp. 87–93.

[24] "Dedicated VMware vCloud: A Managed Private Cloud Powered by VMware Technology and Rackspace Fanatical Support." [Online]. Available: http://www.rackspace.com/blog/2009/11/09/nosql-ecosystem/. [Accessed: 29-Jan-2015].

[25] "Design Principles: Visual Perception And The Principles Of Gestalt," *Smashing Magazine*. [Online]. Available: http://www.smashingmagazine.com/2014/03/28/design-principles-visual-perception-and-the-principles-of-gestalt/. [Accessed: 04-Jan-2015].

[26] 19 August 2010 Thursday, "Document Databases Compared: CouchDB, MongoDB, RavenDB." [Online]. Available: http://nosql.mypopescu.com/post/978742866/document-databases-compared-couchdb-mongodb. [Accessed: 30-Jan-2015].

[27] S. Vieira, "Easily create stunning animated charts with Chart.js," *Webdesigner Depot*, 04-Nov-2013. .

[28] R. K. L. Ko, P. Jagadpramana, and B. S. Lee, "Flogger: A File-Centric Logger for Monitoring File Access and Transfers Within Cloud Computing Environments," in *Proceedings of the 2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, Washington, DC, USA, 2011, pp. 765–771.

[29] "Gestalt Psychology in German Culture, 1890–1967 Holism and the Quest for Objectivity | History of science and technology," *Cambridge University Press*.

[Online]. Available: http://www.cambridge.org/nz/academic/subjects/history/history-science-and-technology/gestalt-psychology-german-culture-18901967-holism-and-quest-objectivity. [Accessed: 04-Jan-2015].

[30] "Gestalt Theory," 08:13:41 UTC.

[31] K. Smith-Gratto and Mercedes M. Fisher, "Gestalt Theory: A Foundation for Instructional Screen Design," *Journal of Educational Technology Systems*, vol. 27, no. 4, pp. 1–1, Jun. 1999.

[32] D. Chang, L. Dooley, and J. E. Tuovinen, "Gestalt Theory in Visual Screen Design: A New Look at an Old Subject," in *Proceedings of the Seventh World Conference on Computers in Education Conference on Computers in Education: Australian Topics - Volume 8*, Darlinghurst, Australia, Australia, 2002, pp. 5–12.

[33] S. Phillai, "Hadoop Core Components | BigData Jury," *BigData Jury*, 01-Mar-2014. .

[34] "Hadoop Tutorial: Hello World - An Overview of Hadoop with HCatalog, Hive and Pig," *Hortonworks*. [Online]. Available: http://hortonworks.com/hadoop-tutorial/hello-world-an-introduction-to-hadoop-hcatalog-hive-and-pig/. [Accessed: 05-Jan-2015].

[35] R. Meadows, "Half of NZ businesses not ready for cyber attack," *Stuff.co.nz*, New Zealand, 03-Oct-2014.

[36] *Handbook of Human Centric Visualization*, Huang, Weidong (Ed). 2014.

[37] D. J. Pohly, S. McLaughlin, P. McDaniel, and K. Butler, "Hi-Fi: Collecting High-fidelity Whole-system Provenance," in *Proceedings of the 28th Annual Computer Security Applications Conference*, New York, NY, USA, 2012, pp. 259–268.

[38] "Hortonworks. We Do Hadoop.," *Hortonworks*. [Online]. Available: http://hortonworks.com/. [Accessed: 01-Feb-2015].

[39] A. Nambiar, "How to Install WordPress on Windows Localhost Server using WAMP," *ShoutAboutThis*, 08-Jun-2014. .

[40] O. Q. Zhang, R. K. L. Ko, M. Kirchberg, C. H. Suen, P. Jagadpramana, and B. S. Lee, "How to Track Your Data: Rule-Based Data Provenance Tracing Algorithms," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, Los Alamitos, CA, USA, 2012, vol. 0, pp. 1429–1437.

[41] O. Q. Zhang, M. Kirchberg, R. K. L. Ko, and B.-S. Lee, "How to Track Your Data: The Case for Cloud Computing Provenance," in *IEEE 3rd International Conference on Cloud Computing Technology and Science, CloudCom 2011, Athens, Greece, November 29 - December 1, 2011*, 2011, pp. 446–453.

[42] J. Zhang, K. A. Johnson, J. T. Malin, and J. W. Smith, "Human-Centered Information Visualization," presented at the Internation Workshop on Dynamic Visualization and Learning, Tubingen, Germany, 2002.

[43] M. D. Boyd, "InProv: Visualizing Provenance Graphs with Radial Layouts and Time-Based Hierarchical Grouping," Harvard College Cambridge,

Massachusetts, 2012.

[44] “Karma Provenance Framework.” [Online]. Available: http://www.extreme.indiana.edu/karma/. [Accessed: 04-Jan-2015].

[45] “Laws of Pragnanz | in Chapter 04: Senses | from Psychology: An Introduction by Russ Dewey.” [Online]. Available: http://www.intropsych.com/ch04_senses/laws_of_pragnanz.html. [Accessed: 08-Jan-2015].

[46] “Learn the Gestalt Laws of Perceptual Organization,” *About.com Education*. [Online]. Available: http://psychology.about.com/od/sensationandperception/ss/gestaltlaws_4.htm. [Accessed: 08-Jan-2015].

[47] J. Levin, “List of the Gestalt Principles,” 31-Mar-2006. [Online]. Available: http://pages.ucsd.edu/~jalevin//gp/gestalt-principles.html. [Accessed: 04-Jan-2015].

[48] T. Mala and T. V. Geetha, *Multi Level Categorization Visualization Based on Gestalt Perception Model*, vol. 31, 1 vols. 2009.

[49] “Multi Level Categorization Visualization Based on Gestalt Perception Model.” .

[50] “MySQL Views Tutorial,” *MySQL Tutorial*. . [Online]. Available: http://www.mysqltutorial.org/mysql-views-tutorial.aspx. [Accessed: 02-Feb-2015].

[51] J. Clark, “Neoformix - Blog,” 05-Nov-2014. [Online]. Available: http://www.neoformix.com/archive.html. [Accessed: 11-Jan-2015].

[52] T. Munzner, C. Johnson, R. Moorhead, H. Pfister, P. Rheingans, and T. S. Yoo, “NIH-NSF visualization research challenges report summary,” *IEEE Computer Graphics and Applications*, vol. 26, no. 2, pp. 20–24, Mar. 2006.

[53] A. Yehia, “Organizational Perception,” 01-Jan-2011. [Online]. Available: http://students.aub.edu.lb/~awy01/psychology/organizationalperception.html. [Accessed: 09-Jan-2015].

[54] G. Windholz, “Pavlov and the demise of the influence of Gestalt psychology in the Soviet Union,” *Psychol. Res*, vol. 46, no. 3, pp. 187–206, Aug. 1984.

[55] K. Koffka, “Perception: an introduction to the Gestalt-Theorie.,” *Psychological Bulletin*, vol. 19, no. 10, pp. 531–585, 1922.

[56] Boundless, “Perceptual Mapping,” *Boundless*, Nov. 2014.

[57] M. A. Nestrud and H. T. Lawless, “Perceptual Mapping of Apples and Cheeses Using Projective Mapping and Sorting,” *Journal of Sensory Studies*, vol. 25, no. 3, pp. 390–405, 2010.

[58] R. K. L. Ko and M. A. Will, “Progger: An Efficient, Tamper-Evident Kernel-Space Logger for Cloud Data Provenance Tracking,” in *2014 IEEE 7th International Conference on Cloud Computing (CLOUD)*, 2014, pp. 881–889.

[59] J. Cheney, S. Chong, N. Foster, M. Seltzer, and S. Vansummeren, “Provenance: A Future History,” in *Proceedings of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems*

Languages and Applications, New York, NY, USA, 2009, pp. 957–964.

[60] C. Hadjigeorgiou, "RDBMS vs NoSQL: Performance and Scaling Comparison," presented at the EPCC, The University of Edinburgh, 23-Aug-2013.

[61] J. Wood, "Real World CouchDB," 25-Jun-2011. [Online]. Available: http://www.slideshare.net/jwoodslideshare/real-world-couchdb. [Accessed: 01-Feb-2015].

[62] D. A. Keim, M. Ankerst, and H.-P. Kriegel, "Recursive Pattern: A Technique for Visualizing Very Large Amounts of Data," in *Proceedings of the 6th Conference on Visualization '95*, Washington, DC, USA, 1995, p. 279–.

[63] S. E. Palmer, "Hierarchical structure in perceptual representation," *Cognitive Psychology*, vol. 9, no. 4, pp. 441–474, Oct. 1977.

[64] B. Thuraisingham, T. Cadenhead, M. Kantarcioglu, and V. Khadilkar, *Secure Data Provenance and Inference Control with Semantic Web*. CRC Press, 2014.

[65] V. Tan, P. Groth, S. Miles, S. Jiang, S. Munroe, S. Tsasakou, and L. Moreau, "Security Issues in a SOA-Based Provenance System," in *Proceedings of the 2006 International Conference on Provenance and Annotation of Data*, Berlin, Heidelberg, 2006, pp. 203–211.

[66] P. Buneman, D. Suciu, and Eds, "Special Issue on Data Provenance," *Bulletin of the Technical Committee on Data Engineering, IEEE Computer Society*, vol. 32, no. 4, pp. 1–58.

[67] M. Bruls, K. Huizing, and J. J. van Wijk, "Squarified Treemaps," in *Data Visualization 2000*, D. ir W. C. de Leeuw and ir R. van Liere, Eds. Springer Vienna, 2000, pp. 33–42.

[68] C. Vermehren, "The Art of Web Analytics," 12-Oct-2009. . [Online]. Available: http://www.theartofwebanalytics.com/. [Accessed: 19-Jan-2015].

[69] B. B. Bederson and B. Shneiderman, *The Craft of Information Visualization: Readings and Reflections*. Elsevier, 2003.

[70] H. G. Nelson and E. Stolterman, *The Design Way: Intentional Change in an Unpredictable World : Foundations and Fundamentals of Design Competence*. Educational Technology, 2003.

[71] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations," in *, IEEE Symposium on Visual Languages, 1996. Proceedings*, 1996, pp. 336–343.

[72] M. Kevin, "The Gestalt Laws of Perception and how to use them in UI design | Architecting Usability." .

[73] J. J. van Wijk, "The value of visualization," in *IEEE Visualization, 2005. VIS 05*, 2005, pp. 79–86.

[74] D. Paradi, "Treemap Diagrams; November 12, 2013 | Think Outside The Slide," 12-Nov-2013. [Online]. Available: http://www.thinkoutsidetheslide.com/november-12-2013/. [Accessed: 24-Jan-2015].

[75] S. Bhulai, P. Kampstra, L. Kooiman, G. Koole, M. Deurloo, and B. Kok,

"Trend Visualization on Twitter: What's Hot and What's Not?," presented at the DATA ANALYTICS 2012, The First International Conference on Data Analytics, 2012, pp. 43–48.

[76] S. E. Palmer, *Vision science - Photons to phenomenology*. MIT Press, Cambridge, MA. 1999 (hardcover, 810 pp).

[77] "Visualization," *Oxford English Dictionary*. .

[78] P. Chen, B. Plale, Y. Cheah, D. Ghoshal, S. Jensen, and Y. Luo, "Visualization of network data provenance," in *2012 19th International Conference on High Performance Computing (HiPC)*, 2012, pp. 1–9.

[79] R. Bourdon, "WampServer," *WampServer*. [Online]. Available: http://www.wampserver.com/en/. [Accessed: 01-Feb-2015].

[80] "Welcome to Apache™ Hadoop®!," *Hadoop*, 12-Dec-2014. [Online]. Available: http://hadoop.apache.org/. [Accessed: 01-Feb-2015].

[81] O. Kulyk, R. Kosara, J. Urquiza-Fuentes, and I. Wassink. Human-Centered Visualization Environments, volume 4417 of Lecture Notes in Computer Science, chapter Human-Centered Aspects, pages 13–75. Springer, Human-centered aspects, 2007.

[82] "JSON." [Online]. Available: http://json.org/. [Accessed: 08-Feb-2015].

[83] "JSON in JavaScript." [Online]. Available: http://www.json.org/js.html. [Accessed: 08-Feb-2015].

[84] D. J. Repici, "CSV Comma Separated Value File Format - How To - Creativyst - Explored,Designed,Delivered.(sm)." [Online]. Available: http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm. [Accessed: 08-Feb-2015].

APPENDIX A

A.1 ETHICAL APPLICATION APPROVAL LETTER.

Faculty of Computing and Mathematical Sciences
Rorohiko me ngā Pūtaiao Pāngarau
The University of Waikato
Private Bag 3105
Hamilton
New Zealand

Phone +64 7 838 4322
www.fcms.waikato.ac.nz

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

27 November 2014

Jeffery Garae
C/- CROW
THE UNIVERSITY OF WAIKATO

Dear Jeffery

Application for approval under the Ethical Conduct in Human Research and Related Activities Regulations

I have considered your application for a research project involving human participants entitled "User-Centric Visualization of Data Provenance with Gestalt Principals" which is being conducted as part your Master's degree research.

The procedure described in your request is acceptable. I note that participants involved in the study will not be identified in publications and/or reports, and that upon completion of the research study, the raw data will be deleted from the data storage. The final anonymised data collected will be stored in the FCMS Data Archive for 5 years and at the end of the research, the anonymised data may be compiled together and released as a public dataset for future research uses in this area.

The Participant Information Sheet, Research Consent Form and Visualization Questionnaire comply with the requirements of the University's human research ethics policies and procedures.

I therefore approve your application to perform the research project.

Yours sincerely

Michael Mayo
Human Research Ethics Committee
Faculty of Computing and Mathematical Sciences

APPENDIX B

B.1 USER STUDY QUESTIONNAIRE 1 - PART 1

User-centric questions on provenance based on 5W1H

Note: *using the types of Visualization excel table I have chosen from Vis-type 3-to-102 as the solutions of the questions below.*

1. Where are my files stored?
   ANSWER: Vis – 3, 4, 5, 11, 36, 37, 47, 51, 66, 92, 93,
2. How many copies of my file stored in the cloud now?
   ANSWER: Vis – 3, 5, 8, 11, 23, 38, 51, 64
3. When are my files accessed in the cloud?
   ANSWER: Vis – 5, 8, 11, 23, 27, 28, 38, 51, 64
4. When my files are last accessed in the cloud?
   ANSWER: Vis – 5, 11, 27, 28, 48, 49, 57, 58, 59, 60, 64, 67,
5. How many times are my files accessed over the last month?
   ANSWER: Vis – 5, 8, 11, 27, 28, 48, 49, 57, 58, 59, 60, 64, 67,
6. Why is my file renamed?
   ANSWER: Vis – 3, 4, 5, 7, 8, 11, 22,
7. Why is my file moved from its original directory?
   ANSWER: Vis – 5, 6, 7, 11, 13, 22,
8. Who is accessing my files from the cloud?
   ANSWER: Vis – 5, 7, 8, 11, 62, 63, 77, 78
9. Who can access my files in the cloud?
   ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 77, 78,
10. Who can access my files from the Physical machine?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 77, 78,
11. What can store my files?
    ANSWER: Vis – 5, 11, 51, 94, 95,
12. What are the process in accessing my files from the cloud?
    ANSWER: Vis – 5, 11, 53, 54, 94, 95,
13. What happened to my files when stored in the cloud?
    ANSWER: Vis – 5, 11, 94, 95,

B.2 USER STUDY QUESTIONNAIRE 1 - PART 2

15. How can I track my files in the cloud?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 78,

16. When can my files be accessed by others?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 47, 51, 67, 68, 69, 70, 71, 72, 73, 74, 78,

17. What is the cause of my file being changed, renamed or deleted?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 78,

18. When do I check on the status of my files in the cloud or physical machine?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 78,

19. What will happen if I allow others to access my files?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 78,

20. Who can monitor my files in the cloud and physical machine?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 67, 68, 69, 70, 71, 72, 73, 74, 77, 78,

21. How can I see my file movements (activities) in the cloud?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 78, 85,

22. What do I need to track when monitoring my files in the cloud?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 62, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 78, 84, 85

23. How can I know if my files are changed, copied, renamed?
    ANSWER: Vis – 3, 5, 11, 38, 39, 40, 62, 67, 68, 69, 70, 71, 72, 73, 74, 78, 84, 85,

NOTE:

In addition to that, the users will be given an option to use other visualization that are not mentioned, for further understanding if needed. Alongside the use of Gestalt in the visualization will allow the users of the visualization tool to allow them to perceive final thoughts and outcomes of their file movements in the cloud / physical machines.

B.3 User Study Questionnaire 2

If you a user of the cloud or a local machine, what are the 3 most important questions you would consider with regards to the value of your data / information?

1. Where are my files?
2. How many copies of my file stored in the cloud now?
3. When are my files accessed in the cloud?
4. When my files are last accessed in the cloud?
5. How many times are my files accessed over the last month?
6. Why is my file renamed?
7. Why is my file moved from its original directory?
8. Who is accessing my files from the cloud? (users, processes etc.) links to
9. Who can access my files in the cloud?
10. Who can access my files from the Physical machine?
11. What can store my files?
12. What are the process in accessing my files from the cloud?
13. What happened to my files when stored in the cloud?
14. Why is my file accessed?
15. How can I track my files in the cloud?
16. When can my files be accessed by others?
17. What is the cause of my file being changed, renamed or deleted?
18. When do I check on the status of my files in the cloud or physical machine?
19. What will happen if I allow others to access my files?
20. Who can monitor my files in the cloud and physical machine?
21. How can I see my file movements (activities) in the cloud?
22. What do I need to track when monitoring my files in the cloud?
23. How can I know if my files are changed, copied, renamed?

B.4 User Study findings

Type of Visualization	Goal of Visualization	D3 Visualization representation	Gestalt Approach					
			similarity	uniformility	Closure	continuation	Proximity	figure
Open file actions base on different PID	statistics on File access or modification	Treemap / Buble chart	√					
File created (open), read, write and close	User Report	Force directed Graph/ treemap /heatmap		√		√	√	
copying of a file and saving it to new directory	keeping track of the file movement/activities	Collapsible Tree layoutTree / Buble chart	√	√				
moving a file from one directory to another (SCP)	keeping track of the file movement/activities/in case of leakage	Froce directed Graph	√					
renaming a file	statistics, tracking of data/file	Treemap / Buble chart	√					
rename directory	statistics, tracking of data/file	treemap / buble chart	√					
Open file -> Read -> Write ->save in new directory	File Process and tracking	Treemap / Buble chart / heatmap / heatmap	√	√	√			√
delete file action	statistics to show which possible types of files deleted	buble chart	√					
tracking multiple CRWC actions	percieve visualization for new paterns of processes	Collapsible Force layout /heatmap	√		√		√	√
identifying Socket processes	Statistics of whats happening behind the seen (kernal	Treemap / Buble chart /heatmap	√			√		
moving file from VM to PM (file = any type of file)	tracking of Data movement for leakage	Treemap / Buble chart	√				√	
Single create, read, close file in VM then move to PM	File Tracking	Collapsible Tree layoutTree / Buble chart		√	√			√
multiple Data CRWC file then	tracking for leakage	Streamgraph / heatmap		√	√			√
File created (open), read, write and close in VM	statistics purposes	buble chart / heat map	√		√	√	√	√
File created (open), read, write and close in PM	statistics purposes	buble chart / heat map	√		√	√	√	√
Create, read, write file	Statistics purposes	zoomable Treemap	√				√	
sendMSG action in the kernal	statistics	Treemap / Buble chart	√	√				
recvMSG ation in the kernal	statistics purposes and also identify who the msg is sent	Donut chats with labels	√	√				
Change of file permission	statistics and possibly know why permision has been	bubble charts / donut chat/	√					
remove file from location	statistics	Treemap / Buble chart	√					
link file to an existing file	tracking of file, maybe for leakage or malicious activities	Bounded force layout	√			√		
create socket	statistics and percieve why socket is created	treemap	√					
Establish connection to remote host (e.g. from VM to PM)	tracking of connections	treemaps	√					
Establish connection to remote host (e.g. from VM to another VM)	tracking of connections	treemaps	√					
File create, write and save to two	Tracking of files for leakage	treemaps / heat maps	√		√		√	
File open, modify and save	tracking purposes	treemaps / buble chart / heat map	√			√	√	
Create Directory action	statistics and tracking	Donut chats with labels	√	√				
Directory move action from one drive / location to another locally	Tracking	treemaps / 3D Donut chart	√					
Directory move action from one drive / location to another remotely	Tracking	treemaps	√	√				
directory delete action in VM	statistics and tracking	treemaps	√					
Directory rename action in VM	statistics and tracking	treemaps	√					
Directory delete action in PM	statistics and tracking	treemaps	√					
Directory rename action in PM	statistics and tracking	treemaps	√	√				
File save to external drive in VM	tracking	treemaps	√	√				
File save to external drive in PM	Tracking	buble chart	√					
Process execution jorney (track multiple process phases)	tracking and understanding kernal processes	treemaps / heatmap	√	√		√	√	
directory change permission in VM	tracking	buble chart	√	√				
Directory change permission in PM	tracking	buble chart	√	√				
unintensional file creation in VM	tracking and understanding kernal processes	buble chart	√					
unintensional file creation in PM	tracking and understanding kernal processes	buble chart	√					

temp-file creation	tracking and understanding kernal processes	buble chart	√					
temp-file deletion	tracking and understanding kernal processes	buble chart	√					
socket link creation	understanding kernal	buble chart	√					
path creation	understanding kernal	treemaps	√					
file destination process	tracking	Treemaps /heatmap	√	√		√		
file movement from host to	tracking	treemaps / heatmap	√			√	√	
file attachment to email from host	tracking	treemaps / heatmap	√			√	√	
attaechment sent via email	tracking	treemaps	√					
file write to Database	tracking	treemaps	√					
file print action	statistics	buble chart	√					
open application action in VM	statistics	buble chart	√					
open application action in PM	statistics	buble chart	√					
create an empty file in VM (echo	file tracking	Collapsible Force layout	√					
create and empty file in PM (echo	File tracking	Collapsible Force layout	√					
compress file in VM	File tracking	buble chart	√					
compress file in PM	File tracking	buble chart	√					
Compress directory	directory Tracking	buble chart	√					
decompress directory in VM or PM	directory tracking	Buble chart / Sunburst	√					
decompress file in VM or PM	File tracking	buble chart	√					
read file either in VM or PM	statistics and tracking	Donut chats with labels	√					
read directory in VM or PM	statistics and tracking	Donut chats with labels	√					
Change file privilege ("chmod")	statistics and tracking	Clustered Force Layout III	√					
Change directory privilege	statistics and tracking	Clustered Force Layout III	√					
Change group of a file	statistics and tracking	circle Packing / buble	√					
Change owner of file (using "Sudo" chown)	statistics and tracking purposes	Calender View / Collision detection	√					
Change owner of directory (using sudo chown)	statistics and tracking purposes	Calender View / Collision detection	√					
Change owner and group of a file (using sudo)	statistics and tracking purposes	Calender View / Collision detection	√					
change owner and group of a directory (using sudo)	statistics and tracking purposes	calender View / Collision detection/ circle packing/ buble chart	√					
change owner and group of a directory recusively (using sudo)	statistics and tracking purposes	collapsible Force layout /heatmap	√	√		√	√	
change privilage of a directory using sudo	statistics and tracking purposes	Collapsible Force layout /heatmap	√			√	√	
Change privilage of a file recursively (using sudo)	statistics and tracking purposes	Collapsible Force layout /heatmap	√			√	√	
Change executive of a file (chmod	statistics and tracking	Force-Directed Graph	√					
Change executive of a file (using	statistics and tracking	Force-Directed Graph	√					
change executive of a directory recursively (using sudo)	statistics and tracking purposes	buble charts / heatmap	√		√	√	√	
Add user action	statistics and tracking	buble / donut charts /	√		√		√	
Add group action	statistics and tracking	buble / donut charts	√					
Change user password	statistics and tracking	buble / donut charts with	√					
Change user's group	statistics and tracking	Buble / donut chart with	√					
Change date	statistics and system logging	Buble /donut chart with	√					
mount volume	statistics and system logging	Buble / Donut chart with	√					
auto mount via fstab	tracking and understanding kernal processes	Buble / Donut chart with labels	√			√		
create a schedule Task	tracking and understanding kernal processes	buble charts / 3D Donut chart						
delet a schedule Task	tracking and understanding kernal processes	Buble chart / Sunburst						
scheduled task	statistics and system logging	Force-Directed Graph	√					
partition volume	statistics and system logging	Force-Directed Graph	√					
Change NIC settings (ifconfig)	statistics and system logging	buble charts	√					
download / upload (sftp/scp)	tracking purposes	Clustered Force Layout III	√					
connect locally via (ssh - 127.0.0.1)	tracking purposes	Clustered Force Layout III	√					
connect remotely (ssh)	tracking purposes	Clustered Force Layout III	√					
transfer via tcp /udp port	tracking purposes	treemaps	√	√		√	√	
list command ("ls")	statistics and history of	Force-Directed Graph	√					
download via "get" command	tracking purposes	buble charts	√					
upload via "put" command	statistics and tracking	buble charts	√			√		
change Directory command	statistics and tracking	treemaps / sunburst	√					
change root password	tracking purposes	sunburst	√					

Port scan (host)	tracking and understanding kernal processes	Force directed Graph/ treemap / heatmap	√	√	√	√	√	
Possible Malicious detection	tracking purposes	Force directed Graph/ treemap /heatmap	√	√	√	√	√	
password bruteforce (host)	tracking purposes and malware detection	Force directed Graph/ treemap	√	√	√	√		
user account rename	statistics and tracking	buble charts / 3D Donut	√					
detect DOS attack	Tracking and malisious attacks	heatmap / calender chart /network map / heatmap	√		√	√		

B.5 UVISP – USER VISUALIZATION QUESTIONNAIRE AND FEEDBACK

UVisP – User Visualization Questionnaire & feedbacks

Visualization Type	Is the Visualization Informative? (Scale: 1 – 5) 5 being the most informative	Level of Understanding (Scale: 1 – 5) 1 being the easiest & 5 is the hardest	Gestalt law Application (P, S, Pr, CF)	Time taken to visualize (sec)
Vis_Type_1	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 □		
Vis_Type_2	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 □		
Vis_Type_3	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 □		
Vis_Type_4	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 □		
Vis_Type_5	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 □		
Vis_Type_6	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 □		

Feedbacks:

B.6 UVisP – User Visualization Questionnaire and Feedback Sample

B.6.1 User ID: 1 Test Results

ID: 1 (23:05pm) (05/Feb/2014)

UVisP – User Visualization Questionnaire & feedbacks

Visualization Type	Is the Visualization Informative? (Scale: 1 – 5) 5 being the most informative	Level of Understanding (Scale: 1 – 5) 1 being the easiest & 5 is the hardest	Gestalt law Application (P, S, Pr, CF)	Time taken to visualize (sec)
Vis_Type_1	1 □ 2 ☒ 3 ☒ 4 □ 5 □	1 □ 2 □ 3 ☒ 4 □ 5 □	S, Pr	< 1
Vis_Type_2	1 ☒ 2 □ 3 □ 4 □ 5 □	1 ☒ 2 □ 3 □ 4 □ 5 □	P	< 1
Vis_Type_3	1 □ 2 ☒ 3 □ 4 □ 5 □	1 ☒ 2 □ 3 □ 4 □ 5 □	Pr	< 1
Vis_Type_4	1 ☒ 2 □ 3 □ 4 □ 5 ☒	1 ☒ 2 □ 3 □ 4 □ 5 ☒	CF	3
Vis_Type_5	1 □ 2 ☒ 3 □ 4 □ 5 □	1 □ 2 ☒ 3 □ 4 □ 5 □	CF	2
Feedbacks: Vis-Type_6	1□ 2□ 3☒4□5□	1☒2□3□4□5□	S	4

for Vis-type-4 is hard to understand
for Vis-type-3 very easy to understand with a lot of information and clear.

B.6.2 USER ID: 2 TEST RESULTS

ID 2 · 08 · 20 (09/FEB/2019)

UVisP – User Visualization Questionnaire & feedbacks				
Visualization Type	Is the Visualization Informative? (Scale: 1 – 5) 5 being the most informative	Level of Understanding (Scale: 1 – 5) 1 being the easiest & 5 is the hardest	Gestalt law Application (P, S, Pr, CF)	Time taken to visualize (sec)
Vis_Type_1	1 □ 2 □ 3 ☒ 4 □ 5 □	1 □ 2 □ 3 ☒ 4 □ 5 □	CF	120
Vis_Type_2	1 □ 2 □ 3 □ 4 ☒ 5 □	1 □ 2 □ 3 □ 4 ☒ 5 □	P	60
Vis_Type_3	1 □ 2 □ 3 □ 4 □ 5 ☒	1 ☒ 2 □ 3 □ 4 □ 5 ☒	P	60
Vis_Type_4	1 □ 2 □ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 □ 5 ☒	Pr	120
Vis_Type_5	1 □ 2 ☒ 3 □ 4 □ 5 □	1 □ 2 □ 3 □ 4 ☒ 5 □	S	60
Vis_Type_6	1 ☒ 2 □ 3 □ 4 □ 5 □	1 ☒ 2 □ 3 □ 4 □ 5 □	S	240

Feedbacks:

- Vis_Type_1 is very hard, too much information
- Vis_Type_2 very complicated with less information
- Vis_Type_3 is very good, easy need information
- Vis_Type_3 easy and can follow.

APPENDIX C

Source codes

C.1 INDEX.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">
<HTML>

<HEAD>
<TITLE>CROW ~~ Provenance Visualization</TITLE>
</HEAD>
<?php
    include "connection1.php";
?>

<FRAMESET cols="12%, 75%">
  <FRAMESET rows="10%, 50%">
        <FRAME
src="http://localhost/CirclebarVis/image/crow_gold_black_bg.gi
f">
      <FRAME
src="http://localhost/CirclebarVis/indexCBVv1.html">
  </FRAMESET>
  <frameset rows="40%, 50%">
  <FRAME src="http://localhost/CirclebarVis/indexCBV.html">
  <FRAME src="http://localhost/CirclebarVis/indexCBVv4.html">
  </frameset>
  <NOFRAMES>
      <P>This frameset document contains:
      <UL>
         <LI><A  href="contents_of_frame1.html">Cyber  Security
Researchers of Waikato</A>
         <LI><IMG src="Content of Frame2" alt="A neat image">
         <LI><A  href="Content  of  Frame3">Some  other  neat
contents</A>
         <LI><A  href="Content  of  Frame4">Some  other  neat
contents</A>
      </UL>
      </P>
  </NOFRAMES>
</FRAMESET>
</HTML>
```

C.2 VISUALIZATION SOURCE CODES

```
    var vis = d3.select("#viz").append("svg:svg")
.attr("width", 2000)
.attr("height", 1500)
.append("svg:g")
.attr("transform", "translate(40, 0)");

    var tree = d3.layout.tree()
.size([1500,700]);

    var diagonal = d3.svg.diagonal()
.projection(function(d) { return [d.y, d.x]; });

    var nodes = tree.nodes(treeData);
    var link = vis.selectAll("pathlink")
.data(tree.links(nodes))
.enter().append("svg:path")
.attr("class", "link")
.attr("d", diagonal);

    var node = vis.selectAll("g.node")
.data(nodes)
.enter().append("svg:g")
.attr("transform", function(d) { return "translate(" + d.y + "," + d.x + ")"; })

    node.append("svg:circle")
.attr("r", 4.5);

    node.append("svg:text")
```

C.3 INDEXCBV.HTML

```
  <meta charset="UTF-8">

  <title>Circle Bar Chart - CodePen</title>

    <link rel="stylesheet" href="css/style.css" media="screen"
type="text/css" />

      <div class="headerContainer">
        <h1><a href="">User-Centric Provenance Visualization with
Gestalt Principles</a></h1>
        <h2><a href="">File Action Statistics:</a></h2>

 <input type="file" id="file_input">

<script type="text/javascript">
  // Re-upload the file on each change
  document.getElementById('file_input').addEventListener('change',
fileHandler, false);
  var file = "null";

  function fileHandler(event){
    var files = event.target.files;
    var file = files[0];
    var reader = new FileReader();
    // This is what happens after the file has finished loading.
    reader.onload = function() {
      processFile(this.result); // output of result is the file
object
    }
    reader.readAsText(file);
  }

  // input file data into the objects
  function processFile(file){
    var lines = file.toString().split(/\r\n|\n\r|\n|\r/);
    // setTimeout 0 is used to create a psuedo asynchronous function,
otherwise it freezes the browser
    setTimeout(function(){
      // Start from 1 to skip the module inserted, and therefore - 1
because we are now offset by one.
      for(var i = 1; i < lines.length - 1; i++){
        addEntry(lines[i]); // Calls a function that adds to the
Taffy database
      }
    },0);
  }
  script
```

B.4 INDEXCBVV4.HTML

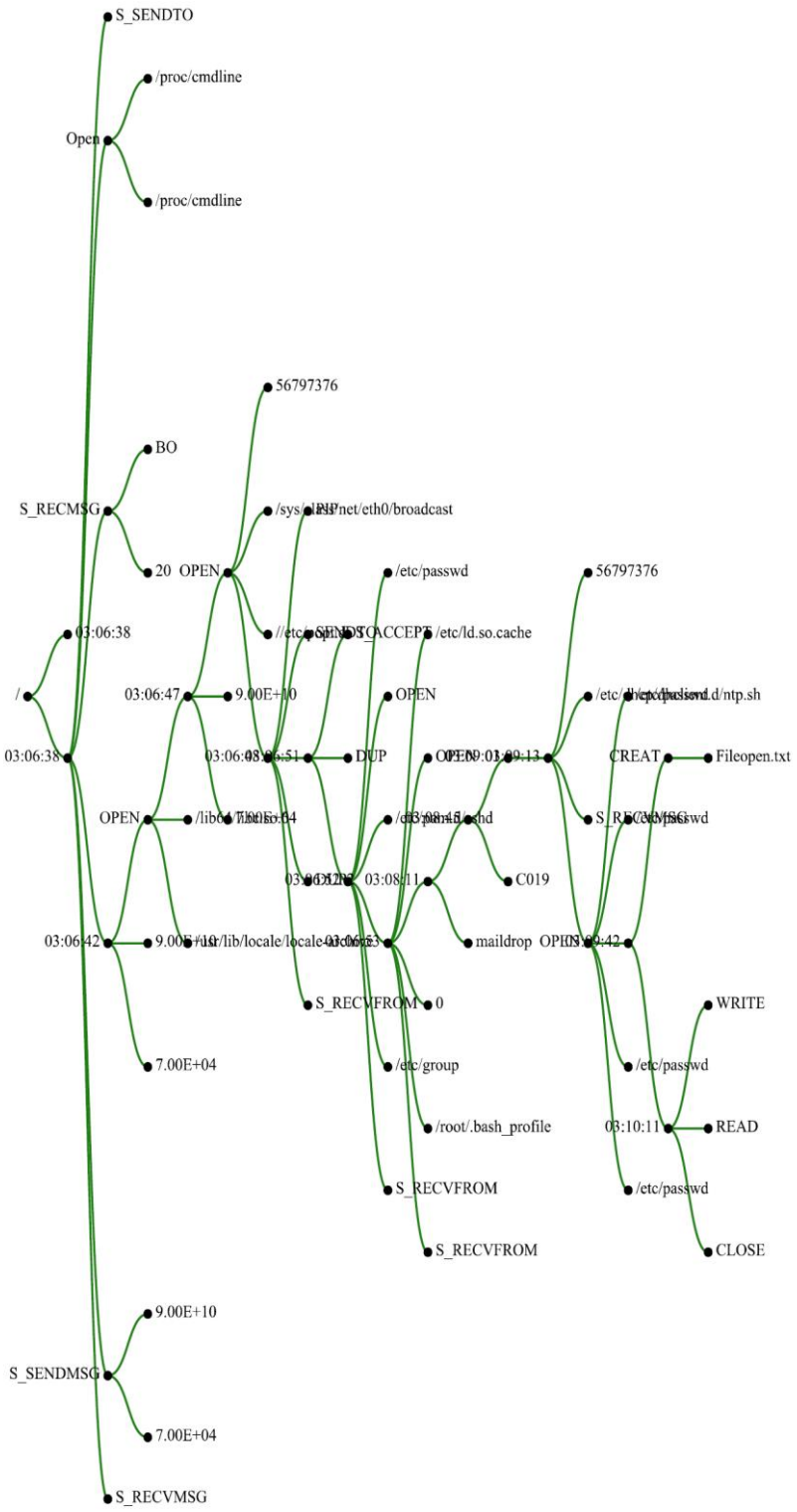
```
// ************** Generate the tree diagram   *********************
var margin = {top: 20, right: 120, bottom: 20, left: 120},
  width = 1200 - margin.right - margin.left,
  height = 900 - margin.top - margin.bottom;

var i = 0;
  duration = 750;

var tree = d3.layout.tree()
  .size([height, width]);
  //.size([size.height, size.width -
maxLabelLength*options.fontSize]);

var diagonal = d3.svg.diagonal()
  .projection(function(d) { return [d.y, d.x]; });

var svg = d3.select("body").append("svg")
  .attr("width", width + margin.right + margin.left)
  .attr("height", height + margin.top + margin.bottom)
  .append("g")
  .attr("transform", "translate(" + margin.left + "," +
                                    margin.top + ")");

// load the external data
d3.csv("http://localhost/CirclebarVis/ps.csv", function(error, data)
{

  // *********** Convert flat data into a nice tree **************
  // create a name: node map
  var dataMap = data.reduce(function(map, node) {
    map[node.name] = node;
    return map;
  }, {});

  // create the tree array
  var treeData = [];
  data.forEach(function(node) {
    // add to parent
    var parent = dataMap[node.parent];
    if (parent) {
      // create child array if it doesn't exist
      (parent.children || (parent.children = []))
        // add node to child array
        .push(node);
    } else {
      // parent is null or missing
      treeData.push(node);
    }
  });

  root = treeData[0];
```

C.5 HADOOP-HIVE-THRIFT-CLIENT CONNECTION (CONNECTION.PHP)

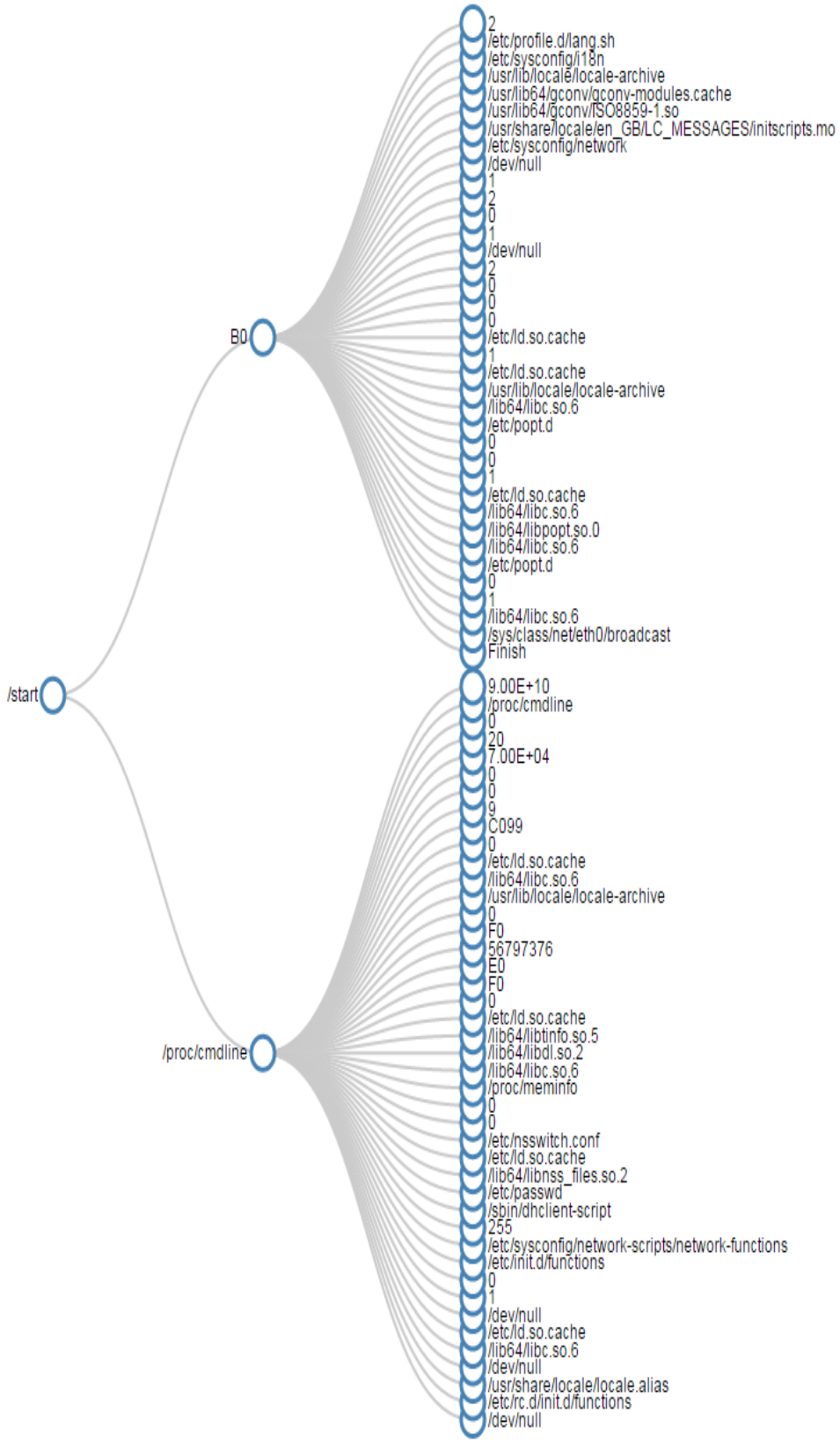
```php
<?php
// set THRIFT_ROOT to php directory of the hive distribution
$GLOBALS['THRIFT_ROOT'] = 'C:\Hadoop\php-thrift-hive-client-master\lib';
// load the required files for connecting to Hive
require_once $GLOBALS['THRIFT_ROOT'] . 'ThriftHive.php';
require_once $GLOBALS['THRIFT_ROOT'] . 'transport/TSocket.php';
require_once $GLOBALS['THRIFT_ROOT'] . 'protocol/TBinaryProtocol.php';

// Set up the transport/protocol/client
$transport = new TSocket('127.0.0.1', 10000);
$protocol = new TBinaryProtocol($transport);
$client = new ThriftHiveClient($protocol);
$transport->open();

// run queries, metadata calls etc
$client->execute('SELECT * from progger_scp');
$client->execute('SHOW TABLES');
$tables = $client->fetchAll();
var_dump($client->fetchAll());
$transport->close();

//echo '**********************************************************************';

?>
```

APPENDIX D

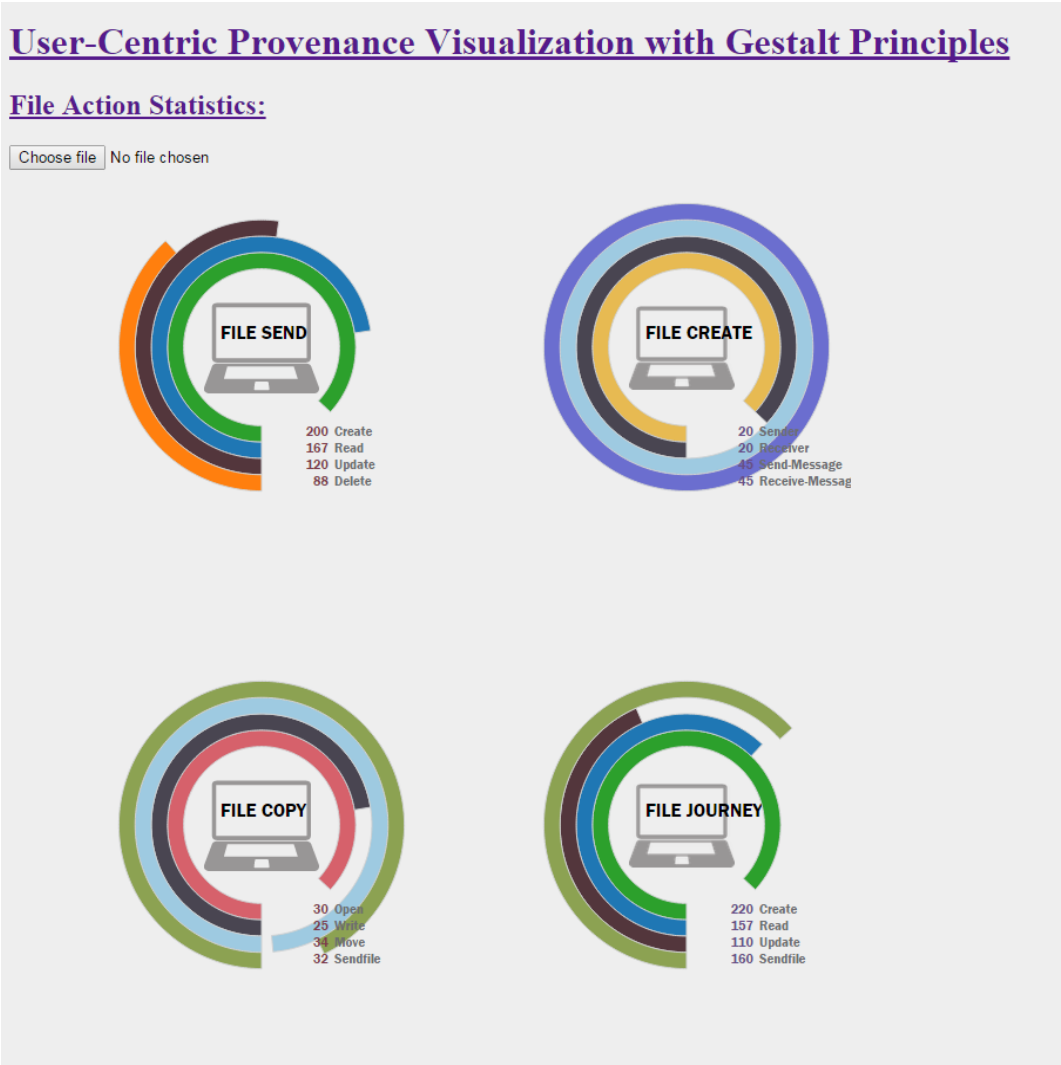
D.1 VIS_TYPE_1



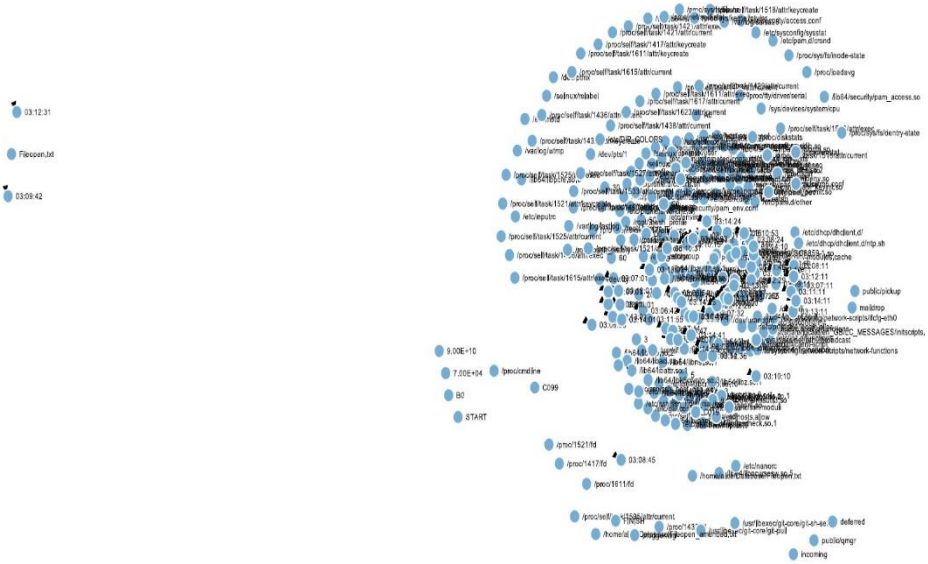
D.2 VIS_TYPE_2


/start
B0
2
/etc/profile.d/lang.sh
/etc/sysconfig/i18n
/usr/lib/locale/locale-archive
/usr/lib64/gconv/gconv-modules.cache
/usr/lib64/gconv/ISO8859-1.so
/usr/share/locale/en_GB/LC_MESSAGES/initscripts.mo
/etc/sysconfig/network
/dev/null
1
2
0
1
/dev/null
2
0
0
0
/etc/ld.so.cache
1
/etc/ld.so.cache
/usr/lib/locale/locale-archive
/lib64/libc.so.6
/etc/popt.d
0
0
1
/etc/ld.so.cache
/lib64/libc.so.6
/lib64/libpopt.so.0
/lib64/libc.so.6
/etc/popt.d
0
1
/lib64/libc.so.6
/sys/class/net/eth0/broadcast
Finish
/proc/cmdline
9.00E+10
/proc/cmdline
0
20
7.00E+04
0
0
9
C099
0
/etc/ld.so.cache
/lib64/libc.so.6
/usr/lib/locale/locale-archive
0
F0
56797376
E0
F0
0
/etc/ld.so.cache
/lib64/libtinfo.so.5
/lib64/libdl.so.2
/lib64/libc.so.6
/proc/meminfo
0
0
/etc/nsswitch.conf
/etc/ld.so.cache
/lib64/libnss_files.so.2
/etc/passwd
/sbin/dhclient-script
255
/etc/sysconfig/network-scripts/network-functions
/etc/init.d/functions
0
1
/dev/null
/etc/ld.so.cache
/lib64/libc.so.6
/dev/null
/usr/share/locale/locale.alias
/etc/rc.d/init.d/functions
/dev/null

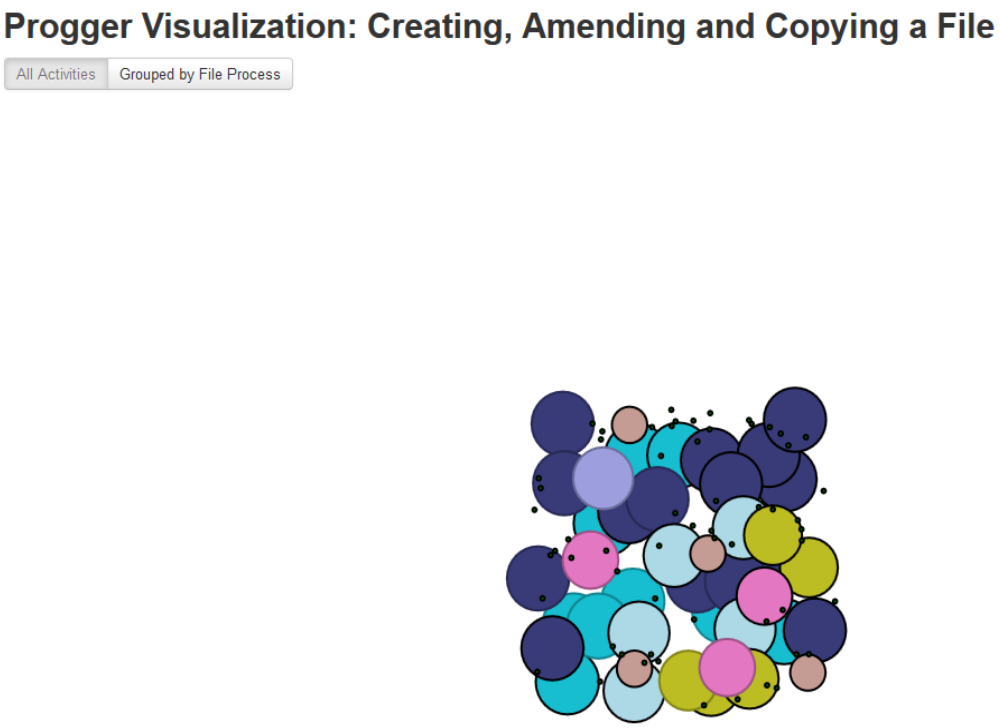
D.3 VIS_TYPE_3


User-Centric Provenance Visualization with Gestalt Principles
File Action Statistics:
Choose file
No file chosen
FILE SEND
200 Create
167 Read
120 Update
88 Delete
FILE CREATE
20 Sender
20 Receiver
45 Send-Message
45 Receive-Messag
FILE COPY
30 Open
25 Write
34 Move
32 Sendfile
FILE JOURNEY
220 Create
157 Read
110 Update
160 Sendfile

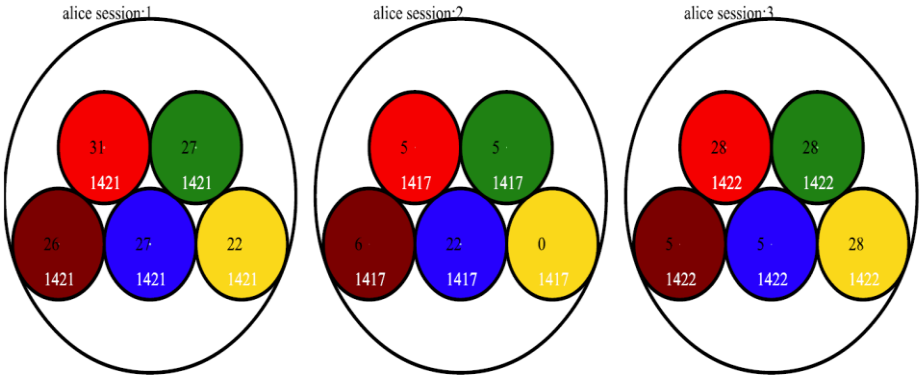
D.4 VIS_TYPE_4



D.5 VIS_TYPE_5


Progger Visualization: Creating, Amending and Copying a File
All Activities
Grouped by File Process

D.6 VIS_TYPE_6


Provenance Visualization
Visualizing System Calls against Process IDs!
Visualization showing different activities happening when a SCP command is executed in CentOS!
alice session:1
31
1421
27
1421
26
1421
27
1421
22
1421
alice session:2
5
1417
5
1417
6
1417
22
1417
0
1417
alice session:3
28
1422
28
1422
5
1422
5
1422
28
1422

APPENDIX E

E.1 PROGGER LOG FORMAT

```
//Log Formats

#define LOG_OPEN(type,user,pid,ppid,audit,paudit,pname,filename,path,flags,mode,fd) printk(KERN_INFO "%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s,%u,%u,%lu\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,pname,filename,path,flags,mode,fd)

#define LOG_CLOSE(type,user,pid,ppid,audit,paudit,fd) printk(KERN_INFO "%s%d,%s,%lu,%lu,%lu,%lu,%u\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,fd)

#define LOG_S_CLOSE(type,user,pid,ppid,audit,paudit,fd) printk(KERN_INFO "%s%d,%s,%lu,%lu,%lu,%lu,%us\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,fd)

#define LOG_RENAME(type,user,pid,ppid,audit,paudit,pname,oldfile,newfile,path) printk(KERN_INFO "%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s,%s\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,pname,oldfile,newfile,path)

#define LOG_UNLINK(type,user,pid,ppid,audit,paudit,pname,filename,path) printk(KERN_INFO "%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,pname,filename,path)

#define LOG_UNLINKAT(type,user,pid,ppid,audit,paudit,pname,filename,path,dirfd,flags) printk(KERN_INFO "%s%d,%s,%lu,%lu,%lu,%lu,%s,%s,%s,%u,%u\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,pname,filename,path,dirfd,flags)

#define LOG_DUP2(type,user,pid,ppid,audit,paudit,oldfd,newfd)printk(KERN_INFO"%s%d,%s,%lu,%lu,%lu,%lu,%u,%u\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,oldfd,newfd)

#define LOG_RDWR(type,user,pid,ppid,audit,paudit,fd,pos,data)printk(KERN_INFO"%s%d,%s,%lu,%lu,%lu,%lu,%u,%lu,%s\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,fd,pos,data)

#define LOG_S_RDWR(type,user,pid,ppid,audit,paudit,fd,pos,data) printk(KERN_INFO"%s%d,%s,%lu,%lu,%lu,%lu,%us,%lu,%s\n",PROGGER_ID,type,user,pid,ppid,audit,paudit,fd,pos,data)

#define LOG_MKDIR(type,user,pid,audit,name,path,mode)
```

E.2 PROGGER SYSTEM CALLS LISTS

```
#define SYSCALL_OPEN            0
#define SYSCALL_UNLINK          1
#define SYSCALL_WRITE           2
#define SYSCALL_CREAT           3
#define SYSCALL_MOVE            4
#define SYSCALL_CLOSE           5
#define SYSCALL_READ            6
#define SYSCALL_S_CONNECT       7
#define SYSCALL_S_SENDTO        8
#define SYSCALL_UNLINKAT        9
#define SYSCALL_MKDIR           10
#define SYSCALL_RMDIR           11
#define SYSCALL_SYMLINK         12
#define SYSCALL_LINK            13
#define SYSCALL_LINKAT          14
#define SYSCALL_CHOWN           15
#define SYSCALL_FCHOWN          16
#define SYSCALL_LCHOWN          17
#define SYSCALL_FCHOWNAT        18
#define SYSCALL_CHMOD           19
#define SYSCALL_FCHMOD          20
#define SYSCALL_FCHMODAT        21
#define SYSCALL_S_SENDMSG       22
#define SYSCALL_S_ACCEPT        23
#define SYSCALL_S_SOCKET        24
#define SYSCALL_SENDFILE        25
#define SYSCALL_S_RECVFROM      26
#define SYSCALL_S_RECVMSG       27
#define SYSCALL_DUP2            28
#define SYSCALL_PIPE            29
#define SYSCALL_PIPE2           30
#define SYSCALL_DUP             31
```

E.3 A PROGGER RAW DATA SAMPLE FROM A 'SCP' SCENARIO.

```
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:5,alice,1516,1422,1422,1421,3
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:0,alice,1516,1422,1422,1421,ls,/usr/lib/locale/lo
cale-archive,/home/alice/Database/,0,238612496,3
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:5,alice,1516,1422,1422,1421,3
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:0,alice,1516,1422,1422,1421,ls,.,/home/alice/Data
base/,591872,1,3
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:5,alice,1516,1422,1422,1421,3
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:2,alice,1516,1422,1422,1421,1,0,70726F676765722E6
C6F670A
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:5,alice,1516,1422,1422,1421,1
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:5,alice,1516,1422,1422,1421,2
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,1,0,1B5D303B616C69636
540686F73742D3139322D3136382D39382D353A7E2F44617461626173
6507
Jul 16 03:33:26 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,5B616C69636540686
F73742D3139322D3136382D39382D3520446174616261736 55D2420
Jul 16 03:33:28 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,73
Jul 16 03:33:28 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,73
Jul 16 03:33:28 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,63
Jul 16 03:33:28 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,63
Jul 16 03:33:28 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,70
Jul 16 03:33:28 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,70
Jul 16 03:33:29 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,20
Jul 16 03:33:29 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,20
Jul 16 03:33:31 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,70
Jul 16 03:33:31 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,70
Jul 16 03:33:33 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,72
Jul 16 03:33:33 host-192-168-98-5 kernel:
Progger:2,alice,1422,1421,1422,1421,2,0,72
Jul 16 03:33:33 host-192-168-98-5 kernel:
Progger:6,alice,1422,1421,1422,1421,0,0,6F
```

APPENDIX F

F.1 MYSQL DATABASE IMPLEMENTATION – STEPS

In order to start using this service and start storing data, the user has to create a new database, tables and load the data into the Database. Below we outline the steps of configuring phpMyAdmin.

Step 1: Login to the phpMyAdmin page by clicking on the phpMyAdmin icon in the "services" / quick Tap icon as shown below:

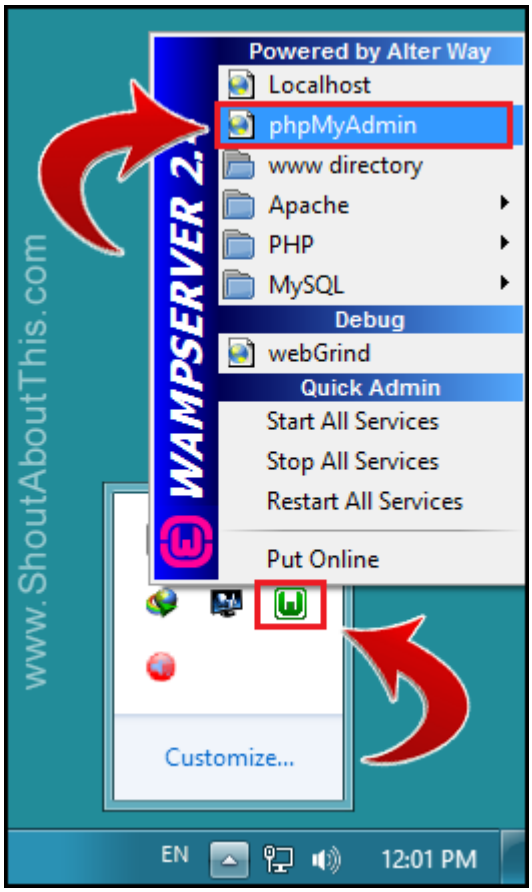


Figure 3.1. An image showing the status of WampServer and phpMyAdmin [39].

Figure 3.1 also indicates that the WampServer service is running. Other features such as PHP and Apache can be accessed using this quick icon (WampServer). All services can 'start', 'stop' and 'MySQL'.

Step 2: Navigate and click on "phpMyAdmin" as shown in Figure 3.1 or 'type' http://localhost/phpmyadmin/. This should load a new web page as shown in Figure 3.2. Figure 3.2 shows that a database has been created by the name of "proggerdb". Inside the proggerdb, a table has been created and several 'views' have also been created. However, a newly setup WampServer, "Step 3 (Figure 3.3)" of this section will illustrate to us how to create a Database, tables and views.

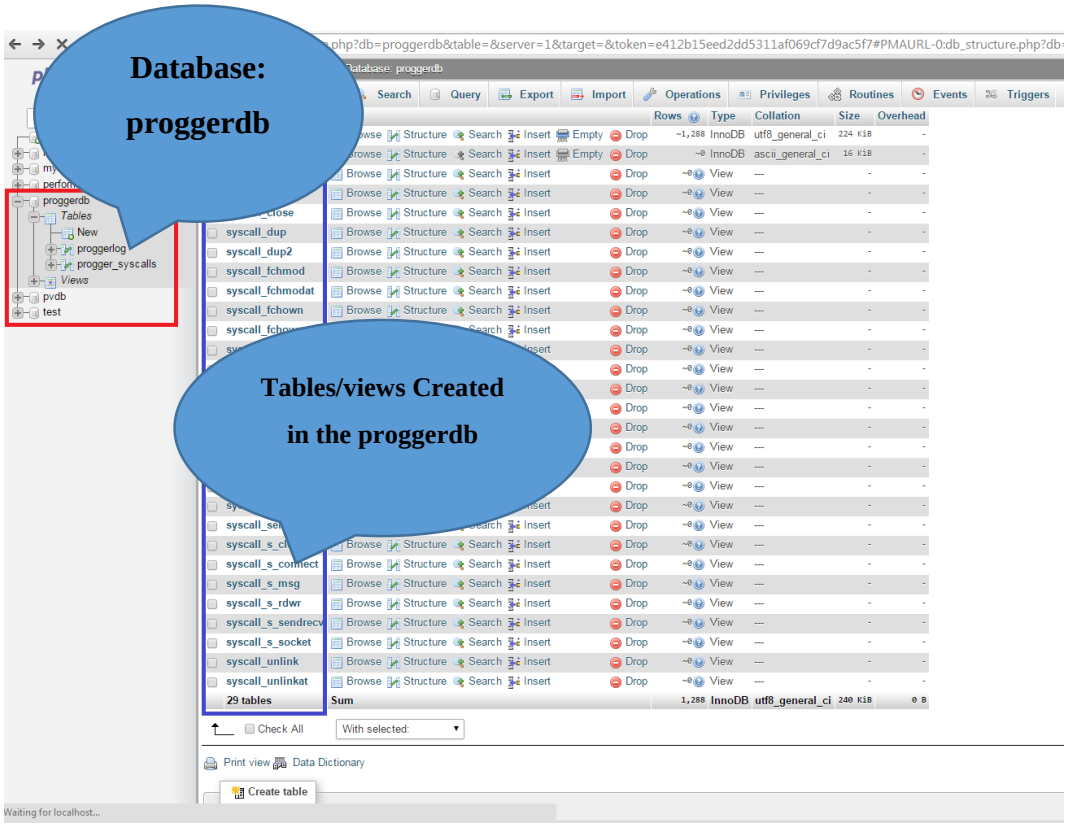


Figure 3.2. An example of a phpMyAdmin snapshot of a database.

Step 3: Create the required database by giving it a name.

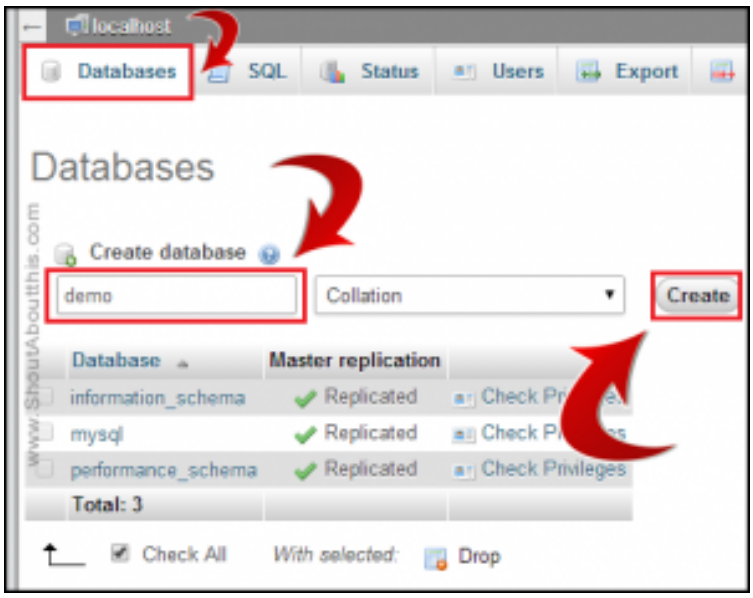


Figure 3.3. A snapshot of how to create databases in phpMyAdmin [39].

Step 4: Create "Tables" for this database.

Step 5: Upload data into the tables.

Since the table contains a lot of data and attributes, the use of "view" comes in handy to speed up the loading process when querying the database for data to process and visualize. A database view is a virtual table that allows the user to query the data inside it [50].

Step 6: Create "Views" based on the existing tables and data type.

Step 7: Establish connection to the proggerdb database with the front-end UVisP technique, by using a php connect script. (getdata.php). Figure 3.4 shows the getadata script sample.

Figure 3.4 is written in php, and it consist of mainly four parts. The top part of the php script, we see an 'authentication' part of the script', where the user has to supply login-credentials. The second part of this is to establish connection to the database itself. The third part is to provide SQL queries to the database. And finally, data is fetched and pushed to the UVisP for visualization.

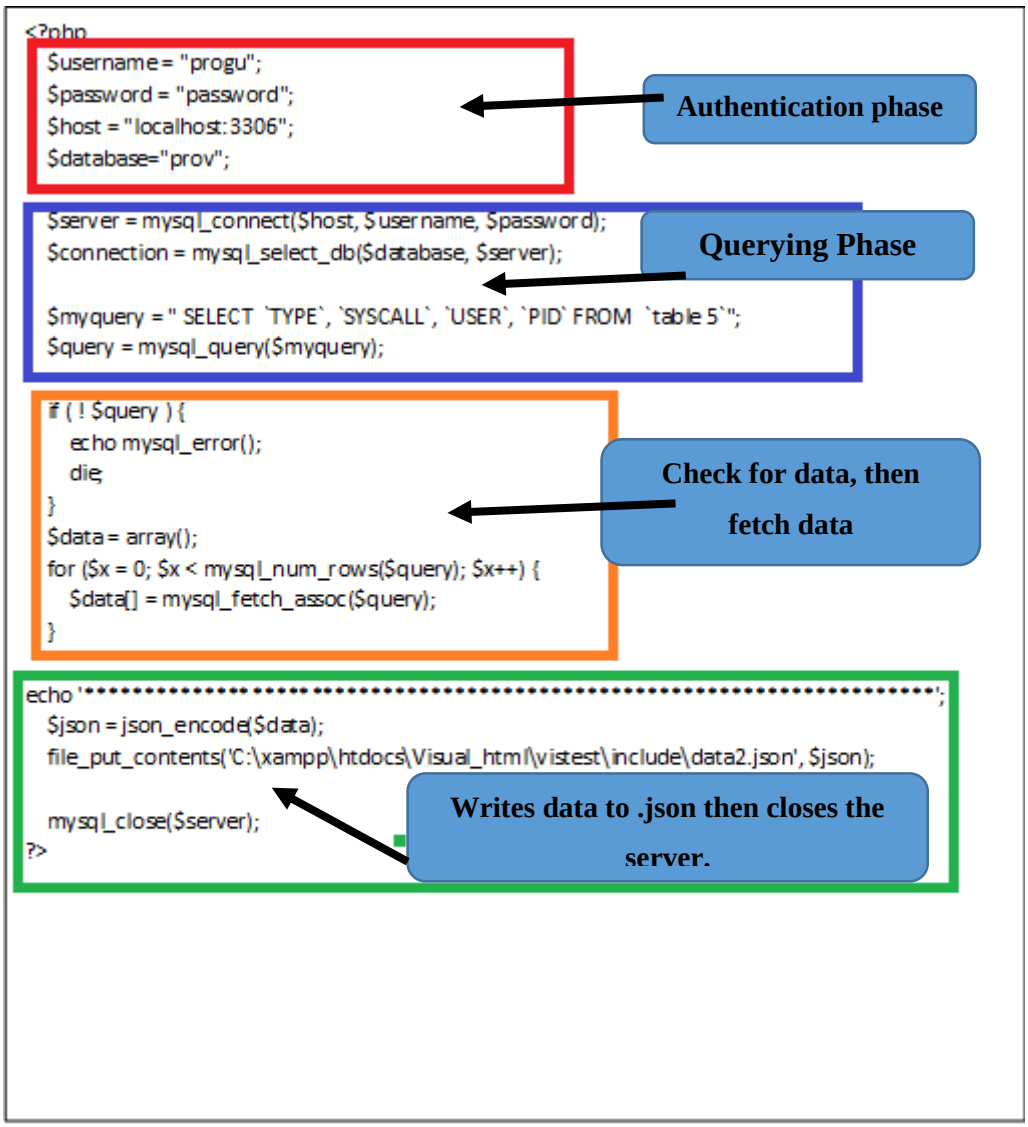


Figure 3.4. A snapshot of a 'getadata.php' file to connect onto the front-end of UVisP.

Step 8: UVisP queries for the different types of visualizations and outputs them for the user to visualize them.

F.2 HADOOP (APACHE HIVE) - DATABASE IMPLEMENTATION – STEPS

Step 1: Download Hadoop using the Hortonworks 2.2 Sandbox (virtual Image. Link: http://hortonworks.com/products/hortonworks-sandbox/

Step 2: Execute by importing the Hortonworks Sandbox virtual image using a virtualization environment such as Virtual Box.

Step 3: Configure network settings so that the windows host machine can communicate with the virtual Hadoop host.

Step 4: Access the Hadoop framework using the web browser: http://127.0.0.1:8888 or carrying out a "ssh root@127.0.0.1 –p 2222"

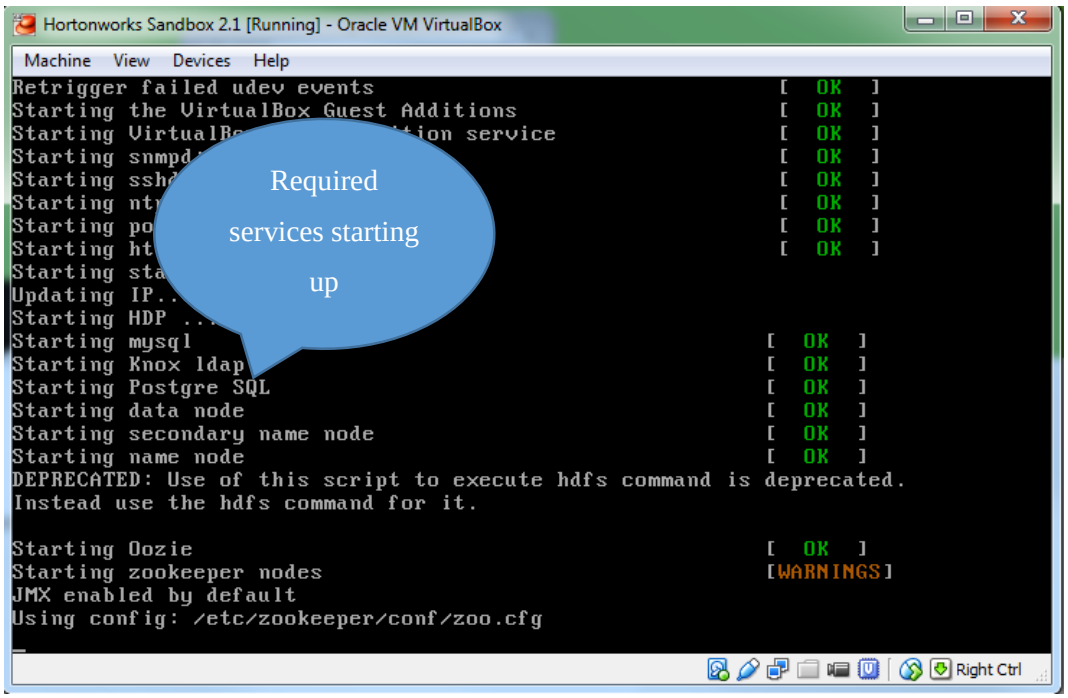


Figure 3. 6. A snapshot of the Hortonworks installation/import process in Oracle VirtualBox environment.

Figure 3.6 shows the pre-startup stages of the Hortonworks virtual image booting up.

Figure 3.7 shows the welcome page of the Hadoop – Hortonworks Sandbox 2.1 on this welcome page, there are sets of instructions on how to access the Hadoop framework.

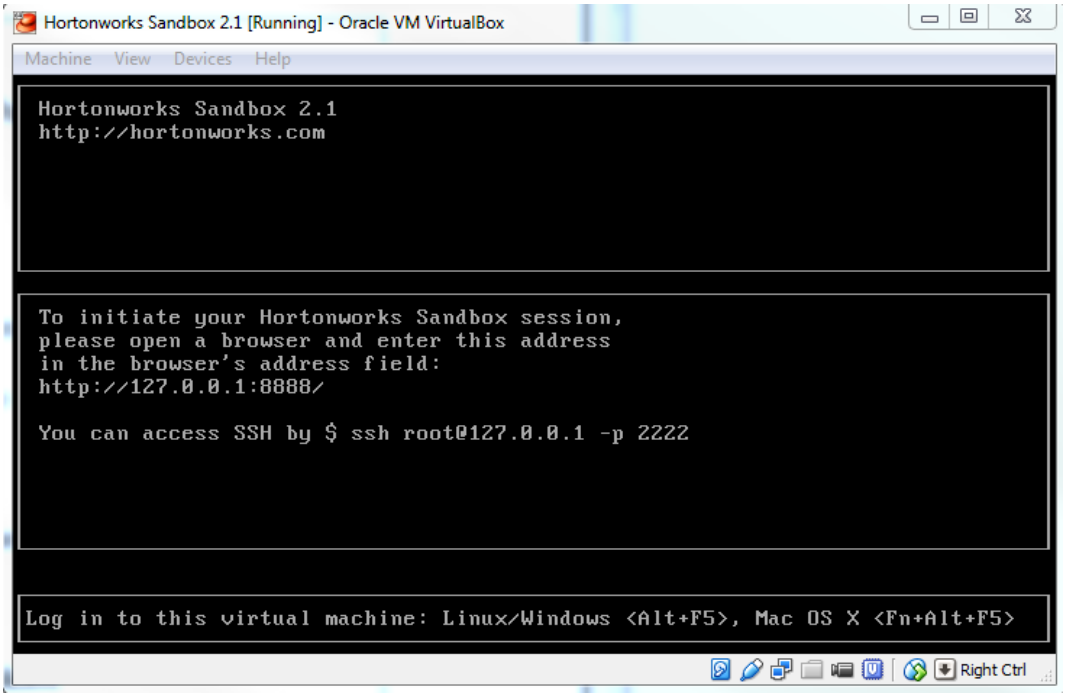


Figure 3.7. The Hadoop login interface.

Step 5: Create database in Hive and load data into the Hive database using the 'upload' feature in the Hadoop framework.

Step 6: Create a connection link between the UVisP and Hadoop – Hive database. This connection can be done using several options:

1. PHP Script: connection.php
2. DBMS connector
3. Thrift Server

Step 7: Create tables and load data into the Hive database.

Step 8: Create Pig queries using the Pig interface. All queries are saved inside the Pig Module for future use.

Step 9: All queries can be called using D3.js for the visualizations.