A Survey of Visualization Techniques for Network Security Analytics

Amit Tomar, Beena Kumari, Shivam Agarwal, and
Jaya Sreevalsan Nair

Graphics-Visualization-Computing Lab,
Center for Data Sciences,
International Institute of Information Technology Bangalore
http://cds.iiitb.ac.in/gvcl

February 15, 2015

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In today's world, computer networks have become ubiquitous. A network is a group of computers and peripheral devices connected by some means of communication links, example, wired or wireless, with the ability to share information among the network components. Examples of different networks are: LAN, MAN, WAN, Intranet, Internet, etc. Computer networks provide a means to share information across the world possibly faster than other facilities. It provides a lot of benefits to different parts of our society, for example, industry associations, corporations, professional societies, government and education, etc. such that now it has become the ubiquitous part of our life. It allows the user to vastly access remote programs and remote databases within the same organization or from a different organization. One of the pertinent challenges with computer networks is that it is vulnerable to security threats. It is very difficult to design a computer network free of security flaws because of economic and design constraints. These problems have created the necessity of an intrusion detection system which may provide some type of security or alert to the user against network-security threats.

"Intrusion detection system is a piece of software which runs on a computer network to identify malicious activity or policy violations like unauthorized use, misuse by system insiders and external penetrators and generates a report" [15]. Intrusion detection system gives some kind of trust to users regarding their data confidentiality, security, data and computer integrity. It monitors network or system activities for malicious activities, unauthorized access or policy violations and produces reports to a manager station. The basic principle behind the intrusion detection system is that intruder's behaviour will be different from that of a normal user. It captures the abnormality in the behaviour of the system or network and produces the report.

Intrusion detection systems are of two types: Network based Intrusion detection system and Host based Intrusion detection system [15]. Host based Intrusion detection system (HIDS) examines activities on each individual computer or host by installing anti-threat applications such as firewalls, anti-virus software on each of the system in a network. HIDS examines each computers internal operations like files accessed by the user on the system or applications which were used. In a Network Intrusion Detection System (NIDS), events between computers or network traffic are analyzed by installing anti-threat software only at specific points such as network interfaces like servers. NIDS examines network

traffic by examining individual traffic flowing through the network. Operational use of both Intrusion detection systems is different but otherwise their roots are almost similar. Both identify suspicious activities based on abnormalities in the usual pattern. To inspect all inbound and outbound network activities, NIDS and HIDS both systems are required in the intrusion detection system.

1.1 Need for Visualization in Security Analytics

The ID analysts monitor output of intrusion detection system such as event logs, network and firewall logs, etc. They sort and break down the outputs and separate the interesting or suspicious results from normal action. Then they further analyze the suspicious events in more detail by combining the different event logs together and/or analyzing the suspicious events over a period of time. These event logs are in textual format which are enormously large in size and quite complex in nature. This manual analysis of IDS's output is a real labor intensive task for security analysts which results in both false alerts and undetected attacks. To make intrusion detection system more effective for security analysts, information retrieval and information visualization techniques are combined into a single system known as a network visualization tool.

Data visualization plays a significant role in data analysis with respect to communicating summary/overview as well as providing the ability to explore data. A network visualization tool allows the security analysts quickly examine the large amount of information by rendering a millions of events and log entries in a single graphical view. Now a picture is worth a thousand log entries (words). It enables the analyst to not only visually identify the suspicious patterns, but also filters the data and performs queries.

As opposed to the obvious advantages, the limitations with current network visualization tools include usage of rudimentary visualization techniques for displaying results of intrusion detection system. Instead of only visualizing the final output of intrusion detection system, we can empower visualization to do more, such as, visualize the data flow at various components of the intrusion detection system. This can give us more insight into the malicious data to identify the security threats. Various visualization techniques can also be combined to analyze the information in more details.

1.2 Project Proposal

We have undertaken this project to demonstrate how visualization can be a value-add to products pertaining to analysing security systems. Given that our proposed visualization system will be used for enhancing security analytics, the goals of this project are:

1. Investigate and identify appropriate visualization techniques for results from execution of individual (pre-canned or user-defined) analytic components

2. Investigate and identify appropriate visualization techniques for results from execution of workflow-based composition of multiple analytic components

3. Identify suitable libraries which will help in visualization generation, presentation and navigation

1.2.1 Outline of the Document

We have identified the two usecases for applying visualization to: (a) data provenance, and (b) multivariate data. The former pertains to goals 1 and 2; and the latter to goal 3. Data provenance is necessary for developing workflows, hence we have done a literature survey on existing visualization techniques for data provenance in Chapter 2. Security logs, which is an important subject of visualization at RSA, falls in the category of multivariate data. Multiple variables and attributes are associated with communications or transactions across the networks. In Chapter 3, we have identified popularly used multivariate visualization techniques. Further, we have proposed introducing hierarchical modeling of the log data, which can be visualized using space-filling methods, as explained in Chapter 3. In Chapter 4, we discuss the rationale behind several design choices for our work. We provide details of our implementation, and discussions on the results of our work, in Chapter 5. In Chapter 6, we summarize our work done in the project.

In the rest of this chapter, we will introduce the underlying concepts in Information Visualization. For our work, given the non-spatially-referenced nature of the data of our interest, we will be used widely used information visualization techniques. Section 1.3 describes information visualization techniques and space filling methods for hierarchical structures are explained in Section 3.2. Composite visualization methods are briefly explained in Section 1.3.1.

1.3 Information Visualization

Information Visualization focuses on visualization of abstract-data using standard spatial representation methods. These spatial methods may be chosen based on three criteria as shown in Figure 1.1: abstract-data type, the visualization technique, and the interaction and distortion techniques.

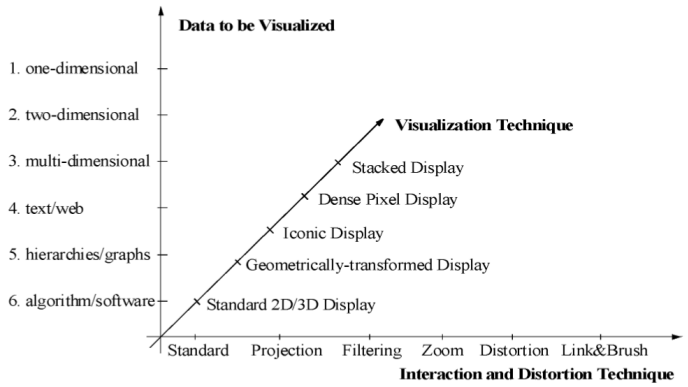

Figure 1.1: Classification of Information Visualization Techniques [17]

Abstract Data is data which do not encode spatial references explicitly. In information visualization, abstract datasets have different attributes or variables which are used to visualize them. The dimensions of data are equal to the number of variables they have. As shown in Figure 1.1, abstract data may fall into one of the following categories [17]:

- One dimensional data - It has one dimensional data such as temporal data. An example is time series of stock prices
- Two dimensional data - It has two distinct dimensions. Scatter plot is an example of a two dimensional visualization techniques
- Multi-dimensional data - Data sets consists of more than three attributes. Relational data-bases are an example of multidimensional data. Parallel coordinate system is one of the multi-dimensional visualization techniques used for multi-dimensional data
- Text and Hypertext - These data types are not in a number form. Therefore, Standard visualization techniques cannot be used. To visualize them, they are transformed into description vectors and these vectors are used for visualization purpose. An example is a word count.
- Hierarchies and graphs - These are discussed in details in next sub-section.
- Algorithms and Software- The idea behind this visualization is to support software development by understanding the algorithm. An example is a flow-chart, representing structures of source line codes as graph etc.

Visualization Techniques Various types of visualization techniques are used for visualization of data. Different visualization techniques can also be combined to gain more insight into a data. This method is know as composite visualization. Composite visualization paradigm is discussed in details in Section 1.3.1. From Figure 1.1, we can see that major categories of visualization techniques are [17]:

- Standard 2D/3D displays - X-Y (X-Y-Z) plots, bar charts, line graphs, etc.
- Geometrically transformed displays - Aim at finding "interesting" transformations of multidimensional data sets. Examples are scatter plot, hyper-plot, etc.
- Icon-based displays - The idea is to map the attribute values of a multi-dimensional data item to the features of an icon. Examples are star-plot, sticky figures etc.
- Dense pixel displays - Here idea is to map each dimension value to a colored pixel and group the pixels belonging to each dimension into adjacent areas. For example circle segment techniques
- Stacked displays - Partition the data and present it in a hierarchical fashion like tree map

These visualization techniques are discussed in more details in chapter 2.

Interaction and distortion techniques This provides the facility of user interaction with the data. User can dynamically interact with data to do some changes like filtering of data [17]. Different types of Interaction and distortion techniques, as shown in Figure 1.1, are:

1. Interactive Projection - Dynamically change the projection to analyze multi-dimensional data-sets
2. Interactive Filtering - Filter out the data based on some criteria and focus on the area of interest
3. Interactive Zooming - Interactively zoom-in or zoom-out the data to see the details at various levels
4. Interactive Distortion - It is a widely used technique in data-exploration field, also known as drill-down operation. Here, some portions of data are present with high details while some are present with low details
5. Interactive Linking and Brushing - Sometimes single visualization technique does not give complete information about data. The idea is to combine different visualization techniques to gain more insight into data. Data-set is visualized using different techniques in a single view. Certain portion of data can be highlighted interactively in one visualization which is automatically reflected into other visualizations.

Composing visualizations using various techniques is necessary to enrich the visual summarization as well as exploratory processes. We have discussed arious aspects of composite visualization in Section 1.3.1.

1.3.1 Composite Visualization

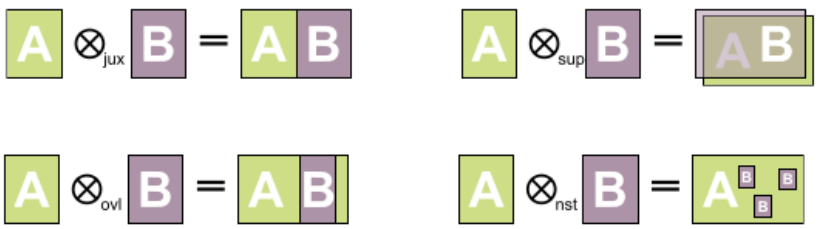

Figure 1.2: Four different visual composition operators : juxtaposition, superimposition, overloading, and nesting [16]

Composite Visualization is a theoretical model to visualize different visual structures in a single visual space [16]. In literature, there are mainly three different methods for composite visualization:

- CMV (Coordinated multiple views)
- Node Trix

- Spark Clouds

Although these three methods are widely used for composite visualization, CMV alone has been formally defined in literature. CMV uses juxtaposition method and different linking mechanisms to visualize multiple visual structures in a single geometric space.

In the paper [16], composite visualization model (CVV) are proposed to visualize multiple visual structures in one geometric space using existing visualization methods. CVV has identified the five steps as a design pattern to merge different visualization techniques into one as shown in Figure 1.2. Those five design patterns are listed below:

- Juxtaposition - Placing visualizations side-by-side in one view (Coordinated Multiple Views)
- Integration - Placing visualizations in the same view with visual links
- Overloading - Utilizing the space of one visualization for another
- Superimposition: Overlaying two visualizations in a single view
- Nesting: Nesting the contents of one visualization inside another visualization

Chapter 2

Visualization of Provenance

The goals of our work include identifying visualization techniques which will enable analytics workflow. Essentially workflow-related visualizations are synonymous to visualization of provenance of specific data, or outcomes.

In Section 2.1, we have discussed the use of provenance of data as a viable data model for a visualization system. Provenance is the history of how a data object originated and reached its current form after application of various operations on it. It may be helpful in securing related log analysis, on when data objects can have more than one accessing entities with different privilege of accessing the data. It may also be helpful in tracing the error, illegal access, validation, etc. that could have brought an object to its current state.

2.1 Provenance

Provenance information is a lineage of a discrete collection of information and data, along with all its entities and process starting from the origin, responsible for its current state. The goal of generating provenance is to create an accurate recorded ancestry of all constituent components and all meta-data involved at various stages of information transformation. Provenance can be used to analyze various changes that have been made in the complete workflow of a system. For example, a user can query whether a particular information object has been changed or not and which of the network components produced that change, and how it has affected the attributes of data objects. It helps users to make various decisions about information like, whether a particular piece of information is trusted or not, how to integrate information objects from different components and how to analyze the output at different components.

Provenance is of importance in many real-world applications such as: web of data, security analytics, medical imaging, forecasting systems, etc. A phylogenetic application to demonstrate the importance of provenance information is explained in Section 3.2. Here, large datasets are interlinked to retrieve a particular information based on user query. Provenance information of the application may help users to trust the accuracy and reliability of retrieved information [13]. This example shows that provenance is of paramount importance in many real-world applications.

Provenance information has been broadly divided into two classes: work-flow

provenance and data-flow provenance. Work-flow provenance is the pedigree of computational steps involved in a program as well as human-machine interaction steps. Work-flow provenance has varied amounts of information based upon the application. It can be used as a "proof of correctness" and can also be used to "avoid duplication of efforts" in the system [27]. On the other hand, data provenance gives a detailed recorded history of a single information object [27].

2.2 Provenance and Visualization System

In many areas, visualization is used to explore, analyze and visualize the scientific results such as large datasets and maps. The concept of visualization can be used to view final results or even, intermediate outputs of the system. Visualization systems may not be enough for scientists to completely understand, validate and accept the result or infer some other conclusion. They do not give information about the origin of knowledge, involved processes or attributes of intermediate meta-objects. Their only focus is to represent output in graphical forms. Visualization techniques can use the provenance information to not only visualize the final result but also partial results and intermediate processes [26]. It may also help us to analyze the process and infer more information accurately.

2.2.1 Scientific Workflow

Scientific workflow system is used to capture, manage and query provenance information of a real-time system. It can also be defined as a method to explore and analyze data using workflow specifications. The workflow specification can be thought of as a graph where each vertex represents the module of a system and edge represent the in-out data flow between modules. For example, consider Figure 2.1, it represents the workflow specification for common analysis in biology. M1 takes a set of sequences from general bank and performs an alignment process on them. M2 refines the alignment and feeds the output to M3. M1 and M2 together form the M7 process. M3 randomly chooses a seed and provides it to M4; M4 creates a set of phylogenetic trees with the help of a seed provided by M3 and M2 processes. M5 performs conditional checks to establish if the search space can be adequately sampled or not. If it is not sampled properly, then process repeats from M3 otherwise it feeds the output to M6 and M6 now creates an output tree. M7, M8 and M9 are composite modules. Composite modules are discussed in details in section 1.3.1.

Such a system runs multiple times and generates large amount of intermediate information which may enable scientists to explore, analyze data-sets and draw more inferences. However, it is difficult to capture this intermediate information manually. Scientific workflow systems store provenance information dynamically at run time. They capture the information in the form of event logs such as the start and end time of a process. It not only helps in interpreting results and providing reproducible result but also helps in troubleshooting and optimizing efficiency [11].

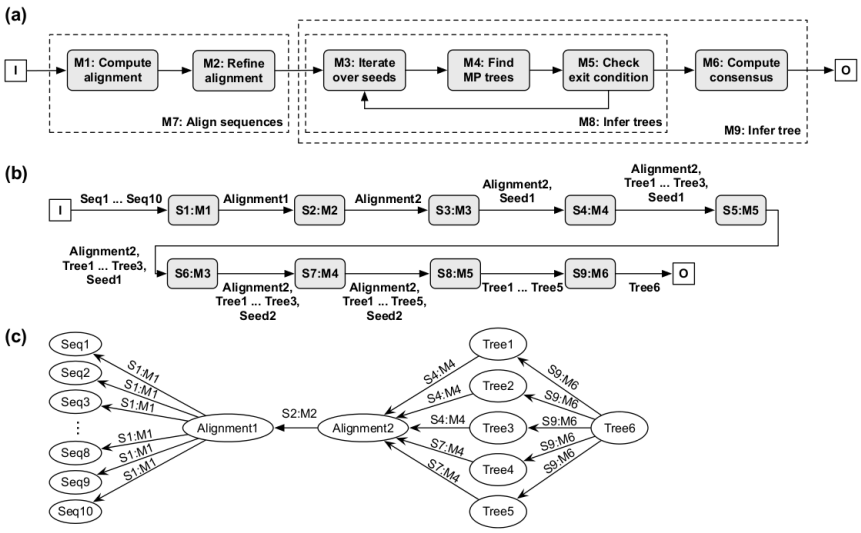

Figure 2.1: Phylogentics workflow specification, run, and data dependency graph [11]

2.3 Relevance in Security Analytics

Implementing separate visualization for individual components and having a way to compose visualizations when composing these components in a workflow, will give a visual guidance to a security analyst to build workflows for specifically security data. The provenance of intermediate components and inspection of its current state will also enable the analyst to execute proof of concept for a customized workflow.

Composable workflows is very nascent in data analytics, and holds promise in enriching the domain of security. Examples of composable analytics include a web-based software platform developed in [12] and an architecture for using several machine learning for several problems including cyber security [25]. Our proposition is that embedding a framework/architecture running composable analytics with visualization will go a long way in decision making. We state that visual depiction in such a scenario can derive classifications provided by the theoretical model of composite visualizations [16].

Chapter 3

Visualization of Security Logs

One of the goals of our work is to identify appropriate visualization techniques for intuitively visualizing security log data. Security log data is inherently multivariate in nature. In information visualization parlance, it may be considered to be multidimensional data prima facie. We propose that additionally deriving hierarchical organization of the data will expand the set of visualization techniques one can use with the data.

In Section 3.1, we focus on the existing widely used visualization techniques for multidimensional data. In Section 3.2, we focus on the hierarchical data visualization techniques one may apply on the log data, once the hierarchical data model is derived from the data.

3.1 Multivariate/Multidimensional Data Visualization

Multidimensional data refers to the dimensionality of the independent dimensions while multivariate data refers to that of the dependent variables [9]. Multivariate data are encountered in all aspects by researchers, scientists, engineers, manufacturers, financial managers and various kinds of analysts [9]. In information visualization, one may assume based on the usual scenario of linearly independent variables, hence the adjectives, "multidimensional" and "multivariate" are used interchangeably.

Most familiar plots for data can accommodate up to three dimensions adequately. Visualization techniques for datasets that have more than three variables are required. There can be different approaches based on which the visualization technique can be classified : goal of the visualization, the types of the variables, mappings of the variables, etc. In this section, we discuss some of such techniques, stating their advantages and disadvantages, wherever applicable.

3.1.1 Scatterplot Matrix

Scatter plots are organized in a matrix format by displaying all of the pairwise correlation information at one place. As shown in Figure 3.1, panel at the i^{th} row and j^{th} column is a scatter plot of X-j versus X-i. Panels that are symmetric with respect to the XYZ diagonal have same variables as their coordinates, rotated 90°. This redundancy is designed to improve the visual linking, so that the patterns can be detected in both horizontal and vertical directions [22]. Problem with scatter plot is that they becomes chaotic when number of data items are too large [9].

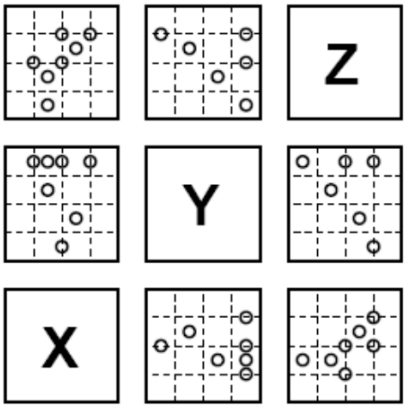

Figure 3.1: Scatter plot visualization [22]

3.1.2 Hyperbox

Hyperbox is a 2-dimensional depiction of a k-dimensional box involving pairwise 2D plots. Panels that are symmetric with respect to the XYZ diagonal have the same variables as their coordinates are rotated 90° as shown in Figure 3.2 . Length of line segments and angles between them are arbitrary [22]. Hyperbox contains $k \times k$ lines and $(k \times (k-1))/2$ faces.
In Figure 3.3, lines 1, 2, 3, 4, and 5 form a direction set. Similarly, lines i, ii, iii, iv and v also form a direction set. Here five variables X, Y, Z, W and U are mapped to these direction sets. Each face of this hyperbox can be used to display 2-dimensional plots, for example, line chart, scatter plot, etc.

3.1.3 Chernoff Faces

Chernoff faces consist of a simplified image of a human face, which is used as a display as shown in Figure 3.4. Different data variables (attributes) are mapped to different facial features of the face.

3.1.4 Stick Figures

Stick figures technique involves representing the data in the form of stick like figures. Figure 3.5 represents the Stick figures. Two of the most important variables from the dataset are chosen and mapped to the two display dimensions.

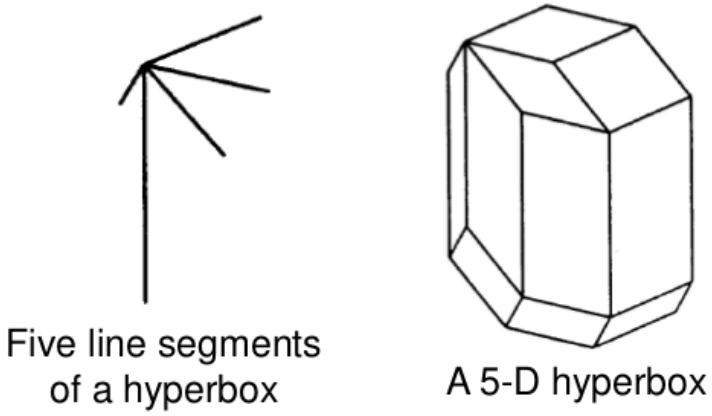

Figure 3.2: Hyperbox [22]

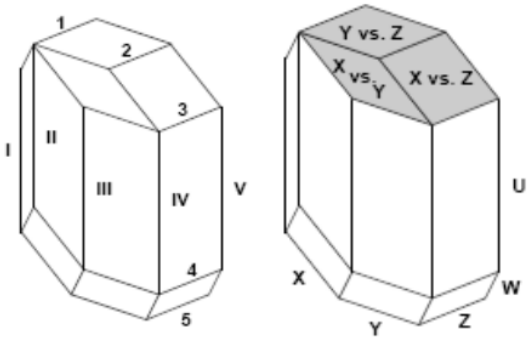

Figure 3.3: HyperBox [22]

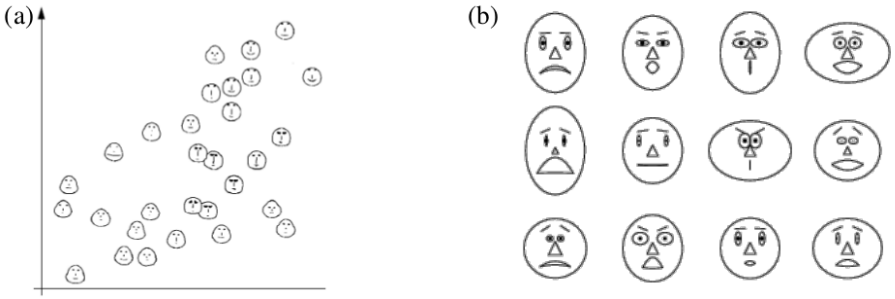

Figure 3.4: Chernoff Faces [9]

Rest of the variables are mapped to angles/length of the limbs of a stick. If the data records are relatively dense, the resulting visualization presents texture patterns as shown in Figure 3.6. Texture patterns vary according to the characteristics of the data and are therefore detectable by pre-attentive perception [22]. Stick figures have been found helpful in finding out the outliers in the data sets.

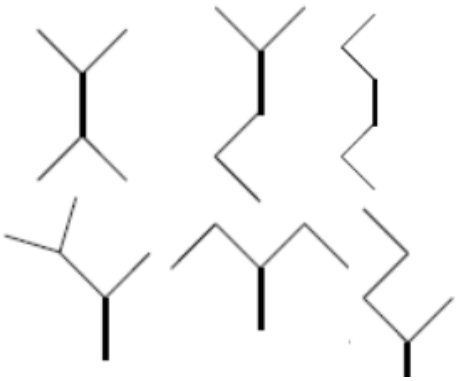

Figure 3.5: Stick Figures [9]

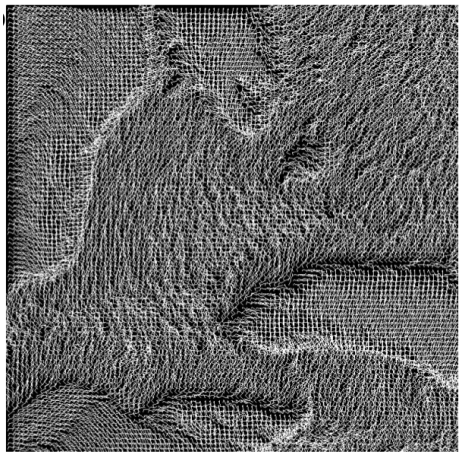

Figure 3.6: 5D - Stick Figures [22]

Visual inspection of such 'textural information' can give insight into the structuring of data.

3.1.5 Star Plots

In star plots, each data record is represented as a star-shaped figure, where one ray represents a variable as shown in Figure 3.7. The length of each ray is proportional to the value of its corresponding variable. Each variable is usually normalized between 0 and 1 as shown in Figure 3.8. In Star plots, as number of rays increases, it becomes increasingly difficult to separate rays.

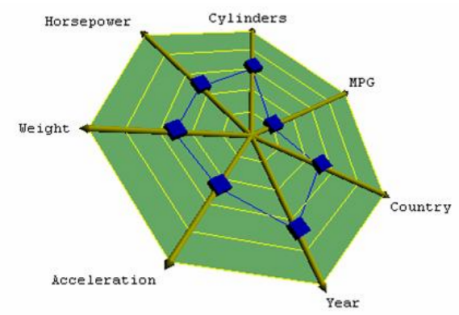

Figure 3.7: Star Plots [9]

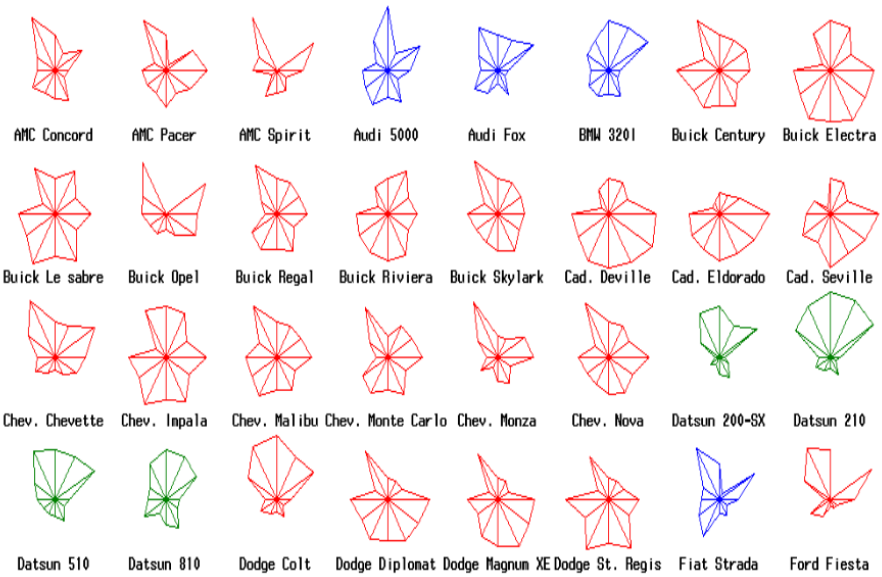

Figure 3.8: Star Plots [22]

3.1.6 Circle Segments

Circle segments represent the data in the form of circle segments as shown in Figure 3.9. If the dataset consists of k variables, the circle is partitioned into k segments, each representing one variable [22].

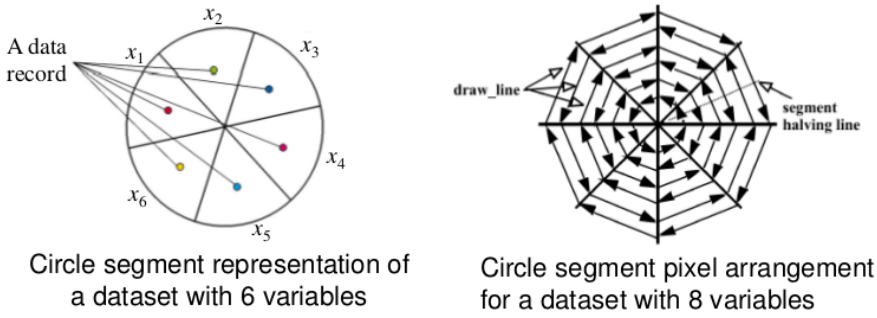

Figure 3.9: Circle Segments [22]

3.1.7 Parallel Coordinates

Parallel coordinates represent multi-dimensional data by plotting the data over multiple parallel axes. K variables are organized as K uniformly spaced vertical lines in a 2-dimensional space as shown in Figure 3.10. Variables are usually normalized. Parallel coordinates have been found to be effective in revealing relationships between adjacent axis variables. They are also effective in showing the distributions of attributes [22].

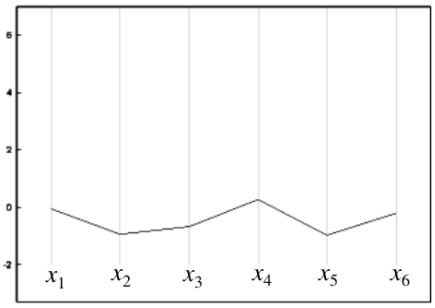

Figure 3.10: Visualization using parallel coordinates, where a point (0,-1,-.75, .25,-1, -.25) is shown [22]

The order in which the axes are arranged matters as it can reveal different kind of information based on different arrangements. With large size of data sets, cluttering can occur. Clustering makes the information less intuitive. In those cases, brushing and linking techniques are used for better analysis of data. The brushing and linking allow to club together the data over some very small neighbourhoods as shown in Figure 3.12. Filtering over this data can also be done to get better results. Parallel and Cartesian coordinate systems show some

dual behaviours which can be helpful in better understanding of data. Some of the dual behaviours are listed below:

1. A Cartesian point is a line in parallel coordinates.
2. A Cartesian line is a point in parallel coordinates.
3. Translation transformation in Cartesian is a rotation in parallel coordinates.
4. Rotation transformation in Cartesian is a translation in parallel coordinates.
5. Hyperbola in Cartesian is an ellipse in parallel coordinates.

For different domains and problem sets, we can analyze the commonly occurring visual patterns in the parallel coordinates system and use them as references while analyzing the data. One such work was done by Hyunsang et. al. by providing visualizations of network attacks based on the parallel coordinates system. Figure 3.11 and Figure 3.10 shows some of the common visual patterns that appear in parallel coordinates systems while analyzing network attacks.

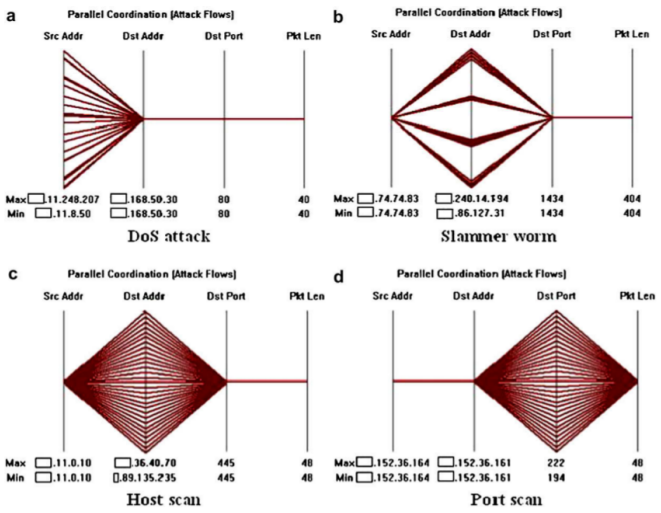

Figure 3.11: Parallel Coordinates : Attacks [10]

3.1.8 Color Icons

In the color icons technique, area on display (a circle or rectangle usually) is subdivided and mapped to the multivariate data as shown in Figure 3.14. This mapping can be a linear or area based. This is explained below.

1. **Linear mapping** : Up to 6 variables can be mapped to the icon, 2 of edges, 2 diagonals, 2 mid-lines. A color is assigned to each thick line according to the value of the corresponding variable.

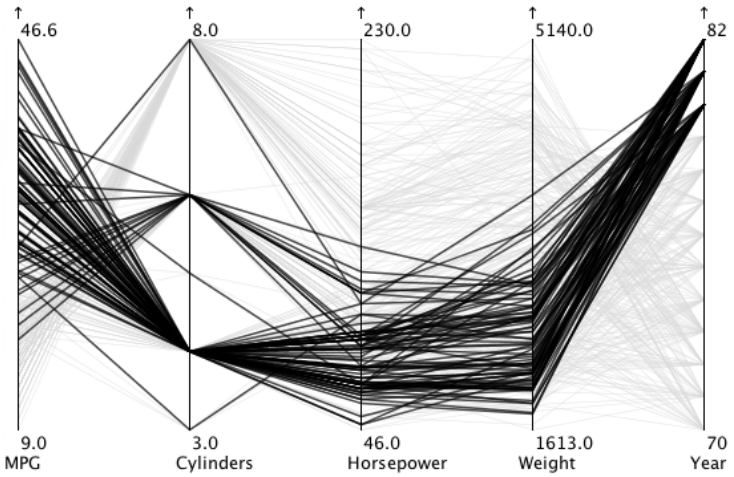

Figure 3.12: Brushing and Linking [19]

Table 1 – Graphical signatures of nine attacks.

Implied Attack	Signature	Divergences
Portscan		1:1:*m*:1
Hostscan		1:*m*:1:1
Worm		1:*m*:1:1
Source-spoofed DoS (port fixed)		*m*:1:1:1
Backscatter		1:*m*:*m*:1
Source-spoofed DoS (port varied)		*m*:1:*m*:1
Distributed hostscan		*m*:*m*:1:1
Network-directed DoS		*m*:*m*:*m*:1
Single-source DoS		1:1:1:1

Figure 3.13: Parallel Coordinates : Attack Signatures [10]

2. **Area mapping** : Each subarea (totally 8) corresponds to one variable. A color is assigned to a subarea according to the value of its corresponding variable.

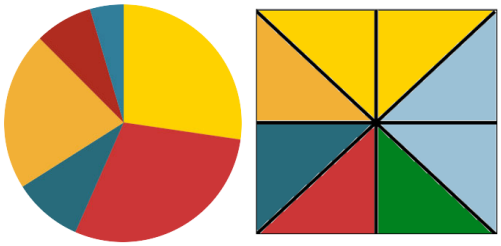

Figure 3.14: Color Icons [9]

3.2 Hierarchical Data Visualization

In this section, we will discuss about the space-filling methods which are generally used for visualizing hierarchical data.

Space Filling Methods for Hierarchical Structures: A large amount of information in the world occurs in the form of some hierarchical structures. Trees are one of the most common structures used to hold relational information among entities. But they often grow rapidly and widen up to not comfortably fit the view area. To avoid such situations, space-filling visualization techniques have been used to make maximum possible usage of the view region on a display.

3.2.1 Sunburst Display

A Sunburst layout is a space-filling visualization technique that displays a radial layout of concentric circles to show the relationships among nodes as shown in Figure 3.15. Centre of this layout represents the top of the hierarchy. Moving away from the centre radially corresponds to moving down from root towards the leaf node of a tree structure. These rings in the layout are further divided into segments to represents the number of nodes at a particular level [7]. The area swept by a segment and its color represents some attributes of the data [4].

Mansmann et. al [24] have shown that how sunburst visualization technique can be used to logically group the complex rules associated with firewalls according to their common characteristics in a concise manner. They grouped the complex firewall rules into access control lists and access groups, thus allowing application of security policies on to groups rather than on individual entities.

To represents them as Sunburst visualization, the first level (after the centre) contains segments of different access lists. Different types of privileges are present at second level. Third level stores different networking protocols, while different sources and destinations follow in the fourth level in the form of different object groups. Identical rule elements are clubbed together by grouping them recursively in above mentioned manner [24]. They go on to provide

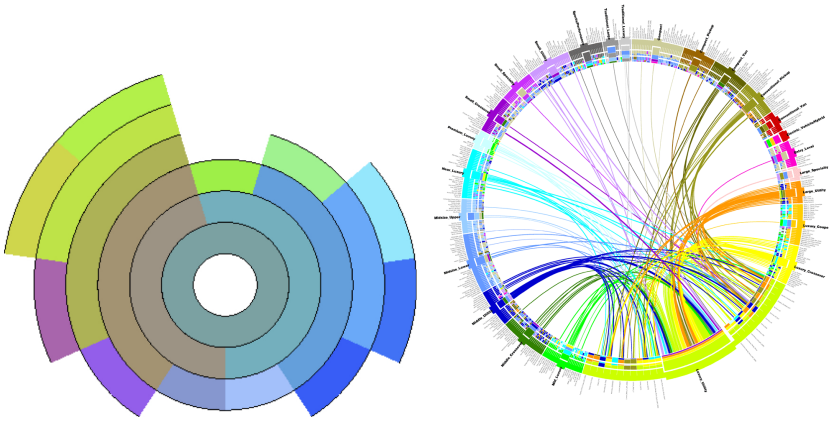

Figure 3.15: (Left) A Sunburst display [28]; (Right) radial layout [20]

functionality to expand individual nodes, search by keyword and visualize the relations among different elements. Lee et. al [21] have proposed several visualizations for real time network security monitoring to form a sort of 'visual firewall'.

3.2.2 Radial Plots

Radial layouts display nodes as sectors in an outer ring and show edges as curves connecting these sectors to the center of the ring [8]. Radial plots emphasize on the relationships among nodes rather than their spatial presence [5]. As shown in Figure 3.15, radial layouts sometimes employ a technique called *edge bundling* to club together parallel edges which have origin or destination in the vicinity, so that number of edges crossing each other can be reduced. In radial layouts, edges are sometimes bundled to increase readability and reduce edge crossings. Edge bundling is a technique that groups parallel edges with nearby origin or destination as if they were cinched together [5].

Borkin et. al [5] have shown how we can use radial plot for provenance aware storage systems. Different processes run on operating system performs different files operations such as creation, deletion, modification and change in location of files, etc. on the file system of an operating system. The File provenance of operating system involves keeping track of all these operations from the time a file is created. Provenance data has been shown to increase the discovery speed of virus entry-points into a system to aid security by allowing more efficient patching of software, and to prevent future malicious attacks [8]. Processes, files, communication channels between processes, non-provenance files, etc. can be represented as nodes. Then, the edges can be used to represent the dependency relationships between these nodes. For example, edges could represent "a process modifies a file" [8]. Depending on the kind of a node, there can be different attributes associated with it, e.g., process ID, file path, etc. These attributes help in getting a clear understanding of what is going on in the system.

3.2.3 Treemaps

Treemaps are space filling visualization techniques which are rectangular in shape. To begin with, a rectangle is taken and recursive division is done by alternative horizontal and vertical partitioning of the rectangle based on the population of the subtrees at a given level [28]. The size and the color of the rectangles can be used to represent different attributes of the data as shown in Figure 3.16. The inherent difference between treemaps and techniques, discussed earlier, such as, radial plots and Sunburst display, is that treemaps traverses a tree inwards (from its bounding box/polygon); while the others expand radially outwards.

In the paper [24], it has shown that how treemaps visualization can be used to support automated intrusion detection systems for analyzing network traffic and intrusion detection events. To analyze the large set of logs generated by the intrusion detection systems, a treeMap visualization of local network hosts connected with external hosts is prepared and analyzed by security analysts for better understanding of attacks on the network [23].

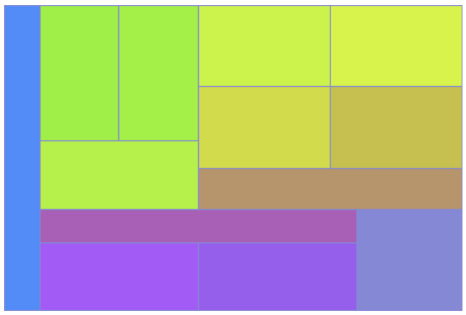

Figure 3.16: A Treemap [28]

Chapter 4

Proposed Solution

Over the past decades or so, web application has been embraced by millions of enterprises and organizations because of availability of internet. Internet has become an in-expensive channel to exchange information across the world. Using the web-browsers, we can explore and exchange huge amount of information at the cheapest cost. Web-based applications enable us to capture, store, processed and transmit data at a very fast rate. Example of web-applications are: shopping cart, on-line banking, ticket-reservation, face-book, etc. Because of it's day-to-day usage, it has become a ubiquitous phenomenon in our daily life.

The aim of our project is to develop a dynamically interactive network visualization tool which is easily pluggable in web-browsers. In Section 4.1, we discuss various technologies available for developing browser plugins as well as for enabling information visualization.

In Section 4.2, we discuss about the design choices we have made for web technologies as well as visualization techniques for implementing visualization of security logs as proof of concept.

4.1 Web Technologies

To build our application, we will use standard web-technologies such as HTML, CSS and JavaScript. Document Object Model (DOM) is generally used for web-applications. The basic model of these technologies is DOM based but their implementation varies for different browsers.

In Section 4.1.1, we briefly discuss about HTML and CSS. In Section 4.1.2, we discuss about JavaScript and it's alternative languages. We have also discussed a few visualization tool-kits in Section 4.1.3.

4.1.1 HTML and CSS

HTML stands for Hypertext Markup Language[1] and it is developed by a scientist Tim Berners-Lee in 1990. The purpose was to make it to design semantics for scientific documents so that researchers at different universities can access each other's work easily. Later it laid the foundation for the World Wide Web (WWW). It is a text markup language which is used to create documents on

[1] `http://www.w3schools.com/html/html_intro.asp`

the web. It is a hidden code in a web-browser which helps browser to create the structure and layout for a web-document. Every page on Internet contains HTML code. It is just like a wrapper used to structure contents for web-browsers by using a variety of tags ant attributes.

HTML concentrates only on structure of text rather than visual appearance. To make the page visually more appealing, CSS style is applied on web page. Cascading Style sheets or CSS was invented to describe the presentation of a document written in markup language. CSS describes look and format of an document element on web-page like color, size, shape of elements, etc. CSS enables us to separate the presentation style such as layout, colors and fonts from the contents of documents.

4.1.2 JavaScript

JavaScript is a lightweight scripting language[2] used for Web pages. It provides a facility to control the browsers, alter the web-document contents, etc. JavaScript manipulates the internal Document Object Model (DOM). It provides various features for web-pages such as check forms, customize graphics selectioin, widgets, graphics and many more. JavaScript is most widely used scripting language but it has few drawbacks also. For example, there is no namespace, import statement and access modifiers in JavaScript. To get rid of these shortcomings of JavaScript, number of other scripting languages have been developed over a time. Three major JavaScript alternatives are:

- CoffeeScript
- TypeScript
- Dart

CoffeeScript It has been developed by Jeremy Ashkenan in 2000. It is an alternative syntax for writing JavsScript. The syntax of CoffeeScript looks like a combination of Ruby and Python. Therefore, people with a python background will find it natural. It adapts the good part of JavaScipt but with better syntax and very less code. It also provides a framework for classes and inheritance which are greatest pitfalls of JavaScript. CoffeScript can directly compile into corresponding JavaScript.

TypeScript Microsoft has developed the TypeScript in an effort to solve the issues they have found with JavaScript. TypeScript is a strict superset of JavaScript and therefore, any valid JavaSript code is also a valid TypeScript Code. TypeScript doesn't hide any pitfalls of JavaScript. But it has added a few features to JavaScript such as static typing and class framework. Class framework has made the job of developers easier by allowing modularization of complex codes. Static typing offers many advanced refactoring operations such as global renaming, reference searching, code completion, etc. In TypeScript, problems can be detected during compile time rather than at runtime. Microsoft also provide IDE support for TypeScript in Visual studio.

[2] `http://www.w3schools.com/js/DEFAULT.asp`

Dart It is developed by Google and it's first beta version appeared in 2011. It's a whole new language and designed to solve many issues which Google faced with JavaScipt. It tends to draw java or C# programmer towards it because of it's resemblance to Java or C# in many ways. Dart can also be used as a general usage language. It has its own IDE which provides lots of good features for developers.

Features	**JavaScript**	**CoffeeScript**	**TypeScript**	**Dart**
Static Type Checking	-	-	X	X
Classes	-	X	X	X
Interfaces	-	-	X	X
Modules	-	X	-	X
List Comprehensions	-	X	-	-
String Interpolations	-	X	-	X
Splats/Rest parameters(...)	-	X	X	-
Intellisense	-	-	X	X
Code Brevity	-	X	-	-
Better speed	-	-	-	X

Table 4.1: Comparison of language features [1]

Table 4.1 shows the comparisons between CoffeeScript, TypeScript and Dart. It shows different features that are added by different languages like CoffeeScript, TypeScript and Dart to the JavaScript. In our application, D3 is used to create the visualizations for network data as it provides all necessary visualization tools for our application framework.

4.1.3 Visualization Toolkits

There are several visualization toolkits available. Some of them are discussed below:

D3.js

D3.js (Data-Driven Documents) is a free and open-source Javascript library developed by the Stanford Visualization Group (Mike Bostock) in 2011. D3.js uses digital data to drive the creation and control of dynamic and interactive graphical forms which run in web browsers. It is a tool for data visualization in W3C-compliant computing, making use of the widely implemented Scalable Vector Graphics (SVG), JavaScript, HTML5, and Cascading Style Sheets (CSS3) standards. It is the successor to the earlier Protovis framework. D3 emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation [6]. D3 has the following advantages.

- It has full capabilities of modern browsers
- Input data which can bind to a DOM can be of following forms: Comma Separated Value, JavaScript Object Notation, eXtensible Markup Language files
- It is fast even in case of large datasets
- Dynamic interactive visualizations can be realized
- Easy to debug and maintain

Processing and Processing.js

Processing is a programming language, development environment, and online community. Processing.js is a 'port' of Processing, designed to make data visualizations, digital art, interactive animations, educational graphs, video games, etc. work using web standards and without any plugins. Processing.js bridges the gap between these two powerful technologies [18] [4].

Prefuse

Prefuse is a set of software tools for creating rich interactive data visualizations. The original prefuse toolkit provides a visualization framework for the Java programming language. The prefuse flare toolkit provides visualization and animation tools for ActionScript and the Adobe Flash Player [14].

JavaScript InfoVis Toolkit

The JavaScript InfoVis Toolkit provide tools for creating Interactive Data Visualizations for the Web. The toolkit implements advanced features of information visualization like TreeMaps, an adapted visualization of trees based on the SpaceTree, a (focus + context) technique to plot hyperbolic trees, a radial layout of trees with advanced animations (RGraph) and other visualizations [2] [3].

4.2 Design Choices

Chapter 5

Implementation of Log Visualization

To understand and check the capabilities of D3.js, we have implemented a few visualization techniques, like bar graph and scatter plot, in the sand box. After a detailed survey of visualization techniques and discussions with EMC-RSA team, we have found that parallel coordinates might be a good choice for security related requirements. Therefore, we have decided to use parallel coordinates in our application to visualize the multi-dimensional network data. Additionally we performed a hierarchical modeling on the data and implemented hierarchical data visualizations, such as treemap, Sunburst, and radial plots on the data.

In Section 5.1, we describe our implementation of the visualization techniques in the sandbox. In Section 5.2, we describe our proposed architecture and implementation of visualization techniques on the mockup of the RSA product, "Investigator", provided by EMC^2-RSA, as a proof of concept.

5.1 Sandbox Implementation

In this section, we explain our implementation of several visualization techniques on synthetic data,

5.1.1 Bar Graph

Bar graphs are a common visualization technique for visualizing data. D3.js however doesn't provide any in-built function to build any visualization directly. What is available is the fundamental primitives which can be used to build any visualization as per the requirements. As shown in Figure 5.1[1], we have created a bar-graph by using a randomly generated login data-set. The format of data is:

{User Name , Monthly Card Swipes}

Following operations are provided over the data:

- Append random data

[1] http://sec-viz.appspot.com/BarGraph.html

- Delete last entry of the data
- Refresh the data set with some random data
- Sort the data

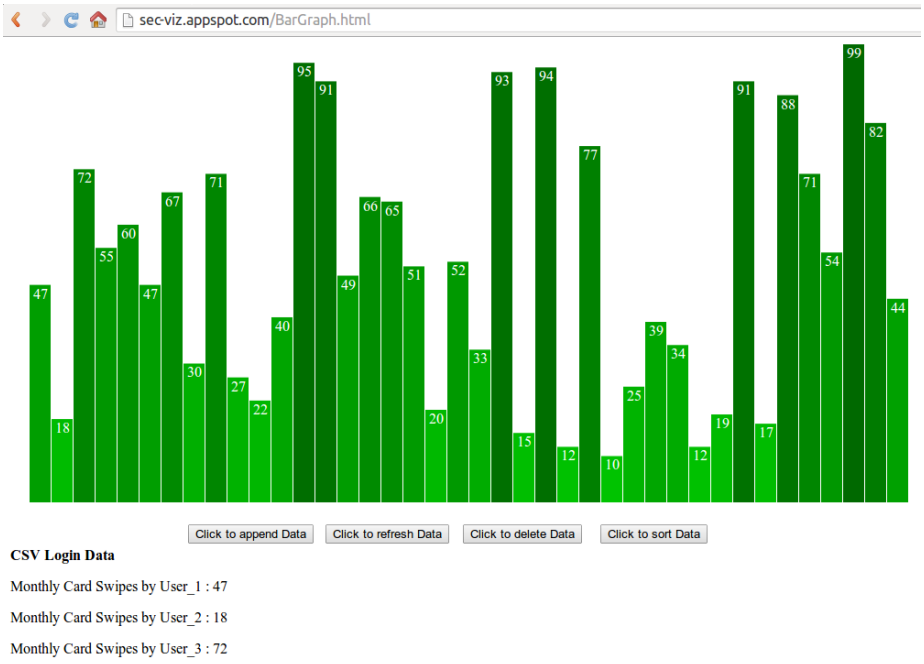

Figure 5.1: Bar graph showing login-data

References for the code is available at the following urls:

- `http://chimera.labs.oreilly.com/books/1230000000345/ch06.html#_making_a_bar_chart`
- `http://bost.ocks.org/mike/bar/`
- `http://mbostock.github.io/d3/tutorial/bar-2.html`

5.1.2 Scatter Plot

Scatter plot is a widely used visualization technique for multi-variable data. We have implemented the scatter plot with the help of randomly generated pairwise data as shown in Figure 5.2[2]. The widget is provided to refresh the data. References for the code :

- `http://chimera.labs.oreilly.com/books/1230000000345/ch06.html#_the_scatterplot`
- `http://bost.ocks.org/mike/bar/`

[2]http://sec-viz.appspot.com/ScatterPlot.html

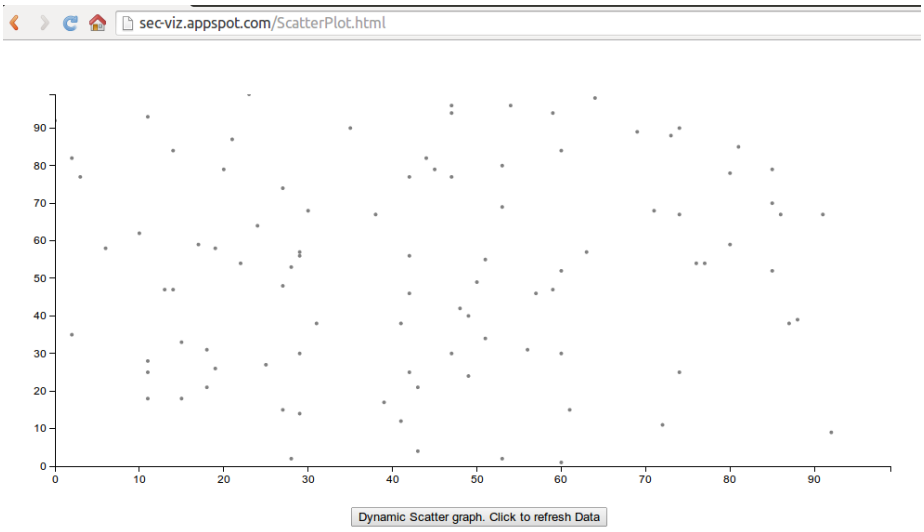

Figure 5.2: Scatter Plot

5.1.3 Geo-referenced Visualization

To check the abilities of D3 to plot the geographical data, we have created a visualization which shows the number of login violations for a company based on it's each state office in USA as shown in Figure 5.3 [3]. We have generated random login data and created a geo-plot for it. Color encoding scheme has been used to show the data values. The format of data is:

{State , Invalid Logins}

References for the code :

- `http://chimera.labs.oreilly.com/books/1230000000345/ch12.html#_json_meet_geojson`

5.1.4 Visualization Using Google Maps

We have also explored the capability of handling the Google Maps and plotting data on the google map. We have used randomly generated packet transfer data and plotted it using javascript and google maps apis. Source locations are shown as red markers, while destination locations with green. The moving arrows are used to show the data-transfer and connect the source and destination locations. The rate of packet transfer is encoded in the form of speed of movement of arrows as shown in Figure 5.4[4]. The format of a data is :

{sourceLAT, sourceLNG, destinationLAT, destinationLNG, trafficSize}

Following operations are provided over the data :

[3] http://sec-viz.appspot.com/GeoData.html
[4] http://sec-viz.appspot.com/GoogleMaps.html

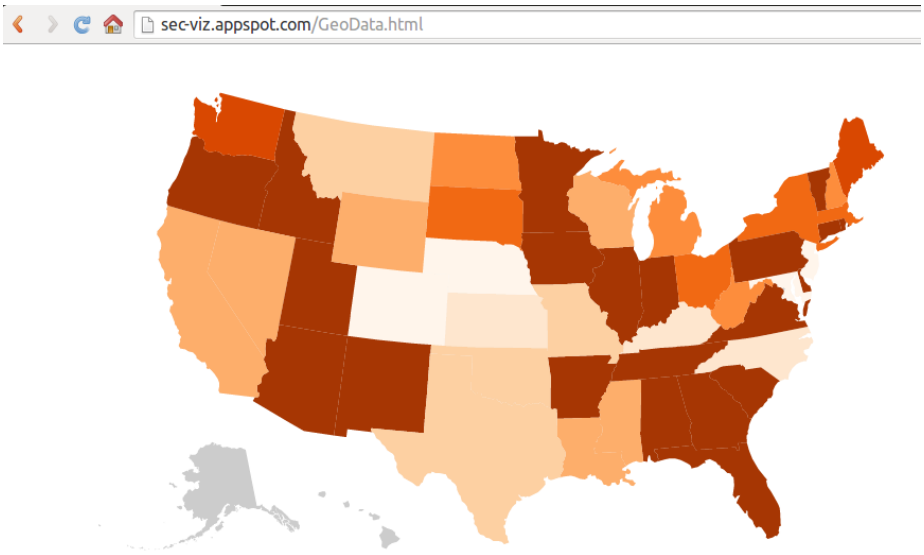

Figure 5.3: Geo data for login violations

- Zoom-in and Zoom-out of map
- Pan the map

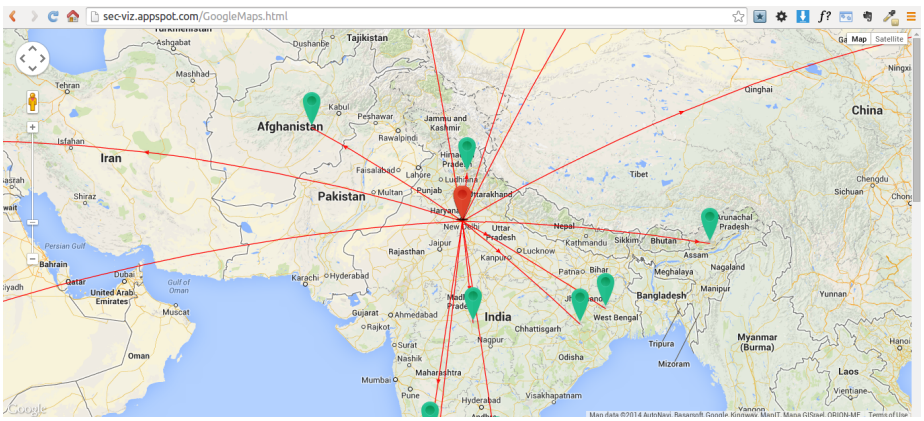

Figure 5.4: Google Maps showing packet data

References for the code :

- `https://developers.google.com/maps/documentation/javascript/`

5.1.5 Parallel Coordinates

Parallel Coordinates is an important visualization technique for multi-variant data. We have used open source wrapper called “Parallel Coordinates” version

0.2.2, written over D3.js to create this visualization. As shown in Figure 5.5[5], we have implemented parallel coordinates to visualize network data. The format of a network data entry is :

{Time, Source, Destination, Protocol, Length, DestPort}

Following operations are provided by this visualization :

- Reloading a new data set
- Re-arranging axis
- Protocol based color encoding
- Brushing and Linking
- Progressive rendering of data
- Select a particular set of axis

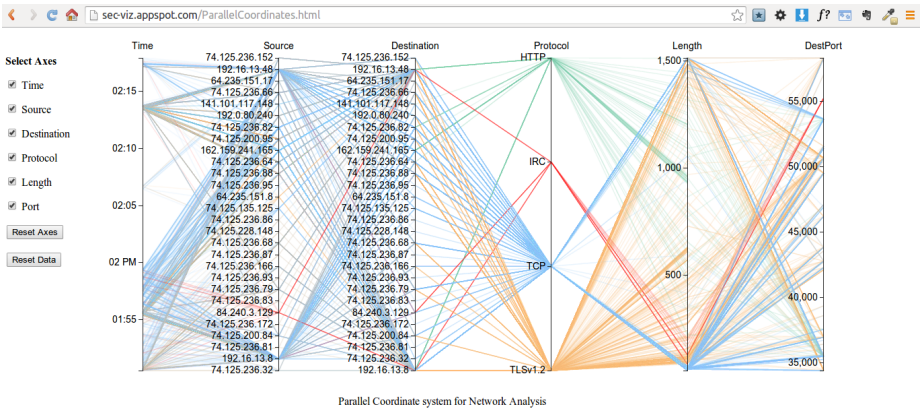

Figure 5.5: Parallel Coordinates showing network data.

References for the code :

- `http://syntagmatic.github.io/parallel-coordinates/`
- `http://bl.ocks.org/jasondavies/1341281`

5.2 Implementation on Real Data

We have used anonymized network data from RSA, implemented various visualization techniques on it, and then integrated the implementation with a mockup of "Investigator"'s source code. In the implementation phase, we have additionally derived hierarchical data from the given input data, and applied appropriate hierarchical data visualization techniques.

Overall, we have achieved a unified representation using linked visualizations on a given data set.

[5]http://sec-viz.appspot.com/ParallelCoordinates.html

5.2.1 Proposed Architecture

Basic architecture is as shown in Figure 5.6. EMC provided us with a mock up of their product Investigator Interface (further referred as investigator) which serves the data based on various queries. This involves querying the data from data base and then converting it to a suitable format in JSON. This JSON is then sent to the requesting browser where it gets rendered using different visualizations.

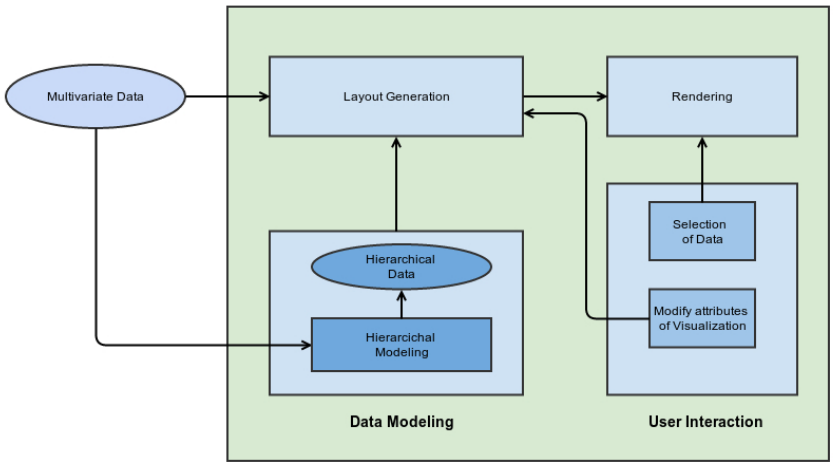

Figure 5.6: System architecture

5.2.2 Choice of Visualization Techniques

We have decided to use parallel coordinates to depict the multivariate nature of the security log data, which can be considered equivalent to multidimensional data.

Hierarchical Modelling: We have implemented hierarchical modeling of the data based on the data selection patterns, based on various fields, an analyst would undertake in his/her exploration of the data. The fields are the same as the variables/dimensions in the multidimensional nature of the data. We have presented one option for building hierarchy, there are several, and one of the popularly used one based on time (for time-varying data). We have decided to implement treemaps, Sunburst display, and radial plots, to depict the hierarchy in the data.

5.2.3 Visualization of Security Logs

In this section, we describe the implementation of the chosen visualization techniques for security logs. We also discuss the user interactive features that each of the visualization have been supported with.

Parallel Coordinates

Parallel Coordinates is an important visualization technique for multi-variant data. We have used open source wrapper called "Parallel Coordinates" version 0.2.2, written over D3.js to create this visualization. As shown in Figure 5.7, we have implemented parallel coordinate to visualize network data. We used meta data provided by EMC team to get the basic understanding of various headers (more than 25 fields) to plot them as parallel axes.
Following operations are provided by this visualization :

- Reloading a new data set
- Re-arranging axis
- Brushing and Linking
- Progressive rendering of data
- Sub-selection of data on an axis (Brushing of data).

In addition to the above mentioned functions, we have added the following features:

- Color encoding per axis based on unique values on that axis
- Sorting of data - IP addresses based
- Parallel Coordinates update in case a subset of overall data is selected
- Show/Hide labels to handle visual clustering in case of too many unique values in axes.
- Handling data types which have first value as NULL. For this we have iterated over the data set and found first non-null value in every field. Based on this value we are detecting the data type of the field. There were few fields in which for all the records were NULL.
- Handling axis with large number of unique values. Earlier it was checked in library to display upto 60 unique values only. This check was disabled

References for the code :

- `http://bl.ocks.org/jasondavies/1341281`
- `http://syntagmatic.github.io/parallel-coordinates/`

Treemap

Treemap is an important visualization technique for hierarchical data. D3.js provides a basic implementation of treemap which we have used. As shown in Figure 5.8, we have implemented treemap to visualize network data. In treemap we are doing 3-level grouping in which the default fields are chosen the same as that of parallel coordinates vertical axes.
Following operations are provided by this visualization :

- Basic grouping of sample data(provided in JSON format)

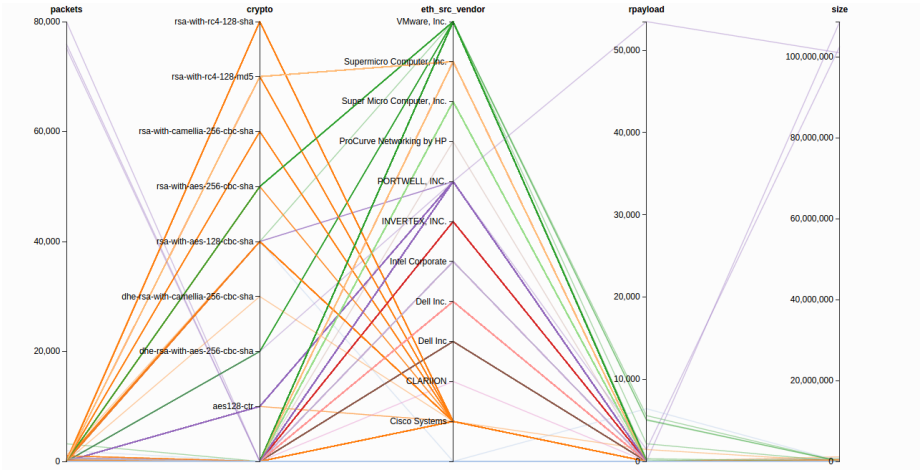

Figure 5.7: Parallel Coordinates showing network data.

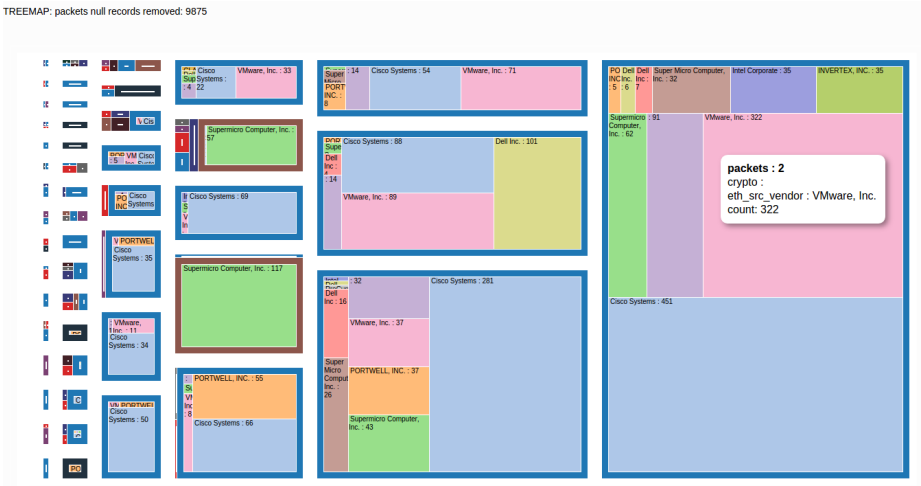

Figure 5.8: Treemap showing network data.

- Different colors to adjacent blocks
- Area based block size of a group

In addition to the above mentioned functions, we have added the following features:

- Structuring the data in hierarchical form (using rollup method)
- Improved visual segregation for different levels:
  - First level: We are separating first level groups with increased padding
  - Second level : To mark the second level of group, we use dark color border of a block. We have used customized color technique to color the same fields in different groups with a same colored dark border
  - Third level : To clearly separate third level groups we use light colors to fill the blocks. Here also we have used the customized color technique to color the same fields in different groups with a same color fill
- Default first three parallel coordinates axis are used for grouping.
- All the three groupings can be customized. An operator can select on which field these levels of grouping has to be done
- In Level 1, null records are not shown in visualization is indicated. We have intentionally removed null values from displaying in Level 1 grouping to save up the screen space. But removing them for second or third level was not feasible as it showcases some useful information to the operator
- In each block we display the number of records found for that field in the particular group.
- When a mouse hover is done over any block, a tooltip clearly shows the three different group fields associated with that data entry

References for the code :

- `https://github.com/mbostock/d3/wiki/Treemap-Layout/`

Sunburst Display

Sunburst display is one of the effective visualization techniques for hierarchical data. D3.js provides a basic implementation of sunburst which we have used. As shown in Figure 5.9, we have implemented sunburst to visualize network data. In sunburst also we are doing 3-level grouping in which the default fields are chosen the same as that of parallel coordinates vertical axes. The fields chosen for grouping in treemap are consistent with Sunburst too.
Following operations are provided by this visualization :

- Basic grouping of sample data(provided in JSON format)
- Different colors to adjacent blocks
- Area based block size of a group

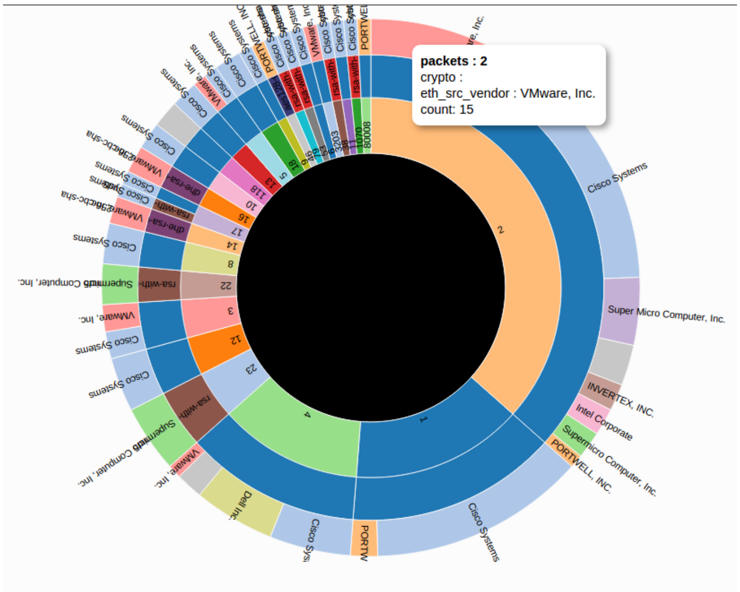

Figure 5.9: Sunburst display showing network data.

- Segregation of different levels as different radii concentric circles
- Ability to zoom in on any group by clicking on the area

In addition to the above mentioned functions, we have added the following features:

- Structuring the data in hierarchical form (using rollup method)
- Default first three parallel coordinates axis are used for grouping.
- All the three groupings can be customized. An operator can select on which field these levels of grouping has to be done
- In Level 1, we have not shown the null records in the visualization. We have intentionally removed null values from displaying in Level 1 grouping to save up the screen space. But removing them for second or third level was not feasible as it showcases some useful information to the operator.
- In each block we display the number of records found for that field in the particular group.
- When a mouse hover is done over any block, a tooltip clearly shows the three different group fields associated with that data entry.

References for the code :

- `http://bl.ocks.org/mbostock/4348373/`

Radial Plots

Radial plot is yet another effective visualization technique for hierarchical data. D3.js provides a basic implementation of radial which we have used. As shown in Figure 5.10, we have implemented radial to visualize network data. In radial also we are doing 3-level grouping in which the default fields are chosen the same as that of parallel coordinates vertical axes. The fields chosen for grouping in treemap are consistent with radial too.

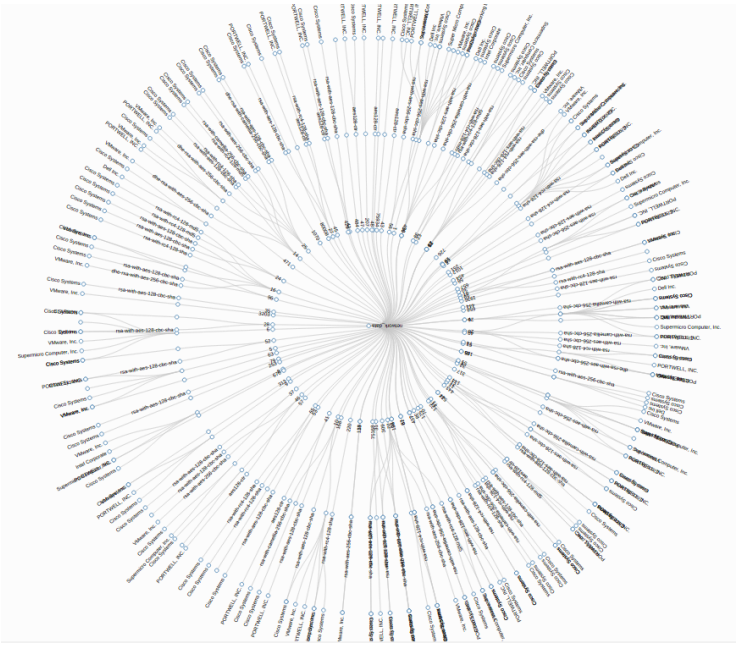

Figure 5.10: Radial plot showing network data.

Following operations are provided by this visualization :

- Basic grouping of sample data(provided in JSON format)
- Unique values are focussed rather than their frequency of occurrence
- Segregation of different levels as values on different radii concentric circles

In addition to the above mentioned functions, we have added the following features:

- Structuring the data in hierarchical form (using rollup method)
- Default first three parallel coordinates axis are used for grouping.
- All the three groupings can be customized. An operator can select on which field these levels of grouping has to be done

References for the code :

- `https://github.com/mbostock/d3/wiki/Tree-Layout`

5.2.4 Specifications for Execution

In this section, we will discuss the instructions for running our code.

Code Dependencies

To run the demonstration, following need to be installed :

1. Flask framework version 0.10.1 or above.
2. Python 2.x

Demo setup

1. Over the command line, change directory to 'db' folder.
2. On the terminal, execute the command :

   ```
   python invest.py
   ```

3. This will execute the python server with the demo hosted at location:

   ```
   http://127.0.0.1:5000/
   ```

   Paste this URL in any browser and press enter. Demonstration will begin.

5.3 Discussions

EMC^2-RSA provided us with a mockup of their product "Investigator". Investigator comprises of textual representation of the security log data, with minimal visualizations. To the best of our knowledge, EMC^2 has integrated a basic version of parallel coordinates in their next generation product called "Security Analytics".

The Investigator mockup provided to us transforms the log data from database to JSON format. This data set is an anonymized and structured version of the log data collected by EMC^2-RSA. There are 12,800 entries in the dataset. Keeping clear seperation between data and the visualizations using modularity by design, has allowed us to keep the code modular. Whle building the visualizations we have considered the implementations of the following user interactions as essential:

- Preserving the multidimensional nature of the data
- Handling updates of dataset
- Filtering of data (Brushing)
- Color/opacity encoding of dimensions
- Filtering of dimensions
- Integrating multiple visualizations and keeping them in sync.
- Handle large amount of data

- Should help find patterns
- Drilling down of the data set
- Good user experience (e.g. progressive rendering while loading of data)

In future we propose to focus on user stories to demonstrate the various capabilities of these visualizations. We also propose to have better linked views for various visualizations so that any interaction in one visualization updates other views as well. There is also a scope of undertaking user surveys to verify and measure the effectiveness of these visualizations.

As an improvisation, we plan to explore more visualization techniques that could help security analysts in analysing logs and identifying security threats. Several improvements that can be done in our framework are:

- Testing for very large datasets
- Improving aesthetics of visualizations to increase the efficiency of operator

Chapter 6

Conclusions

Security visualization is a fast growing field that attempts to solve the problems of computer security by in-cooperating visualization techniques in intrusion detection system. It will reduce the manual labour work for security ID analysts as well as help them in monitoring the ID system and detecting the security threats. Network data is a multivariate dataset and we found that parallel coordinates is a good choice for visualizing network-data. We have also discussed several visualization toolkits along with their pros and cons and decided to use D3 for our application. In future we plan to explore more visualization techniques that could be help security ID analysts to analysing ID system's output and identifying the security threats.

Several improvements that can be done in our framework are :

- Handling very large datasets
- Providing more flexibility in terms of the operations/filtering that can be done over the data sets
- Allowing detailed and abstract views of the data visualizations by allowing drilling down on a smaller data set
- Allowing data pivoting to change visualization among related datasets
- Composite Visualization

Bibliography

[1] Aansa Ali. Evaluation and comparison of alternate programming languages to javascript. In *Proceedings in Research Conference in Technical Disciplines*, number 1, 2013.

[2] N. G. Belmonte. JavaScript InfoVis Toolkit, 2011.

[3] Nicolas Garcia Belmonte. http://philogb.github.io/jit/.

[4] Casey Reas Ben Fry. http://processing.org/.

[5] Michelle A Borkin, Chelsea S Yeh, Madelaine Boyd, Peter Macko, Krzysztof Z Gajos, Margo Seltzer, and Hanspeter Pfister. Evaluation of filesystem provenance visualization tools. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2476–2485, 2013.

[6] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, December 2011.

[7] Mike Bostock. http://strongriley.github.io/d3/ex/sunburst.html.

[8] Madelaine D. Boyd. InProv: Visualizing Provenance Graphs with Radial Layoutsand Time-Based Hierarchical Grouping, 2012.

[9] Winnie Wing-Yi Chan. A survey on multivariate data visualization. *Department of Computer Science and Engineering. Hong Kong University of Science and Technology*, 8(6):1–29, 2006.

[10] Hyunsang Choi, Heejo Lee, and Hyogon Kim. Fast detection and visualization of network attacks on parallel coordinates. *computers & security*, 28(5):276–288, 2009.

[11] Susan B Davidson, Sarah Cohen Boulakia, Anat Eyal, Bertram Ludäscher, Timothy M McPhillips, Shawn Bowers, Manish Kumar Anand, and Juliana Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4):44–50, 2007.

[12] Lars H Fielder and Timothy J Dasey. Systems and methods for composable analytics. Technical report, DTIC Document, 2014.

[13] Olaf Hartig. Provenance information in the web of data. In *LDOW*, 2009.

[14] Jeffrey Heer, Stuart K. Card, and James Landay. Prefuse: A toolkit for interactive information visualization. In *ACM Human Factors in Computing Systems (CHI)*, pages 421–430, 2005.

[15] http://en.wikipedia.org/wiki/Intrusion_detection_system. Wikipedia, intrusion detection system.

[16] Waqas Javed and Niklas Elmqvist. Exploring the design space of composite visualization. In *Pacific Visualization Symposium (PacificVis), 2012 IEEE*, pages 1–8. IEEE, 2012.

[17] Daniel A Keim. Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8, 2002.

[18] Andy Kirk. http://www.visualisingdata.com/index.php/2013/09/essential-resources-programming-languages-toolkits-and-libraries/.

[19] Robert Kosara. http://eagereyes.org/techniques/parallel-coordinates.

[20] Martin I Krzywinski, Jacqueline E Schein, Inanc Birol, Joseph Connors, Randy Gascoyne, Doug Horsman, Steven J Jones, and Marco A Marra. Circos: An information aesthetic for comparative genomics. *Genome Research*, 2009.

[21] Christopher P Lee, Jason Trost, Nicholas Gibbs, Raheem Beyah, and John A Copeland. Visual firewall: real-time network security monitor. In *Visualization for Computer Security, 2005.(VizSEC 05). IEEE Workshop on*, pages 129–136. IEEE, 2005.

[22] Yan Liu. Multivariate data visualization: A review from the perception aspect. In Michael J. Smith and Gavriel Salvendy, editors, *Human Interface and the Management of Information. Interacting with Information - Symposium on Human Interface 2011, Held as Part of HCI International 2011, Orlando, FL, USA, July 9-14, 2011, Proceedings, Part I*, volume 6771 of *Lecture Notes in Computer Science*, pages 221–230. Springer, 2011.

[23] Florian Mansmann, Fabian Fischer, Daniel A Keim, and Stephen C North. Visual support for analyzing network traffic and intrusion detection events using treemap and graph representations. In *Proceedings of the Symposium on Computer Human Interaction for the Management of Information Technology*, page 3. ACM, 2009.

[24] Florian Mansmann, Timo Göbel, and William Cheswick. Visual analysis of complex firewall configurations. In *Proceedings of the Ninth International Symposium on Visualization for Cyber Security*, pages 1–8. ACM, 2012.

[25] David Martinez. Architecture for machine learning techniques to enable augmented cognition in the context of decision support systems. In *Foundations of Augmented Cognition. Advancing Human Performance and Decision-Making through Adaptive Systems*, pages 148–156. Springer, 2014.

[26] Claudio T Silva, Juliana Freire, and Steven P Callahan. Provenance for visualizations: Reproducibility and beyond. *Computing in Science & Engineering*, 9(5):82–89, 2007.

[27] Wang Chiew Tan et al. Provenance in databases: Past, current, and future. *IEEE Data Eng. Bull.*, 30(4):3–12, 2007.

[28] Matthew Ward, Georges Grinstein, and Daniel Keim. *Interactive data visualization: foundations, techniques, and applications.* AK Peters, Ltd., 2010.