Problem Statement

CS461 - Senior Software Engr Project

Instructor: D. Kevin McGrath

Instructor: Kristen Winters

Fall 2018

Bhavya Parikh, Davian Lukman, Eli Laudi, Matthew L. Jansen, and Ryan Sisco

October 18th, 2018

Abstract

In many fields, there is often an issue of data presentation. Data is not an easily consumable thing in its raw form. This paper will look at the challenge of presenting new data in a meaningful and unique way in the form of an application. We will describe the problem in depth, and the implications that it has in industry and academia. Then, we will propose a detailed solution to the issue using an application to visualize data in a 3D environment, which will map not only the objects within each file but their relationships as well. Lastly, we will define performance metrics to measure the applicability and accuracy of the application.

I. PROBLEM DESCRIPTION

One recurring challenge in a variety of fields is the ability to present and visualize data in new ways. This is especially the case for large data sets which can be difficult to visualize, let alone find relationships or patterns in. Humans are able to interpret data sets more accurately and completely when presented in the form of a complex visual image, compared to reading and sorting through several log files, attempting to find correlations and draw conclusions by hand.

Furthermore, given certain data-types such as IP address and their corresponding geo-locations, names and hashes of malware, and other various tags, it can be helpful to visualize these objects in a 3D environment. Without the ability to view data in this format, it can be difficult to see how objects within a data set interact with each other and define how inter networks of these objects might operate in a given context or environment.

This can have a negative impact in both industry and academia, as it prevents a wealth of knowledge from being brought forward to their respective customers and researchers. This pitfall can also have negative impacts in a time-constrained environment, such as when conducting network forensics to determine the source of unwanted malware within a network, and identifying it accordingly based off fingerprints.

II. PROPOSED SOLUTION

There are several open source solutions to display data in a 3D manner. However, these solutions have their own input type that a lot of data might not come prepackaged as. Our solution is to come up with a program which will consume CSV spreadsheet files, extract the data from them, and display that data in a friendly 3D manner. The initial starting point of this project is to handle data both geographically and temporally. For example, the tool will consume a CSV file that McAfee will provide us. It will contain data such as IP-Addresses, hashes of malware, geo-locations, tags, and time stamps. The tool will have to parse the data and translate it into 3D coordinates as well as mapping all associated data to that point.

The first part of this process is to consume the CSV file and parse the data in an efficient manner. There are solutions to do this in nearly every programming language, so the choice in languages comes down to a performance and preference basis, the top contenders being Java or Python. We would most likely translate the data into JSON format, allowing for hierarchical sorting with least-pertinent data nested the deepest. The principle of transforming the data is simple. The complexity of this part of the application would come from the ability for the application to consume CSV files that contain differing types of information. One CSV file might contain geographical data, while another may contain spatially related data; the application would be able to handle both. This is a feature that may not be part of version one, but could very well be implemented.

The second part of the process is the visualization itself. The data obtained from the first part of the process will be used and mapped onto a 3D space where the application will represent the data sets with points in space or on a globe. This data will be organized spatially and use colors to denote user-desired specifications. These specifications will be supplied in the CSV file from part one. The user will be able to explore the 3D data set using click and drag.

A. CSV Data Input

There are several open source solutions to display data in a 3D manner. However, these solutions have their own input type that a lot of data might not come prepackaged as. Our solution is to come up with a program which will consume CSV spreadsheet files, extract the data from them, and display that data in a friendly 3D manner.

Data should be input from a GUI, however, for raw functionality, it can be pulled in from a command line just for the developmental purpose. There are solutions to do this in nearly every programming language, so the choice in languages comes down to a performance and preference basis, the top contenders being Java or Python. We would most likely translate the data into JSON format, allowing for hierarchical sorting with least-pertinent data nested the deepest.

The principle of transforming the data is simple. The complexity of this part of the application would come from the ability of the application to consume CSV files that contain different types of information. One CSV file might contain geographical data, while another may contain spatially related data; the application would be able to handle both. This is a feature that may not be part of version one, but could very well be implemented.

Depending on user preference, a graphical interface will be developed to make it easier to choose files and settings. If preferred, it can also allow a console to input commands directly. This all depends on how its priority to the client.

B. Data Population

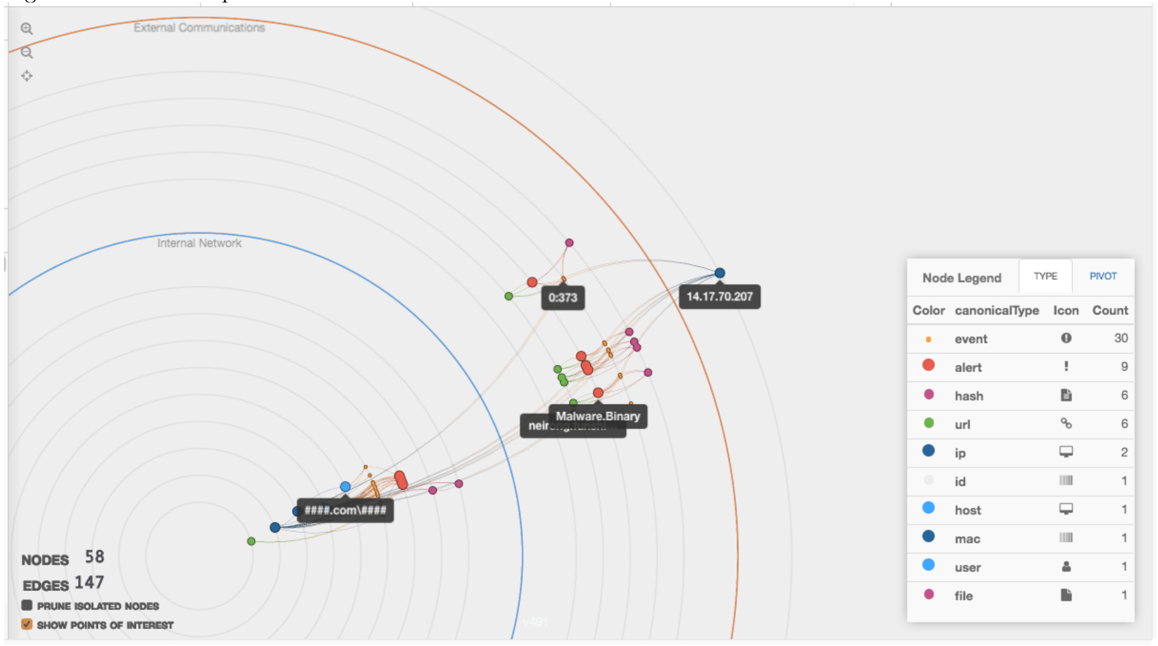
This will be the most challenging part of the project. There are many open source projects that create 3D models of the earth that are able to be populated. This will take a lot of time to get running in a stable way, that works with our given data. We would also like to add extra features such as color, different 3D models, better interface, etc. This would be developed later on after the base project is done.

C. 3D Rendering

After the project works with 1 3D model, we can include and create different visualizations. This is all ”filler” and can be added non-stop until the project is done. We will reach out and collaborate on renderings that are requested by everyone.

One of the most important parts of this assignment is differentiating the data. This project includes adding multiple data points such as dates and malware type. It would be a great addition to add an option to sort colored data by date or malware.

Fig. 1. Malware Graph



Node Legend TYPE PIVOT

Color	canonicalType	Icon	Count
●	event	❶	30
●	alert	!	9
●	hash	🗎	6
●	url	🔗	6
●	ip	🖵	2
●	id	▥	1
●	host	🖵	1
●	mac	▥	1
●	user	👤	1
●	file	🗎	1

This is a small example[1] of what is the end goal, as we are planning on a 3D model, but with the same pinpoints and lines shown. This should be flexible and will offer many different visualizations as time allows.

III. PERFORMANCE METRICS

In order to define how well our application performs and decide the completion of this project, performance metrics need to be created, measured, and fulfilled.

The first metric we will apply is the applications ability to correctly label and parse the input from the CSV file. Specifically, this measures how the application will split the CSV file into individual objects and identify the data type that is being read. This will be measured by testing the application with several different CSV files, containing IP addresses, geo-location, malware hashes, and names. The output of each test will determine if the application was able to parse and label each object in the CSV file correctly and completely.

The second metric we will test is how well the application is able to correlate an object's existence to another object or to find a specific relationship between two unique objects. Specifically, this will test how well the application can find relationships in a variety of data types or data sets and determine what kind of relationship it might be (existence based, communication, geo-location proximity, etc.).

The third metric we will test is the applications ability to form the parsed objects and correlations into a timeline. Specifically, this entails mapping the development of the data set into a set of snapshots, which they are compiled into a story of the evolution of data. Furthermore, this will test the applications ability to determine how many objects to include in a specified snapshot of the story, by analyzing the time intervals between each object's appearance and location within the given input. This metric will be tested by providing multiple CSV files with similar object properties, but differing time intervals and appearances within the file. The output of each test should correspond to the differences in time intervals within the CSV files.

The fourth metric will test the applications ability to develop a context or environment that is relevant to the parsed objects. Specifically, this entails including properties into the output such as output scope and geo-locations correctly in the form of a globe. Doing so has the benefit of identifying outliers and creating a better understanding for the user. Additionally, this includes correctly implementing a 3D view of the data.

References

[1] Graphistry, “Malware graph,” 2018, (Accessed on 10/18/2018. [Online]. Available: https://www.graphistry.com/graph-analytics