Enhancing Intrusion Analysis through Data Visualization

GIAC (GGIA) Gold Certification

Author: Wylie Shanks, giac@infosecmatters.com
Advisor: Angel Alonso Parrizas

Accepted: February 9, 2015

Abstract

This paper examines the role of data visualization in enhancing intrusion analysis. Methods for enhancing intrusion analysis are explored through examples and data visualization techniques. Open source tools contained on the DAVIX live CD, and the Elasticsearch, logstash, and kibana (ELK) stack are utilized. Data analysis improvements are achieved through the addition of Haka and Hakabana to the ELK stack.

1. Introduction

Increasingly, companies are required to sift through large volumes of relevant data in order to meet their governance, risk, compliance and security needs. Detecting and preventing insider threats, external attacks, and recovering from such activities requires analyzing data. Data visualization helps to facilitate understanding and make sense of the thousands of lines of data generated daily. We can quickly locate, identify and compare objects rather than having to sift through an overwhelming amount of raw data and information. Ultimately, the goal of data visualization is to produce graphs that are easy to understand and that support decision-making. This can be accomplished through the detection of outliers, misconfigurations, anomalies, and malicious activity by spotting trends and relationships amongst the data (Conti, G., 2007). Throughout this paper data sources, visualization processes and techniques, and tools are explored.

1.1. Changing Landscape

Cyber criminals target organizations of all sizes for their information and resources. These adversaries exploit organizational weaknesses such as its people, and its infrastructure in order to achieve their objective. For example, a cyber criminal could exploit or infiltrate a business' website or other Internet presence.

Traditionally, incident response practices focus on discovering attacks that have occurred or are in progress. Only through further analysis is the impact of a successful attack understood and steps taken to remediate discovered issues.

1.2. Existing methods

Traditional methods of intrusion analysis include collection and review of system and application logs. Some examples of each type of log are the firewall, intrusion detection/prevention system (IDS/IPS), proxy, authentication (e.g. Lightweight Directory Access Protocol (LDAP), and Remote Authentication Dial-In User Service), application, and system logs. This process can be labor intensive and time-consuming process. In order to identify the cause or source of the incident logs are reviewed for suspicious activity. This analysis may include:

- Firewall port scans (source of the scans), suspicious outbound connections
- Alerts from the IDS/IPS (e.g. malicious traffic)
- Proxy logs (e.g. blocked malware or malformed traffic)
- Abnormal user login time, failed / successful login, password brute-force attempts, account lockout etc.
- System reboot, stopping/starting processes, adding new programs to the startup function

These methods are sufficient when there are a manageable number of small incidents. One means of intrusion analysis is through data-driven security. The next segment of this paper examines data-driven security.

1.2.1. Data-driven security

One aim of most business' security practice is detecting known and previously unknown attacks. Defending systems depends on previous knowledge. For example, firewalls contain predefined rules and intrusion detection/prevention systems use signatures to detect known attacks and protocol anomalies. Data-driven security may be used to enhance the organization's defense capabilities.

Data-driven security begins with posing a question that has an objective answer. For example, "Has the organization seen this attacker's IP address before?" Next, iterations occur through collecting, cleaning, analyzing, refining the data. The original question is then updated as required. Data quality and process repeatability are very important in being able to resolve the question. The chart below outlines this process:

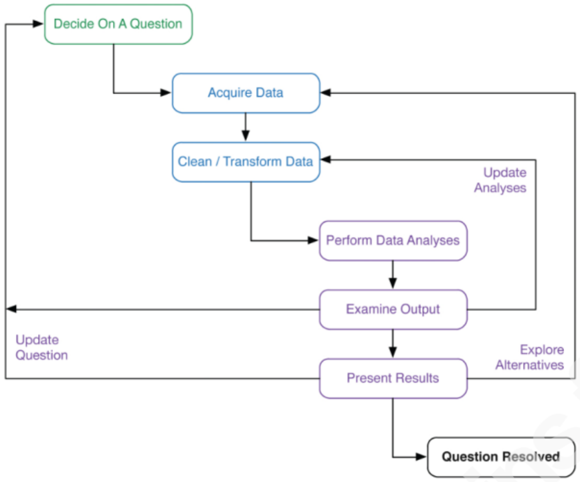


Figure 1. Data-driven security workflow. Reprinted from *Data-driven security analysis, visualization, and dashboards* (p. 605), by J. Jacobs & B. Rudis, 2014, Indianapolis, IN: John Wiley & Sons

In the previous example, logs are reviewed to determine whether the attack is coming from a known IP address. Each log format may need to be normalized, cleaned and/or transformed and then correlated. Once the process is complete the data is analyzed and the resulting output examined. Before completion, additional data analysis and refining of the question may be required.

This traditional approach has been used for many years and is somewhat scalable. However, it may not be best suited to addressing an organization's security needs. Thus, an enhanced approach to intrusion analysis and response is required to focus limited company resources more effectively against attacks. Fortunately, new methods have been developed to address this need. One new method to enhance protection against cyber criminals is through the information visualization process.

1.3. Steps of the Information Visualization Process

The information visualization process graphically displays data to aid interpretation and decision making by the reviewer. This process builds on the data-driven security approach by leveraging visualization as shown below:

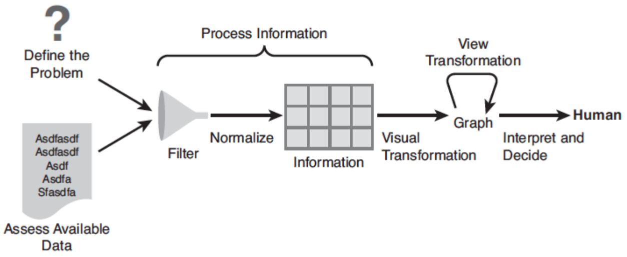


Figure 2. Information visualization process diagram. Reprinted from *Applied Security Visualization* (p. 120), by R. Marty, 2009, Upper Saddle River, NJ: Addison-Wesley.

1.3.1. Define the problem

A problem must be defined and understood in order to find a solution. All intrusion analysis starts with a defined problem. Common examples may include questions relating to how or why a breach occurred and what steps the attacker took. Those steps may include any data or information that was stolen. Ultimately, the question(s) should be answered by the graph generated by the information visualization process.

1.3.2. Assess available data

What data is available to address the problem or question posed? Is the appropriate data available for analysis? What additional data, if any, is required? Without the appropriate data it may not be possible to answer the question.

An understanding of the business, the system environment, and system configuration is advantageous so that only relevant data is included. For example, vendor default configuration may not meet the organization's data requirements, as their customer environment is unique and may not capture necessary data.

1.3.3. Process information

The next step involves processing available data. Data must be processed before it can become useful information. The information visualization process started with a question and data available to answer that question. Next, the data needs to be parsed or transformed into the format necessary so it can be graphed. Once completed, this information may need to be supplemented with additional gathered data. For example, logs may contain an IP address. Augmenting the IP address data with the Domain Name System (DNS) hostname and geographic location of that IP address may make the information more meaningful to the reviewer.

Note that only data relevant to the problem needs to be processed. Filtering of the data ensures that only representative data is used in the analysis. This approach focuses resources effectively and efficiently on the problem. Similarly, data may need to be aggregated in order to more effectively present the data visually. For example, rather than plotting numerous individual IP addresses or port numbers a summarized version may remove any duplication that exists in the data set while providing a visually cleaner view of the data.

1.3.4. Visual transformation

Once the data set is created the process of visually transforming the data can begin. First, the primary dimension (e.g. field or column of data) must be selected. Next, select any other required dimensions and choose whether or not it will be visible on the graph. The visible dimensions must have graph attributes defined such as color, size, and shape. Using multiple dimensions will have an impact on the type of graph used.

See Appendix A to help with selecting the right type of graph for the data to be visualized.

1.3.5. View transformation

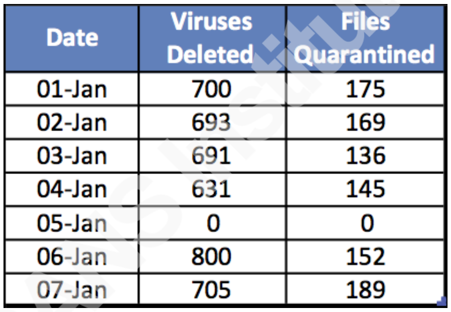
The data has been transformed into a visual representation. However, it might not be in the most desirable format for review or analysis purposes. There may be too many lines, or objects are the wrong size, shape or color. It may be necessary to iteratively change or aggregate the data, select another type of graph, or apply other filters until you have the desired output.

1.3.6. Interpret and decide

Although seemingly a rather straightforward step, interpreting and making decisions from the data is far from trivial. Intuitively, it is understood that the completed graph should contain the information necessary to interpret it and make subsequent decisions. However, it is necessary to return to the original question along with the output, to determine if the question has been answered and the problem has been solved. Comparing the results with another graph may helpful. This comparison may show a history or pattern of behavior that is easily detected when the pattern changes. For example, a daily graph showing detected viruses may have relevant information when the incidence of infection rises or falls sharply. In particular, a sharp decrease in detected incidents could indicate that anti-virus software is not working effectively. This failure may have allowed a virus to enter the company and spread undetected. Comparing the graphs may corroborate or confirm one's understanding of the new graph.

2. Data Visualization

Data visualization "…depends upon graphically presenting security-related data in ways that provide useful and actionable insight." (Conti, 2007, page xvii). For example, the table and chart below represents viruses deleted and files quarantined from January 1 through January 7. Visually, the data is represented in a manner that is more easily read and is understood more quickly than having to read the table. As a result, patterns and outliers are more easily detected.



Date	Viruses Deleted	Files Quarantined
01-Jan	700	175
02-Jan	693	169
03-Jan	691	136
04-Jan	631	145
05-Jan	0	0
06-Jan	800	152
07-Jan	705	189

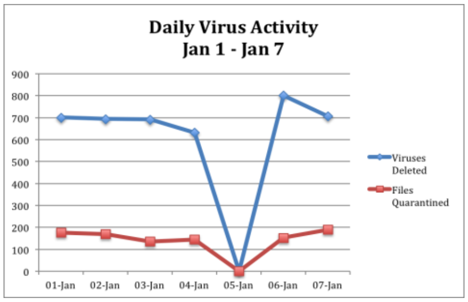


Figure 3 – Daily Virus Activity Jan 1 – Jan 7.

Information rich graphs act as a summarizing snapshot of voluminous data. Similarly, graphs should show the data, present many numbers in a small space, and make large datasets coherent rather than have the viewer think about how the graph was made (Tufte, 1983, page 13).

According to (Marty, 2009, pp. 5-6) the benefits to using data visualization is:

1. **It answers a question** – Rather than reading through a large dataset of textual information the graph displays this data more concisely allowing for a better understanding the relationship between the data.

2. **It poses new questions** – Additional questions may be raised due to unanticipated patterns that appear. For example, unexpected outliers may exist in the results or show communication between systems that warrant further investigation and analysis.
3. **It allows for exploration and discovery** – Using different or multiple types of graphs highlights properties in the data that was previously unknown. Utilizing interactive visualization tools helps the analyst to detect these patterns.
4. **It supports decisions** – Visualization allows for large datasets to be distilled into meaningful information.
5. **It communicates information** – Graphs are a more effective means of communicating information than via text. The old adage, "A picture is worth a thousand words" seems to accurately convey this concept.
6. **It increases efficiency** – Fewer people are required to sift through large datasets, spot patterns and trends, and analyze the results. The data visualization tool speeds the process by transforming the volume of data into a graphical representation. This is especially true as governance, regulatory, and compliance requirements increase the capture and storage of log data. Analyzing the application, host, and network logs to find relevant, actionable data requires the use of graphs to highlight areas of interest.

2.1. Data Sources

There are a variety of data sources (logs) that can be used in incident response. Some of the more common sources include:

- Network packet captures and traffic flows
- Firewall / IDS/IPS logs
- Host system / application logs (e.g. DNS, web or other server, database)
- Authentication system (e.g. LDAP, RADIUS)
- Proxy logs

Log and auditing settings may have to be modified if the existing logs do not contain the necessary data.

2.2. Visually representing data

Thus far the focus of this paper has been the process of transforming data. The next segment of this paper will explore the type of graph to employ as well as meaningfully representing the data.

It is important to select the most appropriate type of graph for what you are trying to understand (i.e. problem to solve). Appendix A provides a list of problem scenarios and suggests the most suitable type of graph to display that data.

A dataset may contain multiple dimensions (e.g. columns or fields) that are represented in the graph. An objective of data visualization is to show the relationship amongst the data. This objective can be achieved through application of the next segment.

2.2.1. Graph properties

Typically, graphs are generated and displayed as two-dimensional objects on paper or a computer screen. According to (Marty, 2009, pp 66-74) there are a number of graphical components that help to display the dimensions in a meaningful way.

2.2.1.1 Color

Color is used as an additional dimension in order to add meaning to a graph. Displaying a variety of bright colors may be preferred as meaning can be lost when colors are too similar. In such an instance, it is more difficult for the reviewer to differentiate between objects. Similarly, if too many colors are used the graph may become saturated and unreadable. One notable feature of using colors in a graph is it can be used to show a continuous range or spectrum. For example, a heat map is used to represent a range of values such as low, medium, and high through colors such as green, yellow and red.

2.2.1.2 Size, Shape, and Orientation

Data meaning can also be illustrated through the size and shape of an object, especially when compared to other objects on the graph. The relative relationship between the values can be more easily presented. In addition, the shape (e.g. circle, square, dot) can be used to differentiate data in the dataset.

Orientation (e.g. a directional arrow) may also be used to reference relevant information. For example, network traffic from a source IP address to a destination IP address is shown via an arrow pointing from the source IP address to the destination IP address.

2.2.1.3 Chart Axes

Chart axes are used to display the data point values. For example, the x-axis is typically used to provide context (i.e. data point values or a series scale starting at zero and rising proportionally) on the vertical axis. Whereas, the y-axis is used to provide context for the horizontal axis. The chart below shows Viruses detected and deleted from Jan 1 – Jan 7 using an x-axis of 0 through 900 and a y-axis of 01-Jan through 07-Jan.

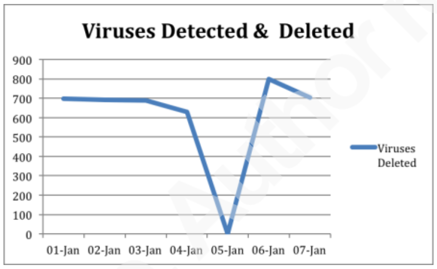


Figure 4 – Viruses Detected & Deleted – 01-Jan through 07-Jan

2.2.1.4 Simple charts

The following is a list of commonly used graphs used in data visualization:

- Pie Chart
- Bar Chart
- Line Chart
- Stacked Pie Chart
- Stacked Bar Chart
- Stacked Line Chart
- Histograms
- Box Plots
- Scatter Plots
- Parallel Coordinates

See Appendix A for further details and examples of these charts. The next segment of this paper focuses on visual security analysis.

2.3. Visual Security Analysis

This section of the paper describes analyzing security data using a visual approach. The three visualization approaches discussed by (Marty, 2009, p161) are:

- Reporting
- Historical Analysis
- Real-time reporting

2.3.1. Reporting

Graphs enhance reporting and are useful when summarizing data. Reporting is not a primary use of data visualization.

2.3.2. Historical analysis

Historical analysis utilizes time-series (e.g. timeline, or trends) or correlation graphs (i.e. relationship between data dimensions). In addition, using interactive analysis (e.g. interacting with the data and graphing function) or forensic analysis (e.g. data document the incident and/or discover attacks and their extent) enhances understanding of historical data.

2.3.3. Real-time reporting

Monitoring systems and applications in real-time provides reporting on events of interest. According to (Few, 2006, p34) a dashboard is a visual display of information used to meet objectives and displayed on a single screen. Dashboards are generally used to provide operational, tactical and strategic information. This may include statistics, status information and metrics (among other data). Several practical examples will be examined in the next segment.

2.4. Use-case (practical) examples

Several use-cases will be discussed in the following sections including binary file visualization; port scan visualization and firewall log visualization.

2.4.1. Binary File Visualization

A text file is opened in a text file viewer to display its contents for examination. Viewing the American Standard Code for Information Interchange (ASCII) values of a binary file displays non-readable data in addition to strings or text values that are easily readable. The use of binary file visualization becomes more compelling when it is used to determine if a file is encrypted. For example, encrypted data is necessarily highly random to avoid encryption-breaking ability through detection of repetitious patterns. Therefore, when graphed, the encrypted file contains a unique uniform visual unlike other files (see the histogram on figure 4 below). Similarly, encryption keys may easily be spotted in an otherwise unremarkable file, as that pattern is highly random. Therefore, incident analysis and triage processes can use binary file visualization to quickly determine important characteristics of the file. The following images were created using a free product called Binary Viewer by Proxoft (http://www.proxoft.com/binaryViewer.aspx).

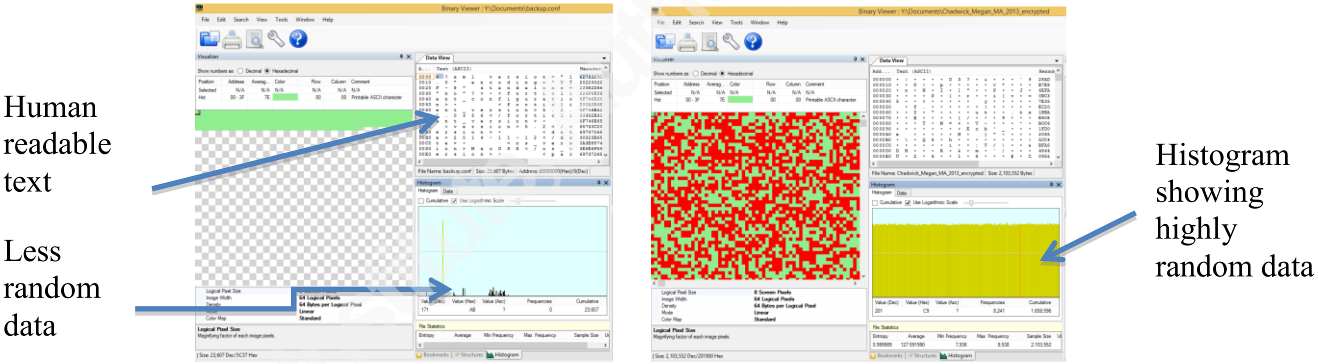
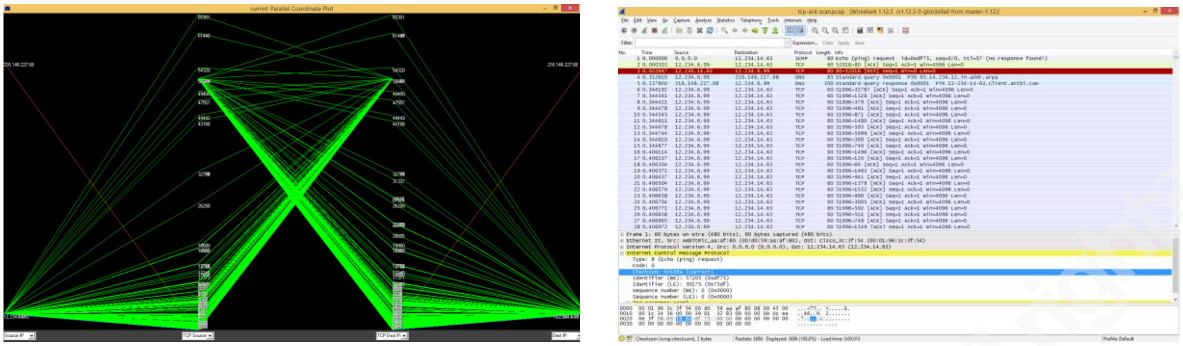

Human readable text
Less random data
Histogram showing highly random data
Binary Viewer : Y:\Documents\backup.conf
Binary Viewer : Y:\Documents\Chadwick_Megan_MA_2013_encrypted

Figure 5 – Sample Binary Viewer output showing text file and encrypted file.

2.4.2. Port scan visualization

Port scan are more easily visualized using parallel coordinates, which plot multidimensional data (multiple types of data) on a single graph. RUMINT (http://www.rumint.org/) was used to plot data from Laura Chappell's Wireshark 101 Essential Skills for Network Analysis book supplement (http://wiresharkbook.com/studyguide_supplements/9781893939943_traces.zip). The tcp-ack-scan.pcapng file was converted to pcap format in order to generate this graph. In this example, four dimensions were selected; source IP, TCP source port, TCP destination port, and destination IP. All traffic was included. Again, the graph is easier to read than the textual data (packet information in Wireshark).


rumint: Parallel Coordinate Plot
Source IP
TCP Source
TCP Dest P
Dest IP
tcp-ack-scan.pcap [Wireshark 1.12.3 (v1.12.3-0-gbb3e9a0 from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.  Time     Source           Destination      Protocol Length Info
  1 0.000000 0.0.0.0          12.234.14.63     ICMP       60 Echo (ping) request  id=0xdf75, seq=0/0, ttl=57 (no response found!)
  2 0.000303 12.234.6.99      12.234.14.63     TCP        60 52016→80 [ACK] Seq=1 Ack=1 Win=4096 Len=0
  3 0.022847 12.234.14.63     12.234.6.99      TCP        60 80→52016 [RST] Seq=1 Win=0 Len=0
  4 0.312810 12.234.6.99      216.148.227.68   DNS        85 Standard query 0x0001  PTR 63.14.234.12.in-addr.arpa
  5 0.337800 216.148.227.68   12.234.6.99      DNS       300 Standard query response 0x0001  PTR 12-234-14-63.client.attbi.com
  6 0.344192 12.234.6.99      12.234.14.63     TCP        60 51996→32787 [ACK] Seq=1 Ack=1 Win=4096 Len=0
  7 0.344341 12.234.6.99      12.234.14.63     TCP        60 51996→1528 [ACK] Seq=1 Ack=1 Win=4096 Len=0
  8 0.344411 12.234.6.99      12.234.14.63     TCP        60 51996→379 [ACK] Seq=1 Ack=1 Win=4096 Len=0
  9 0.344478 12.234.6.99      12.234.14.63     TCP        60 51996→461 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 10 0.344545 12.234.6.99      12.234.14.63     TCP        60 51996→871 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 11 0.344611 12.234.6.99      12.234.14.63     TCP        60 51996→1480 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 12 0.344678 12.234.6.99      12.234.14.63     TCP        60 51996→593 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 13 0.344744 12.234.6.99      12.234.14.63     TCP        60 51996→5999 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 14 0.344810 12.234.6.99      12.234.14.63     TCP        60 51996→300 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 15 0.344877 12.234.6.99      12.234.14.63     TCP        60 51996→740 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 16 0.406114 12.234.6.99      12.234.14.63     TCP        60 51996→1496 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 17 0.406237 12.234.6.99      12.234.14.63     TCP        60 51996→120 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 18 0.406304 12.234.6.99      12.234.14.63     TCP        60 51996→66 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 19 0.406371 12.234.6.99      12.234.14.63     TCP        60 51996→1463 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 20 0.406437 12.234.6.99      12.234.14.63     TCP        60 51996→941 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 21 0.406504 12.234.6.99      12.234.14.63     TCP        60 51996→1378 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 22 0.406570 12.234.6.99      12.234.14.63     TCP        60 51996→1552 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 23 0.406638 12.234.6.99      12.234.14.63     TCP        60 51996→900 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 24 0.406704 12.234.6.99      12.234.14.63     TCP        60 51996→3005 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 25 0.406771 12.234.6.99      12.234.14.63     TCP        60 51996→592 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 26 0.406838 12.234.6.99      12.234.14.63     TCP        60 51996→514 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 27 0.406905 12.234.6.99      12.234.14.63     TCP        60 51996→748 [ACK] Seq=1 Ack=1 Win=4096 Len=0
 28 0.406972 12.234.6.99      12.234.14.63     TCP        60 51996→1549 [ACK] Seq=1 Ack=1 Win=4096 Len=0

⊞ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
⊞ Ethernet II, Src: AmbitMic_aa:af:80 (00:d0:59:aa:af:80), Dst: Cisco_3c:3f:54 (00:01:96:3c:3f:54)
⊞ Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.0), Dst: 12.234.14.63 (12.234.14.63)
⊟ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x188a [correct]
    Identifier (BE): 57205 (0xdf75)
    Identifier (LE): 30175 (0x75df)
    Sequence number (BE): 0 (0x0000)
    Sequence number (LE): 0 (0x0000)

0000  00 01 96 3c 3f 54 00 d0  59 aa af 80 08 00 45 00   ...<?T.. Y.....E.
0010  00 1c 34 36 00 00 39 01  32 83 00 00 00 00 0c ea   ..46..9. 2.......
0020  0e 3f 08 00 18 8a df 75  00 00 00 00 00 00 00 00   .?.....u ........
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....

Checksum (icmp.checksum), 2 bytes | Packets: 3806 · Displayed: 3806 (100.0%) · Load time: 0:00.015 | Profile: Default
```

Figure 6 – A visual comparison between RUMINT and textual wireshark data.

Through filtering, it is possible to zoom in to see specific data. In this example, only TCP packets with a destination port between 1 and 80 are displayed.

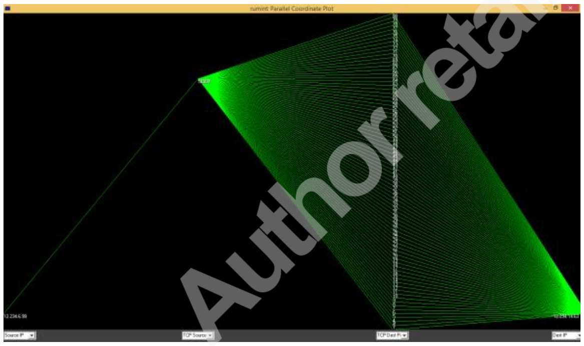


Figure 7 – RUMINT display of destination TCP ports 1 – 80.

In addition, various types of port scans can be visualized in order to be able to detect their unique pattern in the future. Here is the same scan filtered by TCP (all ports) with the following dimensions selected - source IP address, time to live (TTL), TCP source port, TCP destination port, destination IP address.

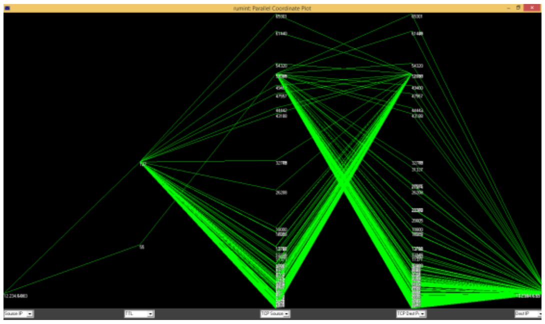
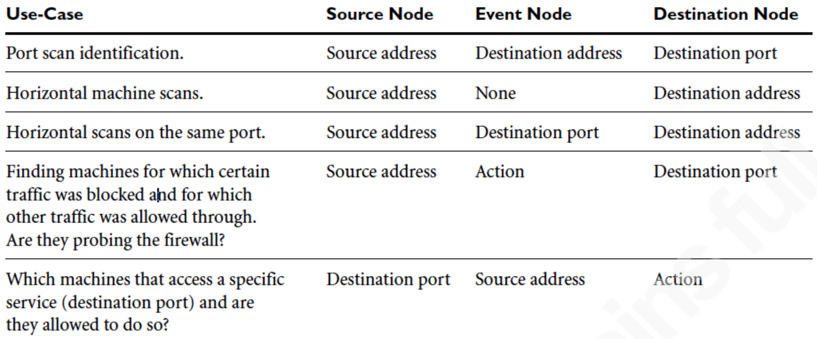


Figure 8 – RUMINT display of all TCP ports

2.4.3. Firewall log visualization

Firewall logs contain a vast amount of data. Searching the dataset for relevant data in the dataset can be a daunting task. Data visualization expedites this process. The source and destination IP address, source and destination port address, and protocol data may provide useful data when solving a specific problem. Consult the figure below for additional use-cases:



Use-Case	Source Node	Event Node	Destination Node
Port scan identification.	Source address	Destination address	Destination port
Horizontal machine scans.	Source address	None	Destination address
Horizontal scans on the same port.	Source address	Destination port	Destination address
Finding machines for which certain traffic was blocked and for which other traffic was allowed through. Are they probing the firewall?	Source address	Action	Destination port
Which machines that access a specific service (destination port) and are they allowed to do so?	Destination port	Source address	Action

Figure 9. Configurations for Firewall Log Files Identifying What Problems They Can Highlight. Reprinted from *Applied Security Visualization* (p. 135), by R. Marty, 2009, Upper Saddle River, NJ: Addison-Wesley.

The focus of this paper shifts to data visualization tools in next segment.

2.5. Data Visualization Tools

Several data visualization tools were used to generate graphs from a variety of data sources. The first tool, DAVIX, is an open source compilation of numerous freely available data visualization tools. Argus, ra, Afterglow and neato were used to create the first graph. The next set of open source tools were part of the Elasticsearch, Logstash, and Kibana (ELK) group of visualization tools. Haka and Hakabana are the final set of tools. They augment the ELK tools by providing visualizations for items such as real-time statistics, network bandwidth, DNS and HTTP traffic, and geolocation information.

2.5.1. DAVIX

Afterglow, written by Raffael Marty, provides a means of displaying this data visually using another tool such as GraphViz (neato). An optional configuration file is used to specify size, shape, and color of the dimensions (e.g. data to be visualized). The desired dimensions (e.g. source and destination address) were used to filter the dataset (downloaded and extracted from http://download.netresec.com/pcap/maccdc-2012/maccdc2012_00000.pcap.gz). The guidance for the following command was provided by (Marty, 2012):

```
argus -r maccdc2012_00000.pcap -w - | ra -r - -nn -s saddr daddr -c, | afterglow –c /opt/davix/afterglow/afterglow.properties -t | neato -Tgif -o maccdc2012_00000.pcap.gif
```

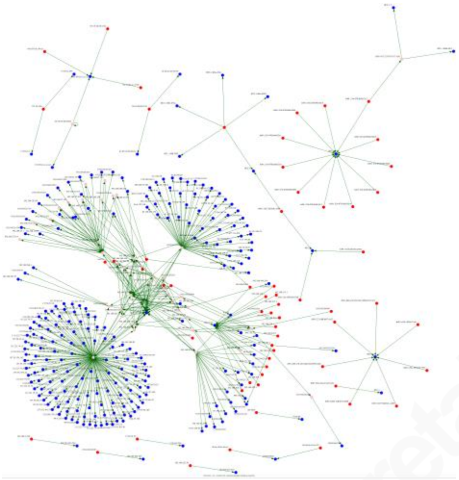


Figure 10 – Afterglow/Neato graph of network PCAP file.

This graph shows a variety of connection attempts and network scanning activity. The source address is a red circle, the event (line) is in green, and a blue circle denotes the destination. As noted above, several tools are required to generate this graph. Audit Record Generation and Utilization System (argus) transforms the network packet capture file data into network activity data for consumption in read argus (ra). The network activity data is filtered based on the specified criteria. In this example, the source and destination IP addresses are selected.

Afterglow creates a dataset to graphically display this data. The output file contains the IP address information, color, and label information. These settings can be changed within the color property file. Sample output from Afterglow is shown below:

```
"172.16.9.27" [fillcolor=red, xlabel="172.16.9.27"]

"192.168.202.77" [fillcolor=pink, xlabel="192.168.202.77"]

"192.168.214.1" [fillcolor=red, xlabel="192.168.214.1"]
```

See Appendix B for a sample Afterglow color property file that defines the source, event, and target colors among others. Alternatively, the sample.properties file installed with Afterglow can be copied and modified as required. Coloring can also be applied conditionally to describe specific data. For example, the color yellow will be used if the source IP address starts with 192.168 by using this command in the color property file: color.source="yellow" if ($fields[0]=~/^192\.168\..*/);

Neato was used to generate the graph in GIF format from the Afterglow data. All of the software required to generate this graph is included on the DAVIX Live CD (http://www.secviz.org/node/89).

2.5.2. ELK – Elasticsearch, logstash and kibana

The ELK stack is used to visually display search and real-time data via the interactive user interface and dashboard. Logstash is used to collect structured and unstructured data and filter, parse, index, and store it. Commonly used input for logstash is reading files and receiving syslog data. Next, data is filtered and/or parsed into structured fields and geoip data added (if configured). Finally, data is output to a file or to Elasticsearch. See Appendix C for installation instructions. The dashboard provides meaningful insight via customized panels.

Panels provide the results of a query in a visualized form such as a histogram, map (world map), table, term (pie chart, bar chart, table) or trend among others. Drilling down into the dataset is accomplished through filters.

The default ELK dashboard does not contain any data and must be populated by data received by logstash. In addition, knowledge of JSON is required in order to customize the dashboard. Fortunately, Haka and Hakabana provides commonly used dashboard panels to expedite the creation and use of a meaningful dashboard.

2.5.3. Haka and Hakabana

Haka and Hakabana contain useful pre-configured dashboard panels such as:

- Bandwidth monitoring
- Network traffic flow (including geolocation)
- DNS and HTTP traffic (including DNS queries, uri, user-agent and host)
- Real-time protocol statistics (e.g. tcp, ip, udp, icmp etc.)

Bandwidth monitoring provides a utilization trend graph. Selecting an area of the graph provides the details in the summary pane called documents. It is located in the bottom panel of the dashboard.

Network traffic flow is visualized using geolocation data to plot the country of orgin (or destination) on a map of the world.

DNS and HTTP traffic displays the count of DNS queries, uri, user-agent and hosts. Similarly, real-time protocol statistics show the percentage of traffic by type (e.g. tcp, ip, udp, icmp etc.).

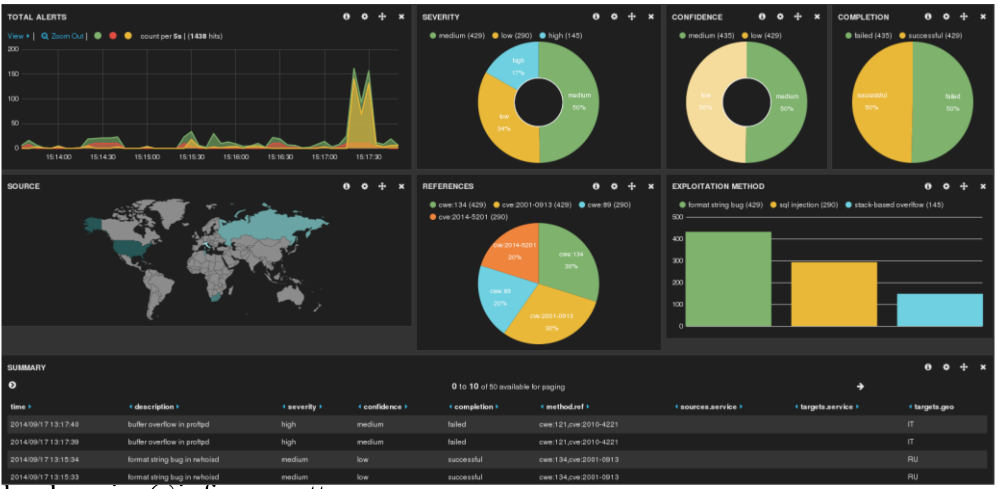
See Appendix C for Haka and Hakabana installation instructions.

Haka and Hakabana benefit from the modular and scalable nature of the ELK stack. As shown in figure 11 below additional panel modules can be configured that provide data such as exploit and incident event severity:

- Total Alerts
- Source (geo-location data)
- Exploitation method and references
- Incident severity and confidence

Configuring this functionality is explained via the link supplied in figure 11. Graphs from these panels provide a summarized view of triggered alerts and events including system or application exploits (e.g. Common Vulnerability and Exposures (CVEs)). In addition, event severity and confidence information provides additional information to reduce false positive detections and the resulting time wasted on remediation.

In the following graph, panels of the dashboard that support drill-down display additional details at the bottom of the dashboard in the summary section.


TOTAL ALERTS
View ▸ | Zoom Out | ● ● ● count per 5s | (1438 hits)
200
150
100
50
0
15:14:00
15:14:30
15:15:00
15:15:30
15:16:00
15:16:30
15:17:00
15:17:30
SEVERITY
● medium (429) ● low (290) ● high (145)
high 17%
medium 50%
low 34%
CONFIDENCE
● medium (435) ● low (429)
low 50%
medium 50%
COMPLETION
● failed (435) ● successful (429)
successful 50%
failed 50%
SOURCE
REFERENCES
● cwe:134 (429) ● cve:2001-0913 (429) ● cwe:89 (290) ● cve:2014-5201 (290)
cve:2014-5201 20%
cwe:134 30%
cwe:89 20%
cve:2001-0913 30%
EXPLOITATION METHOD
● format string bug (429) ● sql injection (290) ● stack-based overflow (145)
500
400
300
200
100
0
SUMMARY

0 to 10 of 50 available for paging

time	description	severity	confidence	completion	method.ref	sources.service	targets.service	targets.geo
2014/09/17 13:17:40	buffer overflow in proftpd	high	medium	failed	cwe:121,cve:2010-4221			IT
2014/09/17 13:17:39	buffer overflow in proftpd	high	medium	failed	cwe:121,cve:2010-4221			IT
2014/09/17 13:15:34	format string bug in rwhoisd	medium	low	successful	cwe:134,cve:2001-0913			RU
2014/09/17 13:15:33	format string bug in rwhoisd	medium	low	successful	cwe:134,cve:2001-0913			RU

Figure 11 – Elasticsearch with Kibana dashboard. Retrieved from http://www.haka-security.org/blog/2014/09/30/visualizing-alerts-using-kibana-and-elasticsearch.html on February 8, 2015.

Further customization of ELK/Haka/Hakabana is possible. A couple of documented examples of extended functionality include:

1. Detecting malicious payloads:

   http://www.haka-security.org/blog/2014/05/27/detecting-malicious-payloads-across-multiple-packets.html

2. Detecting Heartbleed with Haka:

   http://www.haka-security.org/blog/2014/04/25/detecting-heartbleed-with-haka.html

Haka and Hakabana provide a wealth of insight through visualized data and extensible, configurable, dashboard panels. In addition, the ability to drill down into the data is a useful feature that extends the meaningfulness of visualized data.

3. Conclusion

This paper examined the role of data visualization in enhancing intrusion analysis. A variety of data sources and the data visualization process were discussed. Of particular note were the real-world examples and how to apply data visualization techniques through open source tools. The DAVIX live CD provided a number of data visualization tools. However, the DAVIX tools reviewed in this paper did not allow for interactive (drill down) analysis. Interactive functionality in Elasticsearch, logstash, and kibana supported meaningful insights and analysis. Further data analysis improvements were achieved through the addition of Haka and Hakabana to the ELK stack. Additional dashboard panels supported protocol dissection and exploit reporting which dramatically enhanced intrusion analysis through data visualization.

References

Conti, G. (2007). *Security data visualization - graphical techniques for network analysis*. San Francisco: No Starch Press.

Few, S. (2006). *Information Dashboard Design - The Effective Visual Communication of Data*. Sebastopol, CA: O'Reilly.

Jacobs, J., & Rudis, B. (2014). *Data-driven security analysis, visualization, and dashboards*. Indianapolis, IN: John Wiley & Sons.

Marty, R. (2009). *Applied security visualization*. Upper Saddle River, NJ: Addison-Wesley.

Marty, R. (2012, March 21). Visualization Packet Captures For Fun and Profit. Retrieved February 8, 2015 from http://raffy.ch/blog/2012/03/21/visualizing-packet-captures-for-fun-and-profit/.

Tufte, E. (1983). *The visual display of quantitative information*. Cheshire, Connecticut: Graphics Press

Appendix A
Choosing the right graph for the data to be visualized

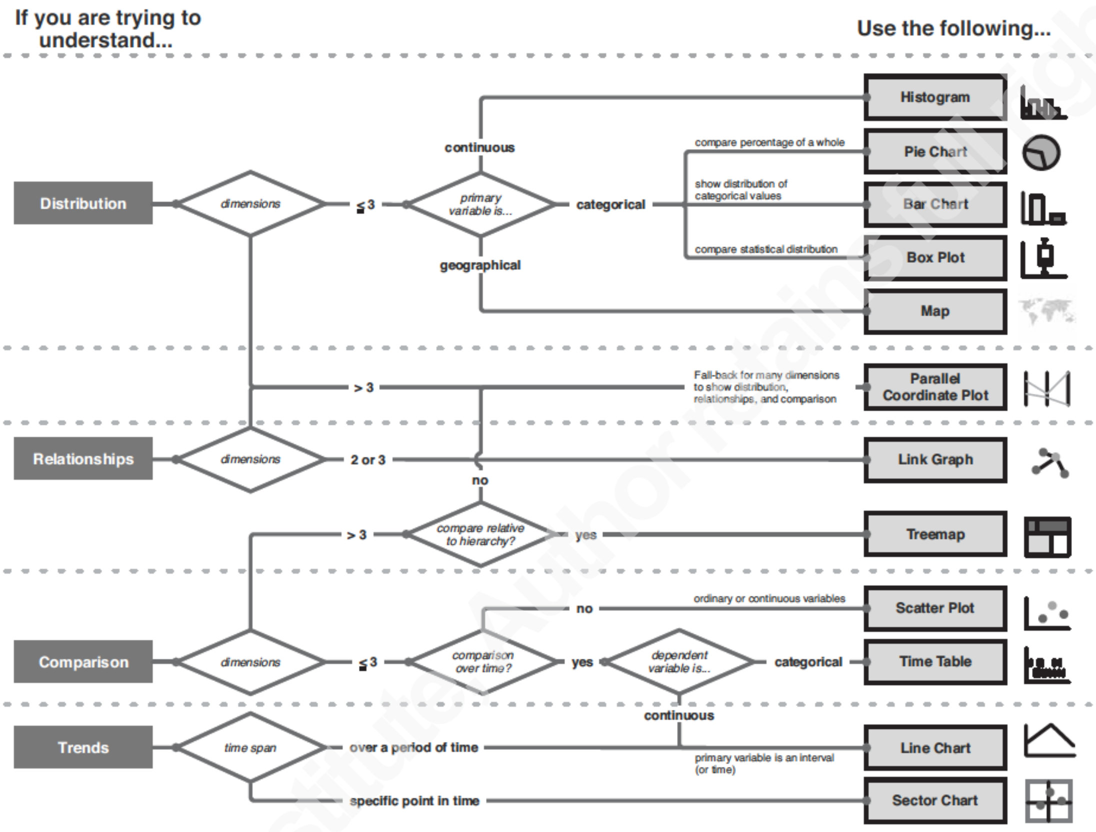


Figure 12. Choosing the right graph for the data to be visualized. Reprinted from *Applied Security Visualization* (p. 114), by R. Marty, 2009, Upper Saddle River, NJ: Addison-Wesley.

Appendix B
Sample Afterglow color property file

```
# AfterGlow Color Property File
#
#gdf=1;
#
# @fields is the array containing the parsed values
# color.source is the color for source nodes
# color.event is the color for event nodes
# color.target is the color for target nodes
#
# The first match wins
#
color.source="red"
color.event="green"
color.target="blue"
color.sourcetarget="pink"

color.edge="darkgreen" if (1)
size.edge=1;

# Changing node labels:
#label=substr(field(),0,10)

# URL for nodes (used for graphviz to enable image map functionality)
# This is an example of how to use AfterGlow with Splunk
url=http://localhost:8000/en-
        US/app/search/flashtimeline?q=%20\N%20starthoursago%3A%3A24

# Using node sizes:
```

```
#size.source=1;
#size.target=200
#maxNodeSize=0.2

# More complicated node size example:
#   This will size the source nodes and event nodes based
#   on the frequency of their occurrence. This is also a
# great example of how you can use internal AfterGlow variables
# in your configs.
#maxnodesize=1;
#size.source=$sourceCount{$sourceName};
#size.event=$eventCount{$eventName};
#size=0.5
#sum.source=0;
#shape.target=triangle
```

Appendix C
Installing the ELK stack and Haka / Hakabana

Instructions on how to install the ELK stack were retrieved from:
http://blog.dimaj.net/content/howto-view-and-analyze-your-logs-web-page

Information on how to install Haka / Hakabana were retrieved from:
http://www.haka-security.org/hakabana.html