Visualizations and Analysts

Christopher G. Healey, Lihua Hao and Steve E. Hutchinson

1 Introduction

The challenges of CSA discussed in previous chapters call for ways to provide assistance to analysts and decision-makers. In many fields, analyses of complex systems and activities benefit from visualization of data and analytical products. Analysts use images in order to engage their visual perception in identifying features in the data, and to apply the analysts. domain knowledge. One would expect the same to be true in the practice of cyber analysts as they try to form situational awareness of complex networks. Earlier, the Cognition and Technology chapter introduced the topic of visualization: its criticality to the users, e.g., cyber analysts, as well as its pitfalls and limitations. Now, this chapter takes a close look at visualization for Cyber Situational Awareness. We begin with a basic overview of scientific and information visualization, and of recent visualization systems for cyber situation awareness. Then, we outline a set of requirements, derived largely from discussions with expert cyber analysts, for a candidate visualization system.

We conclude with a case study of a web-based tool that supports our requirements through the use of charts as a core representation framework. A JavaScript charting library is extended to provide interface flexibility and correlation capabilities to the analysts as they explore different hypotheses about potential cyber attacks. We describe key elements of the design, explain how an analyst.s intent is used to generate different visualizations that provide situation assessment to improve the analyst.s situation awareness, and show how the system allows an analyst to quickly produce a sequence of visualizations to explore specific details about a potential attack as they arise.

Data visualization converts raw data into images that allow a viewer to "see" data values and the relationships they form. The motivation is that images allow viewers

Christopher G. Healey
Department of Computer Science, 890 Oval Drive #8206, North Carolina State University, Raleigh NC 27695-8206, e-mail: healey@ncsu.edu

to use their visual perception to identify features in the data, to manage ambiguity, and to apply domain knowledge in ways that would be difficult to do algorithmically.

Visualization has a long and rich history, starting with the use of maps and graphs to represent information. In one famous example, John Snow constructed a dot map to identify clusters of victims during a cholera outbreak in central London in 1854. Based on the location of the clusters, Snow hypothesized that contaminated drinking water was the cause of the disease. Disabling a public water pump in the area confirmed this conclusion. Another example occurred during the Crimean War (1853–1856). Florence Nightingale, volunteering as a nurse, observed very poor living conditions for wounded soldiers. This led her to create a multidimensional pie chart, known a Rose or a Coxcomb chart, to document the causes of deaths during the war. She used her charts to highlight that deaths from preventable disease far outnumbered deaths from injury and other non-preventable causes.

Work in visualization continued to expand on these earlier uses. In the area of statistics Jacques Bertin presented a theory of graphical symbols used to visually represent information [1]. Herman Chernoff proposed the use of facial expression properties (Chernoff faces) to visualize multivariate data [3]. In 1987 the National Science Foundation sponsored a Workshop on Visualization in Scientific Computing. Results were presented to the research community as a foundation for computer-based visualization [6]. The initial focus was on scientific visualization techniques for data with known spatial embeddings: volume visualization of reconstructed CT or MRI data, terrain visualization of geospatial data, or flow visualization of vector fields representing flow data. Later, the field expanded to include information visualization approaches for more abstract data: text visualization for documents or web pages, level-of-detail visualization for data with hierarchical structures, or multivariate visualizations made up of glyphs that vary their visual appearance to represent multi-valued datasets.

Although scientific and information visualization are seen as two sub-areas, significant overlap exists between them. For example, issues of human perception—how our visual system perceives basic properties of colour, texture, and motion, and how we can use this knowledge to build effective visual representations—apply to both areas. Multivariate data—data elements that encode multiple data attribute values—must be considered in both scientific and information visualization.

The field of visualization has matured significantly since the original NSF workshop. The area of visual analytics was proposed in 2005 to explicitly combine data analytics and visualization for iterative data exploration and hypothesis testing [19]. New research continues in many different directions. Christopher Johnson, director of the Scientific Computing and Imaging (SCI) Institute at the University of Utah, published a list of "Top Scientific Visualization Research Problems" [9]. Examples include integrating science into visualization, representing error and uncertainty, integrating perception into visualization, taking advantage of novel hardware, and improving human-computer interaction in visualization systems. A more recent report sponsored by the NIH and the NSF on visualization research challenges echoed these suggestions [10]. Although nearly 10 years have passed since Johnson's original list, many of these areas continue to generate new research results.

2 Formalizing Visualization Design

Numerous researchers have proposed ways to structure or describe a visualization design, for example, by the data being visualized, by the visual properties being used, or by the tasks the visualization supports.

We present a formalization that describes how data is mapped to visual properties like luminance, hue, size, orientation, and so on. Data passed through this data–feature mapping generates a visual representation—a visualization—that displays individual data values and the patterns they form.

An input dataset D is made up of one or more data attributes $A = \{A_1, \ldots, A_n\}$. Each data element e_i stored in D contains a value for each data attribute, $e_i = \{a_{i,1}, \ldots a_{i,n}\}$. In order to visualize D, a set of n visual features $V = \{V_1, \ldots, V_n\}$ are selected, one for every data attribute in A. Finally, mappings $M = \{M_1, \ldots, M_n\}$ are defined to map the domain of A_i to the range of V_i.

As a simple example, we return to the well-known technique of visualizing data on a map. Consider a temperature map similar to those shown on any weather site. Here, D is made up of three attribute: $A = \{A_1 : \text{longitude}, A_2 : \text{latitude}, A_3 : \text{temperature}\}$. The visual features $V = \{V_1 : x, V_2 : y, V_3 : \text{colour}\}$ are used to convert temperature readings throughout the world, stored as data elements $e_i \in D$, into a visual representation. M_1 and M_2 map e_i's longitude and latitude to an absolute x and y location. These can be used to position the map within the visualization window, and to change its size and aspect ratio. M_3 maps temperature values to different colours, often over a discretized rainbow colour scale that mirrors the range of colours seen in a rainbow: violet–indigo–blue–green–yellow–orange–red. This represents cold temperatures with purple and blue, hot temperatures with orange and red, and moderate temperatures with green and yellow, exactly as seen in many temperature maps.

More complicated datasets have more data attributes. For example, suppose we expanded the weather dataset to include not only temperature, but also pressure, humidity, radiation, and precipitation. This requires a visualization design that uses more visual features and data–feature mappings. It quickly becomes difficulty to choose features and mappings in ways that work effectively together. We could also increase the number of weather readings we collect. Even on a high definition display, once the number of data elements exceeds 2.2 million, there are not enough pixels in the display to visualize each element. Introducing additional positional attributes like elevation and time further complicates the visualization's design requirements. Non-numeric data attributes may also exist. Suppose D included an attribute "A_4 : forecast" that provides a text description of the current weather conditions, in the context of average and extreme conditions for the given location and time of year. Choosing a visual feature and a mapping to convert text forecasts into visual representations is itself a challenging problem. Researchers in visualization are studying new techniques that are designed to address exactly these types of issues.

Based on this overview, it seems clear that visualization offers the potential for important contributions to cyber situation awareness. Indeed, many existing situation awareness tools use visualization techniques like charts, maps, and flow diagrams to present information to an analyst. It is critical, however, to study how best to integrate visualization techniques into a cyber situation awareness domain. For example, which techniques are best suited to the data and tasks common to this domain? What is the best way to integrate these techniques into an analyst's existing workflow and mental models? How can problems in cyber situation awareness motivate new and novel research in visualization?

3 Visualization for Cyber Situation Awareness

The visualization community has focused recent attention on the areas of cyber security and cyber situation awareness. Early visual analysis of cyber security data often relied on text-based approaches that present data in text tables or lists. Unfortunately, these approaches do not scale well, and they cannot fully represent important patterns and relationships in complex network or security data. Follow-on work applied more sophisticated visualization approaches like node-link graphs, parallel coordinates, and treemaps to highlight different security properties, patterns in network traffic, and hierarchical data relationships. Because the amount of data generated can be overwhelming, many tools adopt a well-known information visualization approach: overview, zoom and filter, and details on demand. This approach starts by presenting an overview of the data. This allows an analyst to filter and zoom to focus on a subset of the data, then request additional details about the subset as needed. Current security visualization systems often consist of multiple visualizations, each designed to investigate different aspects of a system's security state from different perspectives and at different levels of detail.

3.1 Security Visualization Surveys

Visualization for cyber environments has matured to a point where survey papers on the area are available. These papers provide useful overviews, and also propose ways to organize or categorize techniques along different dimensions.

Shiravi et al presented a survey of visualization techniques for network security [18]. In addition to providing a useful overview of current visualization systems, they define a number of broad categories for data sources and visualization techniques. One axis subdivides techniques by data source: network traces, security events, user and asset context (e.g., vulnerability scans or identity management), network activity, network events, and logs. A second axis considers use cases: host/server monitoring, internal/external monitoring, port activity, attack patterns, and routing behaviour. Numerous techniques are described as examples of different data sources and use cases. The authors specifically address the issue of situation awareness in their future work, noting that many visualization systems try to prioritize important situations and project critical events as ways to summarize the massive amounts of data generated within a network. They distinguish between situation awareness, which they define as "a state of knowledge", and situation assessment, defined as "the process of attaining situation awareness." Converting raw data into visual forms is one method of situation assessment, meant to present information to an analyst to enhance their situation awareness.

Dang and Dang also surveyed security visualization techniques, focusing on web-based environments [5]. Dang chose to classify systems based on where they run: client-side, server-side, or web application. Client-side systems are normally simple, focusing on defending web users from attacks like phising. Server-side visualizations are designed for system administrators or cyber security analysts with an assumed level of technical knowledge. These visualizations are usually larger and more complex, focusing on multivariate displays that present multiple properties of a network to the analyst. Most network security visualization tools fall into the server-side category. A final class of system is security for web applications. This is a complicated problem, since it can involve web developers, administrators, security analysts, and end users. Dang also subdivided server-side visualizations by main goal: network management, monitoring, analysis, and intrusion detection; by visualization algorithm: pixel, chart, graph, and 3D; and by data source: network packet, NetFlows, and application-generated data. Various techniques exist at the intersection of each category.

New security and cyber situation visualization systems are constantly being proposed. We present a number of recent techniques, subdivided by visualization type. This offers an introduction to different visualization methods, in the context of the security and situation awareness domains.

3.2 Charts and Maps

As discussed in the overview, charts and maps are two of the most common visualization techniques. Well-known approaches improve a tool's accessibility by reducing the effort needed for analysts to "learn" the visualizations. It is common to present summarizes of data as bar charts, pie charts, scatterplots, or maps. Abstract data like network traffic or intrusion alerts need to be spatially positioned as part of the visualization design. Embedding the data using a chart's axes—for example, a scatterplot that maps IP addresses, $A = \{A_1 : \text{source IP}, A_2 : \text{destination IP}\}$, to the horizontal and vertical axes, $V = \{V_1 : x, V_2 : y\}$—or assigning a geographic location to each data element—for example, estimating an IP address's longitude and latitude, then converting $A = \{A_1 : \text{longitude}, A_2 : \text{latitude}\}$ to $V = \{V_1 : x, V_2 : y\}$—are common approaches to positioning data elements.

Roberts et al proposed the StatVis system, built on stacked bar graphs and geographic heatmaps, to visualize network health over time [17]. The graphs and heatmaps are used to present overviews of machine status in different geographic regions. A separate reticle visualization is used to present details about individual machines. The result is a combination of overview and details-on-demand for obtaining real-time situation awareness of a computer network's status.

A similar system called VIAssist visualizes network security data by linking between different charts to present the data from multiple perspectives [7]. When data elements are selected (or brushed) in one visualization, the same elements are highlighted (or linked) in the others. This identifies how data elements correlate between the visualizations. An overview uses bar and pie charts to visualize the most frequent elements for any data attribute. Coordinated views use various charts and maps present visualizations that are correlated with one another. This allows analysts to assess different parts of a computer system using different visualization techniques.

3.3 Node-Link Graphs

Another common visualization technique is a node-link graph, where nodes and links correspond to data elements and relationships between the elements. For example, nodes can represent machine clusters and edges network connections between the clusters. Node-link graphs also support the application of graph algorithms to analyze the structure of a network, or the pattern of traffic within the network.

The NetFlow Visualizer uses node-link graphs to display communication as oriented edges between network devices, represented by graph nodes, at different levels of aggregation [15]. This allows analysts to build up a situation awareness of the ongoing state of their networks, and to focus on individual flows of interest. The graph visualization is correlated with a spreadsheet containing specific values for individual network properties. Analysts can assign different attributes to control the size and color of the nodes and edges.

3.4 Timelines

Since changes over time are often critical to understanding a dataset, timelines are another common method of visualization for cyber situation awareness. Although timelines are similar to charts—for example, a line chart with $A = \{A_1 : \text{time}, A_2 : \text{frequency}\}$ mapped to $V = \{V_1 : x, V_2 : y\}$—their specific function is to highlight temporal patterns and relationships in a dataset.

Isis, a system designed by Phan et. al., provides two visualizations—timelines and event plots—that are linked together to support iterative investigation of network intrusions [16]. Isis's timeline presents an overview of temporal sequences of network flows in a histogram chart. The event plot allows an analyst to drill down over a subset of data to reveal patterns in individual events, using a scatterplot with $A = \{A_1 : \text{time}, A_2 : \text{IP address}\}$ mapped to $V = \{V_1 : x, V_2 : y\}$. Markers in the scat-

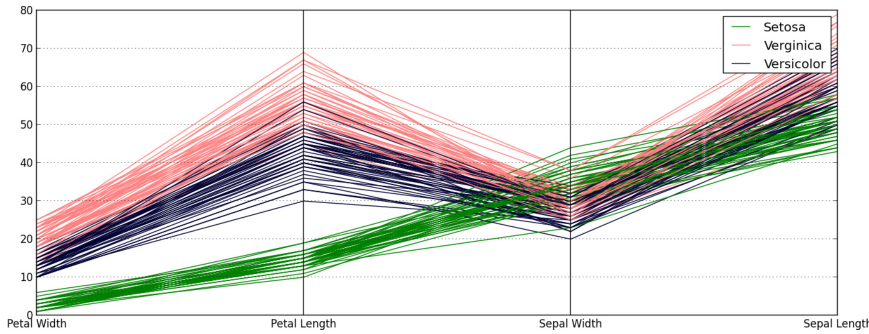


Fig. 1: A parallel coordinates visualization of data from Fisher's Iris dataset

terplot represent individual NetFlows. The markers can vary their shape, size, and colour to visualize NetFlow properties.

PortVis also uses a timeline to visualize port-based security events [14]. The timeline is ideal for summarizing events over a wide time window (e.g., hundreds of hours). It uses a scatterplot to visualize $A = \{A_1 : \text{port number}, A_2 : \text{time}\}$ mapped to the horizontal and vertical axes, $V = \{V_1 : x, V_2 : y\}$. More detailed visualizations are also available for situation assessment over shorter time windows: a main visualization to investigate activity on individual ports, and a detailed visualization that uses a bar chart to present values for different port attributes.

3.5 Parallel Coordinates

Parallel coordinates (PCs) are a technique for visualizing multivariate data using a set of n vertical axes, one per data attribute in a dataset. Each axis covers the domain of its attribute, from the smallest value at the bottom to the largest value at the top. A data element is represented as positions on each axis defined by the element's attribute values. Positions on neighbouring axes are connected with line segments, visualizing the element as a polyline. An important advantage of PCs is their flexibility in the number and type of data attributes they can represent.

As an example, consider Fisher's Iris dataset, which contains 450 measurements for three species of irises, $A = \{A_1 : \text{type}, A_2 : \text{petal width}, A_3 : \text{petal length}, A_4 : \text{sepal width}, A_5 : \text{sepal length}\}$. Plotting the data using $V = \{V_1 : \text{colour}, V_{2-5} : \text{PC axis}\}$ produces the visualization in Fig. 1. Numerous relationships are visible, for example petal length and width are correlated across all three species, as are sepal length and width. Virginica and versicolor irises have similar length and width patterns, but virginica irises have larger petals. Both species have petals that are about the same size as their sepals. Setosa irises, on the other hand, have sepals that are larger than their petals.

Parallel coordinates are used in PicViz to visualize network data [20]. PCs are useful for this type of data, since they can accommodate numbers, times, strings, enumerations, IP addresses, and so on. PicViz is built to investigate correlations across multiple properties of Snort log data. The effectiveness and readability of PCs is often influenced by the ordering of its axes. PicViz supports rapid axis reorganization to determine the best axis sequence for a given investigation. As in Fig. 1, colour is overlaid on an element's polyline to represent an additional user-chosen attribute. This allows for a closer examination of properties of anomalies as they are identified.

A second system, Sol, also uses parallel coordinates, but with horizontal axes [2]. Sol's Flow Capacitor visualizes NetFlows between two PC axes that represent a flow's source at the top and its destination at the bottom. Common data attributes assigned to the source and destination axes include IP address or geographic location. Small "darts" are shown flowing from the source plane to the destination plane, to visualize the amount of NetFlow activity over a user-defined time window. Users can also insert intermediate axes to visualize additional data properties. This causes the NetFlow darts to pass through multiple states, one per additional data attribute, on their way from source to destination.

3.6 Treemaps

Data with hierarchical structures can be visualized as a treemap, a visualization that recursively subdivides rectangular regions based on the frequency of different data attribute values. Intuitively, a treemap visualizes a multi-level tree as a 2D "map" by embedding leaf nodes within their parent node region.

Consider Fig. 2, which subdivides a dataset with $A = \{A_1 : \text{region}, A_2 : \text{parent}, A_3 : \text{revenue}, A_4 : \text{profit}\}$. The parent Global rectangle is subdivided into three continents: North America, Europe, and Africa, with the size of each continent subregion defined by its global profit. The continents are further divided by states and countries, producing a hierarchical decomposition by geography. Visually, $V = \{V_1 : \text{text label}, V_2 : \text{spatial location}, V_3 : \text{size}, V_4 : \text{chromaticity}\}$. The size of each subregion visualizes its revenue, and its combined colour and saturation—or chromaticity—visualize its profit using a double-ended colour scale: green for positive, red for negative, and a stronger hue for more extreme values.

Kan et al use treemaps in NetVis, a tool for monitoring network security [11]. A network within a company is subdivided, first by department, and then by host within department. Colour is used to identify whether a host has experienced Snort alerts during an analyst-defined time window. For nodes with alerts, brightness visualizes the number of alerts: brighter for more alerts.

Mansmann et al compared treemaps to node-link graphs for network monitoring and intrusion detection [13]. Mansmann's treemap consists of a set of analyst-selected local hosts currently under attack, organized hierarchically by IP address. Attacking hosts are arrayed around the boundary of the treemap, with links from at-

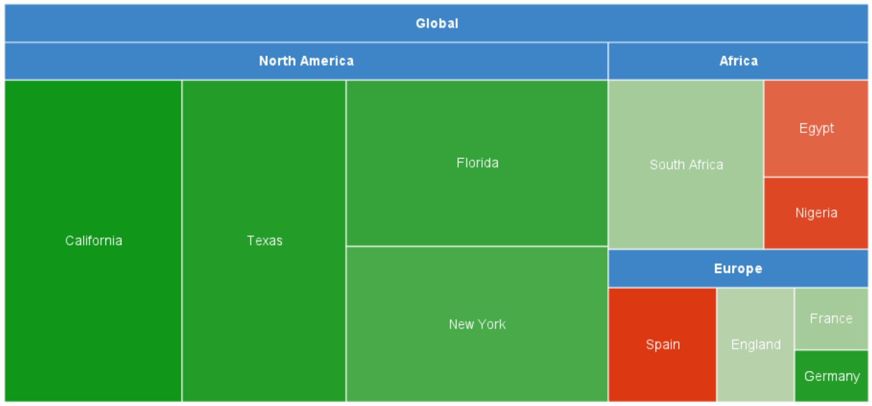


Fig. 2: A treemap of revenue for states and counties in three continents

tacker to target identifying where attacks are occurring. The size and colour of each treemap node can be assigned to different variables, including flow counts, packet counts, or bytes transferred. The authors identified numerous advantages and disadvantages of treemaps versus standard node-link graphs, for example, treemaps better emphasize the relationships between external attackers and internal hosts by subnet structure, but edges between attackers and local hosts can obscure the treemap nodes, hiding the information they visualize.

3.7 Hierarchical Visualization

In many cases a dataset can be structured into multiple levels of detail. For example, IPv4 addresses can be represented as an overview by aggregating individual addresses based on their network identifier, and within this as a detail view by including the host identifier. Other common examples include summaries by geographic location, by type of attack, and so on. The ability to visualize this type of contextual structure is useful for a number of reasons. First, overviews can be used to present an intuitive summary of a dataset. Second, a hierarchical visualization helps an analyst to choose subsets of the data that are logically related based on the hierarchy. This fits well with the approach in information visualization of overview, zoom and filter, then details on demand.

Information visualization has studied two related approaches to visualizing large hierarchies: overview+detail and focus+context [4]. Overview+detail combines an overview of a dataset with internal details. Treemaps are an example of this technique. Focus+context presents an overview of a dataset—the context—and allows a user to request additional details at specific locations—the focus. A well-known focus+context approach initially presents an overview of a dataset, then allows an analyst to position a zoom lens within the overview. Data underneath the lens is "zoomed in" to show internal detail.

NVisionIP visualizes NetFlow data at three different levels of detail: an overview of the network state, summaries of groups of suspicious machines, and details on an individual machine [12]. A galaxy view provides an overview of the current network state as a scatterplot, with $A = \{A_1 : \text{subnet}, A_2 : \text{host}\}$ mapped to $V = \{V_1 : x, V_2 : y\}$. Each marker in the scatterplot identifies an IP address participating in a flow. Analysts can select a set of hosts that show signs of abnormal traffic patterns and zoom in to compare traffic across hosts with two histograms: one representing traffic on a number of well-known ports, and the other representing traffic on all other ports. Finally, a detailed machine view visualizes a machine's byte and flow counts for different protocols and different ports for all network traffic over an analyst-chosen time window.

Although these security visualization systems aim to support more flexible user interactivity and to correlate various data sources, many of them still force an analyst to choose from a fairly limited set of static representations. For example, Phan et al use charts, but with fixed attributes on the x and y-axes. General purpose commercial visualization systems like Tableau or ArcSight offer a more flexible collection of visualizations, but they do not include visualization and human perception guidelines, however, so representing data effectively requires visualization expertise on the part of the analyst. Finally, many systems lack a scalable data management architecture. This means an entire dataset must be loaded into memory prior to filtering, projecting, and visualizing, increasing data transfer cost and limiting dataset size.

4 Visualization Design Philosophy

Our design philosophy is based on discussions with cyber security analysts at various research institutions and government agencies. The analysts overwhelming agreed that, intuitively, visualizations should be very useful. In practice, however, they had rarely realized significant improvements by integrating visualizations into their workflow. A common comment was: "Researchers come to us and say, Here's a visualization tool, let's fit your problem to this tool. But what we really need is a tool built to fit our problem." This is not unique to the security domain, but it suggests that security analysts may be more sensitive to deviations from their current analysis strategies, and therefore less receptive to general-purpose visualization tools and techniques.

This is not to say, however, that visualization researchers should simply provide what the security analysts ask for. Our analysts have high-level suggestions about how they want to visualize their data, but they do not have the visualization experience or expertise to design and evaluate specific solutions to meet their needs. To address these, we initiated a collaboration with colleagues at a major government research laboratory to build visualizations that: (1) meet the needs of the analysts, but also (2) harness the knowledge and best practices that exist in the visualization community.

Again, this approach is not unique, but it offers an opportunity to study its strengths and weaknesses in the context of a cyber security domain. In particular, we were curious to see which general techniques (if any) we could start with, and how significantly these techniques needed to be modified before they would be useful for an analyst. Seen this way, our approach does not focus explicitly on network security data, but rather on network security analysts. By supporting the analysts' situation awareness needs, we are implicitly addressing a goal of effectively visualizing their data.

From our discussions, we defined an initial set of requirements for a successful visualization tool. Interestingly, these do not inform explicit design decisions. For example, they do not define which data attributes we should visualize and how those attributes should be represented. Instead, they implicitly constrain a visualization's design through a high-level set of suggestions about what a real analyst is (and is not) likely to use. We summarized these comments into six general categories:

- **Mental Models.** A visualization must "fit" the mental models the analysts use to investigate problems. Analysts are unlikely to change how they attack a problem in order to use a visualization tool.
- **Working Environment.** The visualization must integrate into the analyst's current working environment. For example, many analysts use a web browser to view data stored in formats defined by their network monitoring tools.
- **Configurability.** Static, pre-defined presentations of the data are typically not useful. Analysts need to look at the data from different perspectives that are driven by the data they are currently investigating.
- **Accessibility.** The visualizations should be familiar to an analyst. Complex representations with a steep learning curve are unlikely to be used, except in very specific situations where a significant cost-benefit advantage can be found.
- **Scalability.** The visualizations must support query and retrieval from multiple data sources, each of which may contain very large numbers of records.
- **Integration.** Analysts will not replace their current problem-solving strategies with new visualization tools. Instead, the visualizations must augment these strategies with useful support.

5 Case Study: Managing Network Alerts

A common task for network security analysts is active monitoring for network alerts within a system. Normally, sets of alerts are categorized by severity—low, medium, and high—annotated with a short text summaries (e.g., a Snort rule), and presented within a web browser every few minutes. The analysts are responsible for quickly deciding which alerts, if any, require additional investigation. When suspicious alerts are identified, additional data sources are queried to search for context and supporting evidence to decide whether the alert should be escalated. Normally, each data source is managed independently. This means results must be correlated manually by the analysts, usually by coordinating multiple findings in their working memory. We were asked to design a system that would support the analysts by: (1) allowing them to identify context more effectively and more efficiently; (2) integrating results from multiple data sources into a single, unified summary; (3) choosing visualization techniques that are best suited to the analysts' data and tasks; but also (4) providing an analyst the ability to control exactly which data to display, and how to present it, as needed.

The analysts' requirements meant that we could not follow a common strategy of defining the analysts' data and tasks, designing a visualization to best represent this data, then modifying the design based on analyst feedback. Working environment, accessibility, and integration constraints, as well as comments from analysts, suggested that a novel visualization with unfamiliar visual representations would not be appropriate. Since no existing tools satisfied all of the analysts' needs, we decided to design a framework of basic, familiar visualizations—charts—that runs in the analysts' web-based environment. We applied a series of modifications to this framework to meet each of the analysts' requirements. Viewed in isolation, each improvement often seemed moderate in scope. However, we felt, and the analysts agreed, that the modifications were the difference between the system possibly being used by the analysts versus never being used. In the end, the modifications afforded a surprising level of expressiveness and flexibility, suggesting that some parts of the design could be useful outside the network security domain.

The configurability, accessibility, scalability, and integration requirements of our design demand flexible user interaction that combines and visualizes multiple large data sources. The working environment requirement further dictates that this happen within the analyst's current workflow. To achieve this, the system combines MySQL, PHP, HTML5, and JavaScript to produce a web-based network security visualization system that uses combinations of user-configurable charts to analyze suspicious network activity.

We adopt Shiravi's definition of situation awareness, "a state of knowledge", and situation assessment, "the process of attaining situation awareness." In this context, the visualization tool is designed to support situation assessment. The expectation is that providing effective situation assessment will lead to an enhanced situation awareness.

5.1 Web-Based Visualization

The visualizations run as a web application using HTML5's canvas element. This works well, since it requires no external plugins and runs in any modern web browser. Network data is visualized using 2D charts [8]. Basic charts are one of the most well known and widely used visualization techniques. This supports the accessibility requirement, because: (1) charts are common in other security visualization systems that analysts have seen, and (2) charts are an effective visualization method for presenting the values, trends, patterns, and relationships that analysts want to explore.

5.2 Interactive Visualization

To realize analyst-driven charts, the system provides a user interface with event handling on the canvas element and the jQueryUI JavaScript library for higher-level UI widgets and operations. This design allows for full control over the data attributes assigned to a chart's axes. This capability turns out to be fairly expressive, and can be used by an analyst to generate an interesting range of charts and chart types. Analysts can also attach additional data attributes to control the appearance of the glyphs representing data elements within a chart. For example, a glyph's colour, size, and shape can all be used to visualize secondary attribute values.

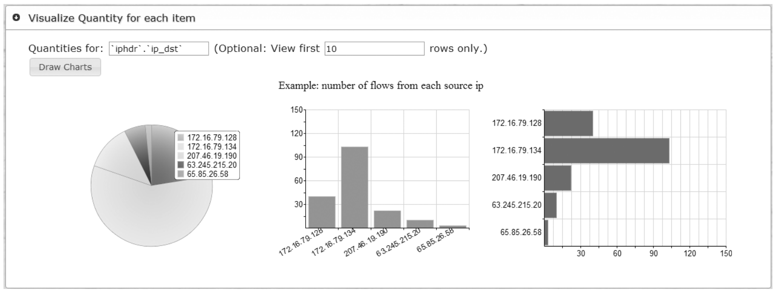
5.3 Analyst-Driven Charts

In a general information visualization tool, the viewer normally defines exactly the visualization they want. The current visualization system automatically chooses an initial chart type based on: (1) knowledge about the strengths, limitations, and uses of different types of charts, and (2) the data the analyst asks to visualize. For example, if the analyst asks to see the distribution of a single data attribute, the system recommends a pie chart or bar chart. If the analyst asks to see the relationship across two data attributes, the system recommends a scatterplot or a Gantt chart.

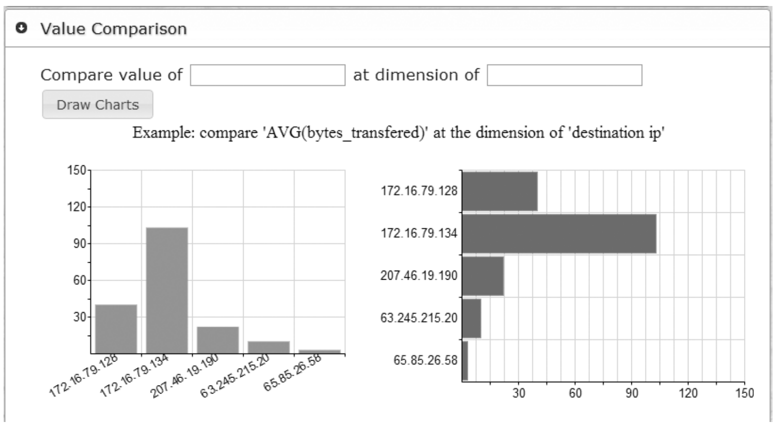
The axes of the charts are initialized based on properties of the data attributes, for example, a categorical attribute on a bar chart's x-axis and an aggregated count on the y-axis. If two categorical attributes like $A = \{A_1 : \text{source IP}, A_2 : \text{destination IP}\}$ are selected, the attributes are mapped to a scatterplot's axes as $V = \{V_1 : x, V_2 : y\}$, with markers shown for flows between pairs of addresses (Fig. 3c).

If the attributes were $A = \{A_1 : \text{time}, A_2 : \text{destination IP}\}$, a scatterplot with $V = \{V_1 : x, V_2 : y\}$ would again be used (Fig. 4a). Visualizing netflow properties like $A = \{A_1 : \text{time}, A_2 : \text{destination IP}, A_3 : \text{duration}\}$ initially produces a Gantt chart with rectangular range glyphs mapped to $V = \{V_1 : x, V_2 : y, V_3 : \text{width}\}$, representing different flows (Fig. 4b). Data elements sharing the same x and y values are grouped together and displayed as a count using additional visual properties. For example, in a scatterplot of traffic between source and destination IPs, the size of each marker indicates the number of connections between two addresses. In a Gantt chart, the opacity of each range bar indicates the number of flows that occurred over the time range for a particular destination IP.

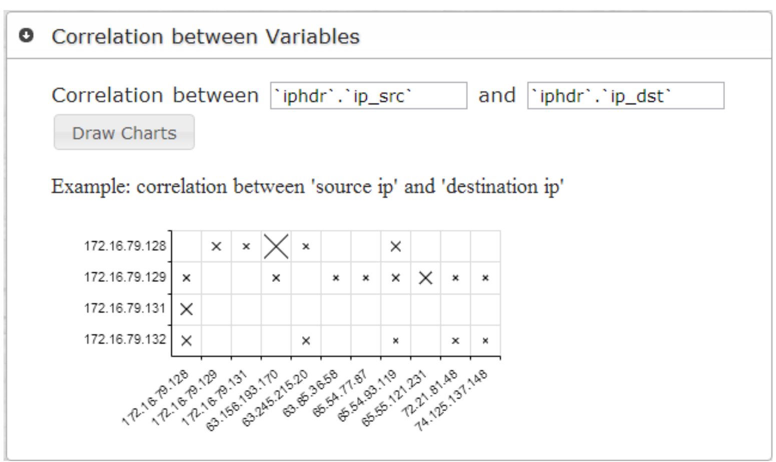
More importantly, the analyst is free to change any of these initial choices. The system will interpret their modifications similar to the processing we perform for



(a)

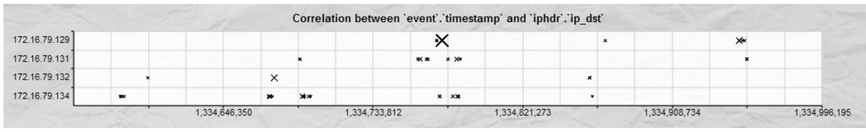


(b)

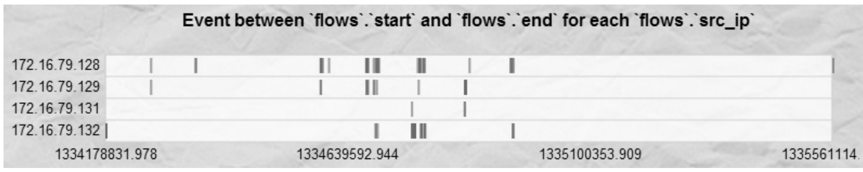


(c)

Fig. 3: Charts classified by use case: (a) pie and bar chart, analysis of proportion; (b) bar chart, value comparison along one dimension; (c) scatterplot, correlation analysis



(a)



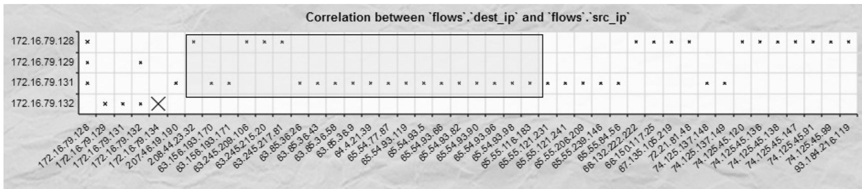
(b)

Fig. 4: Scatterplot and Gantt charts: (a) connection counts over time by destination IP; (b) time ranges for flows by source IP

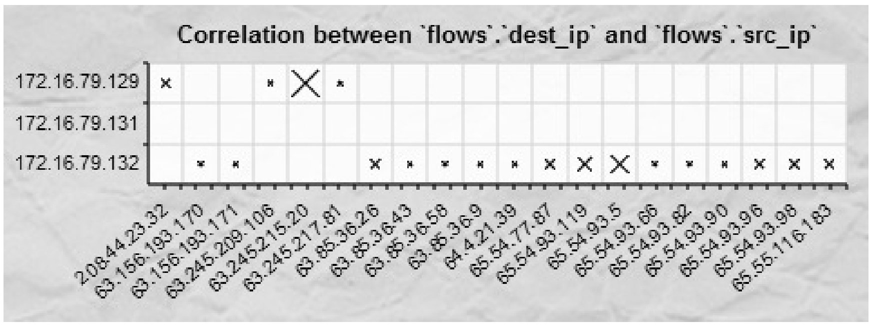
automatically chosen attributes. This allows the analyst to automatically start with the most appropriate chart type (pie, bar, scatterplot, or Gantt) based on their analysis task, the properties of the attributes they assign to a chart's axes, and on any secondary information they ask to visualize at each data point.

5.4 Overview+Detail

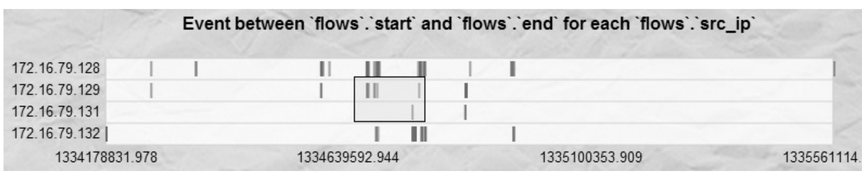
The visualization system allows an analyst to focus on a subset of the data, either by filtering the input, or by interactively selecting a subregion in an existing chart to zoom in on. In either case the chart is redrawn to include only the selected elements. For example, consider a scatterplot created by an analyst, where the size of each tick mark encodes the number of flows for a corresponding source and destination IP (plotted on the chart's x and y-axes). Fig. 5b is the result of zooming in on the sub-region selected in Fig. 5a. In the original scatterplot the difference between the flow counts for the selected region cannot be easily distinguished. After zooming, the size of the tick marks is re-scaled for the currently visible elements, highlighting differences in the number of flows, particularly for destination IP 172.16.79.132 in the bottom row. The same type of zooming can be applied to Gantt charts (Fig. 5c, d). After zooming into a selected area, the flows that occlude one another in the original chart are separated, helping the analyst differentiate timestamps.



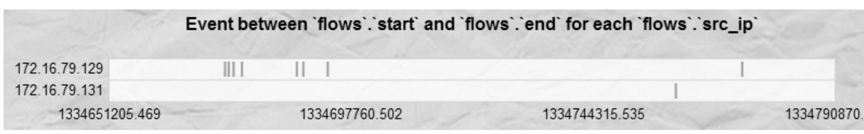
(a)



(b)



(c)



(d)

Fig. 5: Chart zooming: (a) original scatterplot with zoom region selected; (b) zoom result; (c) original Gantt chart with zoom region selected; (d) zoom results

5.5 *Correlated Views*

Analysts normally conduct a sequence of investigations, pursuing new findings by correlating multiple data sources and exploring the data at multiple levels of detail. This necessitates visualizations with multiple views and flexible user interaction. The system correlates multiple data sources by generating correlated SQL queries and extending the RGraph library to support dependencies between different charts.

As an analyst examines a chart, their situation awareness may change, producing new hypothesis about the cause or effect of activity in the network. Correlated charts allow the analyst to immediately generate new visualizations from the current view to explore these hypotheses. In this way, the system allows an analyst to conduct a series of analysis steps, each one building on previous findings, with new visualizations being generated on demand to support the current investigations.

Similar to zooming, analysts can create correlated charts for regions of interest by selecting the region and requesting a sub-canvas. The system generates a constraint to extract the data of interest in a separate window. The analyst can then select new attributes to include or new tables and constraints to add to the new chart.

5.6 Example Analysis Session

To demonstrate the system, we describe a scenario where it is used to explore trap data being captured by network security colleagues at NCSU. The data was designed to act as input for automated intrusion detection algorithms. This provided a real-world test environment, and also offered the possibility of comparing results from an automated system to a human analyst's performance, both with and without visualization support. Four different datasets were available for visualization: a Netflow dataset, an alert dataset, an IP header dataset, and a TCP header dataset.

An NCSU security expert served as the analyst in this example scenario. Visualization starts at an abstract level with an overview that the analyst uses to form an initial situation awareness. This is followed by explorations of different hypotheses that highlight and zoom into subregions of interest. The analyst generates correlated charts to drill down and analyze data at a more detailed level, and imports additional supporting data into the visualization, all with the goal of improving their situation awareness of specific subsets of the network. Including a new flow dataset extends the analysis of a subset of interest to a larger set of data sources. The visualization system supports the analyst by generating different types of charts on demand, based on the analyst's current interest and needs. This leads the analyst to identify a specific NetFlow that contains numerous alerts. This NetFlow is flagged for further investigation.

The analyst starts by building an overview visualization of the number of alerts for each destination IP, $A = \{A_1 : \text{destination IP}, A_2 : \text{alert count}\}$ and using A_1 as the "aggregate for" attribute. Choosing "Draw Charts" displays the aggregated results as pie and bar charts (i.e., using $V = \{V_1 : \text{start angle}, V_2 : \text{arc length}\}$ for the pie chart or $V = \{V_1 : x, V_2 : y\text{-height}\}$ for the bar chart, Fig. 6). This provides an initial situation awareness of how many alerts are occurring within the network, and how those alerts are distributed among different hosts. Pie charts highlight the relative number of alerts for different destination IPs, while bar charts facilitate a more effective comparison of the absolute number of alerts by destination IP. The charts are linked: highlighting a bar in the bar chart will highlight the corresponding section in the pie chart, and vice-versa.

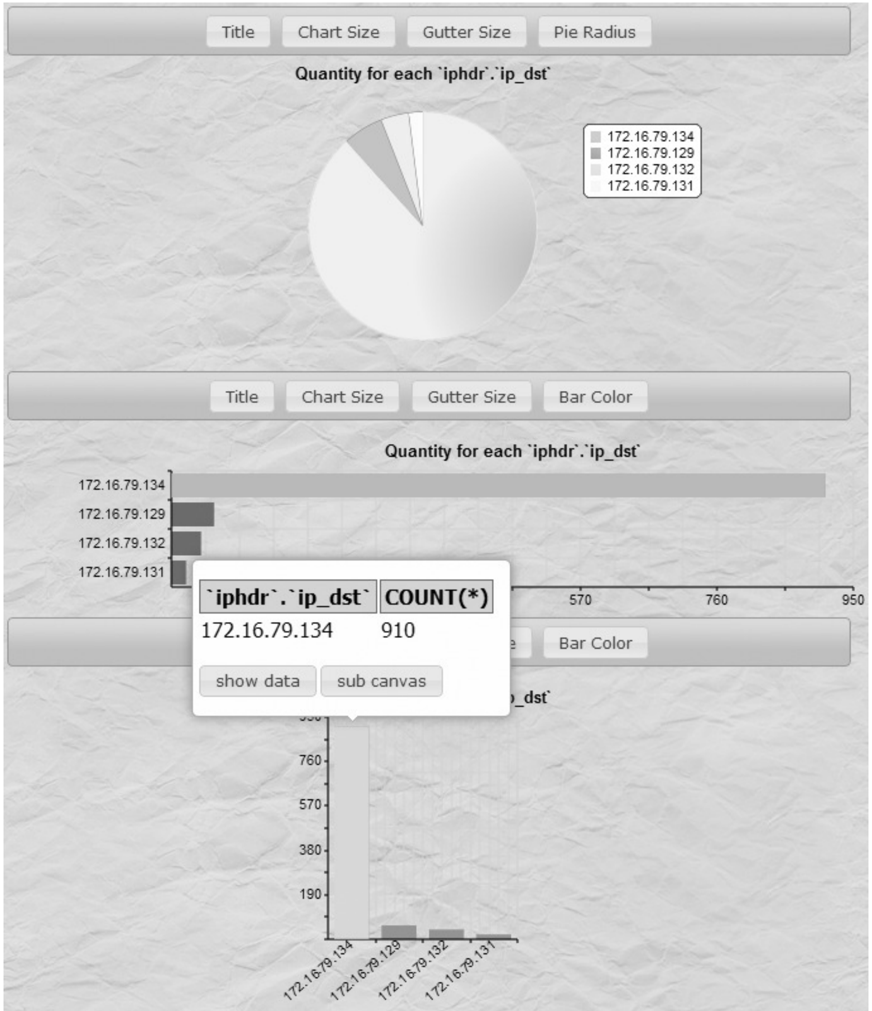
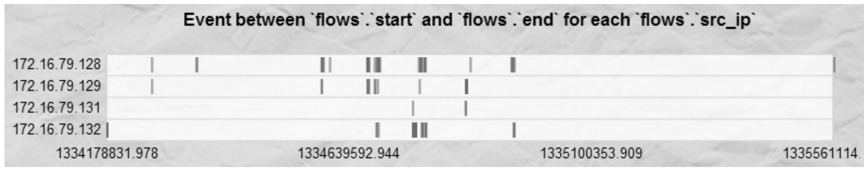
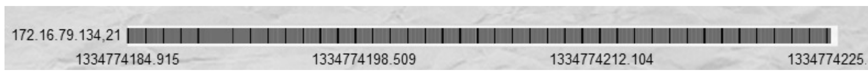


Fig. 6: Aggregated results visualized as a pie chart and horizontal and vertical bar charts

The pie and bar charts indicate that the majority of the alerts (910) happen for destination IP 172.16.79.134. To further analyze alerts associated with this destination IP, the analyst chooses "Sub Canvas" to open a new window with the initial query information (the datasets, data attributes, and constraints) predefined. The filter destination IP = 172.16.79.134 is added to restrict the query for further analysis over this target address. This demonstrates how an analyst can continue to add new constraints or data sources to the query as he requests follow-on visualizations to continue his analysis.



(a)



(b)

(c)

Fig. 7: Gantt chart with alerts for network flows at the destination IP and port of interest: (a) two flows; (b) zoom on the left flow, showing numerous alerts; (c) zoom on the right flow, showing one alert

Next, the analyst chooses to visualize alerts from different source IPs attached to the target destination IP. He uses destination port to analyze the correlation between source and destination through the use of a scatterplot with $A = \{A_1 : \text{source IP}, A_2 : \text{port number}, A_3 : \text{alert count}\}$ and $V = \{V_1 : x, V_2 : y, V_3 : \text{size}\}$. The scatterplot shows there is only one source IP with alerts related to the target destination IP, and that most alerts are sent to port 21. This provides more detailed situation awareness about a specific (source IP, destination IP) pair and port number that the analyst considers suspicious.

The analyst looks more closely at all traffic related to the target destination IP on port 21 by visualizing NetFlows and their associated alerts in a Gantt chart. Here, $A = \{A_1 : \text{start time}, A_2 : \text{duration}, A_3 : [\text{ alert times }]\}$ and $V = \{V_1 : x, V_2 : \text{size}, V_3 : \text{texture hashes}\}$. Collections of flows are drawn in red with endpoints at the flow set's start and end times. Alerts appear as black vertical bars overlaid on top of the flows at the time the alert was detected. Fig. 7a shows most of the flows are distributed over two time ranges. This further augments the analyst's situation awareness, by point to potential times when attacks may have occurred. By zooming in on each flow separately (Fig. 7b, c), the analyst realizes that the vast majority of the alerts occur in the left flow (Fig. 7b). The alerts in this flow are considered suspicious, and are flagged for more detailed investigation. Later discussion with the author of the datasets confirmed that this set of alerts were meant to simulate an unknown intrusion into the system.

This example demonstrates how the system allows an analyst to follow a sequence of steps based on their own strategies and preferences to investigate alerts. The system supports situation assessment based on the analyst's hypotheses about potential attacks within a system. Effective assessment leads to more and more detailed situation awareness, allowing the analyst to confirm or refute the possibility of an intrusion into the system.

6 Summary

Data visualization converts raw data into images that allow a viewer to "see" data values and the relationships they form. The images allow viewers to use their visual perception to identify features in the data, to manage ambiguity, and to apply domain knowledge in ways that would be difficult to do algorithmically. Visualization can be formalized as mapping: data passed through a data–feature mapping generates a visual representation—a visualization—that displays individual data values and the patterns they form. Many existing situation awareness tools use visualization techniques like charts, maps, and flow diagrams to present information to an analyst. The challenge is to determine how best to integrate visualization techniques into a cyber situation awareness domain. Many tools adopt a well-known information visualization approach: overview, zoom and filter, and details on demand. Techniques utilized recently for the security and situation awareness domains include: Charts and Maps, Node-Link Graphs, Timelines, Parallel Coordinates, Treemaps and Hierarchical Visualization. We identified an initial set of requirements for a successful visualization tool. These do not define which data attributes we should visualize and how those attributes should be represented. Instead, they implicitly constrain a visualizations design through a high-level set of suggestions about what a real analyst is (and is not) likely to use: a visualization must "fit" the mental models the analysts use to investigate problems; must integrate into the analysts current working environment; pre-defined presentations of the data are typically not useful; visualizations should be familiar to an analyst; must support query and retrieval from multiple data sources; the visualizations must integrate into existing strategies with useful support.We demonstrate a prototype system for analyzing network alerts based on these guidelines.

References

1. Bertin, J (1967) Sémiologie Graphiques: Les diagrammes, les réseaux, les cartes. Gauthier-Villars, Paris
2. Bradshaw, J M, Carvalho, M, Bunch, L et al (2012) Sol: An agent-based framework for cyber situation awareness. Künstliche Intelligenz 26(2):127–140
3. Chernoff, H (1973) The use of faces to represent points in *k*-dimensional space graphically. Journal of the American Statistical Association 68(342):361–368
4. Cockburn, A, Karlson, A, and Bederson, B B (2008) A review of overview+detail, zooming, and focus+context interfaces. ACM Computing Surveys 41(1):Article 2
5. Dang, K T and Dang, T T (2013) A survey on security visualization techniques for web information systems. International Journal of Web Information Systems 9(1):6–31

6. DeFanti, B H and Brown, T A (1987) Visualization in scientific computing. Computer Graphics 21(6)
7. Goodall, J and Sowul, M (2009) VIAssist: Visual analytics for cyber defense. Paper presented at the IEEE Conference on Technologies for Homeland Security (HST '09), Boston, MA
8. Heyes, R (2014) RGraph: HTML5 charts library. http://www.rgraph.net. Accessed 02 May 2014
9. Johnson, C R (2004) Top scientific visualization research problems. IEEE Computer Graphics & Applications 24(4):13–17
10. Johnson, C R, Moorehead, R, Munzner, T et al (eds) (2006) NIH/NSF Visualization Research Challenges. IEEE Press
11. Kan, Z, Hu, C, Wang, Z et al (2010) NetVis: A network security management visualization tool based on treemap. Paper presented at the 2nd International Conference on Advanced Computer Control (ICACC 2010), Shenyang, China
12. Lakkaraju, K, Yurcik, W and Lee, A J (2004) NVisionIP: Netflow visualizations of system state for security situational awareness. Paper presented at the 2004 ACM Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC '04), Washington, DC
13. Mansmann, F, Fisher, F, Keim, D A et al (2009) Visual support for analyzing network traffic and intrusion detection events using treemap and graph representations. Paper presented at the Symposium on Computer-Human Interaction for Management of Information (CHIMIT 2009), Baltimore, MD
14. McPherson, J, Ma, K, Krystosk, P et al (2004) PortVis: A tool for port-based detection of security events. Paper presented at the Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC '04), Washington, DC
15. Minarik, P and Dymacek, T (2008) NetFlow data visualization based on graphs. In: Visualization for Computer Security, Springer, pp 144–151
16. Phan, D, Gerth, J, Lee, M, Paepcke et al (2007) Visual analysis of network flow data with timelines and event plots. Paper presented in the Proceedings of the 4th International Workshop on Visualization for Cyber Security (VizSec 2007), Sacramento, CA
17. Roberts, J C, Faithfull, W J and Williams, F C B (2012) SitaVis—Interactive situation awareness visualization of large datasets. Paper presented in the Proceedings 2012 Conference on Visual Analytics Science and Technology (VAST 2012), Seattle, WA
18. Shiravi, H, Shiravi, A, and Ghorbani, A A (2012) A survey of visualization systems for network security. IEEE Transactions on Visualization and Computer Graphics 18(8):1313–1329
19. Thomas, J J and Cook, K A (2005) Illuminating the path: The research and development agenda for visual analytics. National Visualization and Analytics Center
20. Tricaud, S, Nance, K, and Saade, P (2011) Visualizing network activity using parallel coordinates. Paper presented in the Proceedings of the 44th Hawaii International Conference on System Sciences (HICSS 2011), Poipu, HI