INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT
*Int. J. Network Mgmt* 2016; **26**:461–491
Published online 20 July 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nem.1945

# Effective network management via dynamic network anomaly visualization

Qi Liao*,† and Ting Li

*Department of Computer Science, Central Michigan University, Mount Pleasant, MI, USA*

SUMMARY

Managed network has become increasingly large, complex, heterogeneous, and dynamic. With tremendous number of network components changing at any moment, deciding which events and changes are possibly abnormal and need further investigation is extremely challenging. While there has been widely adopted practice to log daily activities through standard file formats such as netflow, syslog, firewall, and IDS systems, it is vital for system administrators and network managers to be able to analyze the vast amount of log data in order to detect suspicious behaviors or patterns, possibly because of malicious users/applications or faulty devices. While there are automated systems that are available to generate warnings, whether such alarms are true or false, and more importantly, what are the underlying causes are still difficult to know. To bridge the gap between network logging and anomaly analysis, we design and implement a visualization tool that combines multiple useful visualizations together with algorithms such as graph link anomaly analysis. We study the effects of different visualization methods on detecting and analyzing network and system anomalous events and their causes and show that these views, when combined and linked together, may provide an effective alternative for network management and anomaly analysis. Copyright © 2016 John Wiley & Sons, Ltd.

Received 11 May 2015; Revised 2 May 2016; Accepted 9 June 2016

## 1. INTRODUCTION

The management and analytic complexity of networks has been increasingly challenging over the past decades because of the ever growing size of networks and the explosion of information exchanged over the Internet, known as the *big data* era. There has been common practice for system administrators to record daily operation status data such as netflow, packet traces, syslog, firewalls, and IDS systems. The challenges not only lie in the large volume of such logging data but also the highly dynamic nature of distributed systems and network traffic (i.e., users may come and go, and connections may be built and torn down at any moment). The network data is usually highly dimensional as well with dozens or even hundreds of attributes. How to make sense and find potentially bad or anomalous activities from large amount of standard log data is non-trivial and vital to daily operations. For example, abnormality may be related to faulty hardware/software, misconfiguration, or security related events caused by malicious users and applications. While there has been effort in anomaly-based intrusion detection [1], the detection of abnormal and potentially malicious *connections* has remained challenging. Anomaly analysis is extremely useful in many domains, for example, network managers and administrators need to monitor the latest traffic graphs to increase *situation awareness* for both effective troubleshooting and time-efficient security-related investigation.

In the past, the network anomaly detection problem [2] has focused on statistical or data mining approaches. While these methods have been proved some usefulness in analyzing networks, research

*Correspondence to: Qi Liao, Department of Computer Science, Central Michigan University, Mount Pleasant, MI, USA.
†E-mail: liao1q@cmich.edu

[3] demonstrates challenges in machine learning and data mining approaches in solving complex computer network problems, especially in security domain. In particular, data mining approaches generally are unable to eliminate high error rates (false positives/negatives). Most importantly, there is a so-called semantic gap between the data mining results and decision-making, for example, what does the anomaly mean and what is the reason or cause behind the anomaly.

Because data mining approach alone is probably not enough, there has been research to apply visual data exploration or visual mining [4] to network management and security [5–8]. Visual analytics is a meaningful presentation of data in a visually appealing way, whether using tables, plots, charts, graphs, or any innovative views that can aid knowledge discovery process. The visual data analysis or visual mining process differs in essence from automated methods by incorporating human perception into the process, and it has been suggested that visualization can detect patterns in data missed by traditional automatic methods [9]. While one can use statistical or data mining processes to analyze each of those dimensions of data, it is hard to correlate them in one coherent context for situation awareness. Choosing which attributes to examine can also be a daunting task and computationally infeasible for many automatic data analysis processes.

In this paper, we describe an interactive visualization platform, that is, network anomaly visualization and analysis (NAVA) tool, in combination with automatic statistical and mining methods to address the network management and anomaly analysis task. The advantage of our approach is to bring the human intelligence and experience of domain experts into the loop. By connecting the dots together, the visual analytic tool may quickly identify not only anomalies but possible reasons behind these anomalies. While there have been network management tools, few of them are geared towards anomaly detection of dynamic data. We design and implement a generic network log analysis and visualization tool for situation awareness and anomalous event investigation. The visualization tool (Figure 1) incorporates four interrelated views, that is, link anomaly graphs, parallel coordinates, treemaps, and Gantt charts, by taking advantages of user familiarity and robustness. Each view is suitable for different characteristics of specific data attributes whether one would like to analyze the overall trend over time, distribution of values, or details-on-demand. The interactive features such as zoom-and-pan and linking-and-brushing allow investigators to connect dots together, analyze the network anomaly from different angles, and form a complete picture.

Each view we developed is suitable for analyzing a different type of network data with its own characteristics and anomalies. Network data types have different properties, which makes them hard to be dealt with using any particular visualization technique. In particular, link anomaly graphs examine anomalies in network connectivity. A network graph can be constructed by flow-like data or packet traces collected during a specific time period. Because networks are highly dynamic in terms of topological changes, we apply a link anomaly detection algorithm for dynamic network graphs. Our model includes a time dynamic function and similarity measurement based on the evolving network topologies in consecutive time windows. The time dynamic function takes an assumption that the connections close to the time of investigation should receive more emphasis than earlier connections while the similarity measurement takes metrics such as Jaccard coefficients and Katz measure. During each time window, two types of connection anomalies are visualized with different styles and colors, that is, two unconnected computing nodes that should have connected and two connected nodes that should not have connected.

The parallel coordinates views allow an investigator to examine network flows and their associated attributes in addition to source and destination addresses and ports. In addition to link attributes, the node attributes may be visualized in treemaps and Gantt charts. Treemaps are ideal for overviews of magnitude of node attributes, such as CPU, memory usage, number of network connections, and other health status data, in hierarchical addressing space (network and sub-networks). Gantt chart views are generic enough to allow a timeline trend view of any node and edge attributes and their relative changes over time. Different color codings may be applied to show normal or anomalous changes in network and system health. We study the pros and cons of these views in the setting of network security management, and more importantly, how these views, when linked together, can provide a much better situation awareness and investigation assistance to system administrators and network managers. Implemented with D3 [10], the generic, web-based visualization platform allows network administrators and other researchers to easily view and collaborate on security data analysis and visualization.



DOI: 10.1002/nem

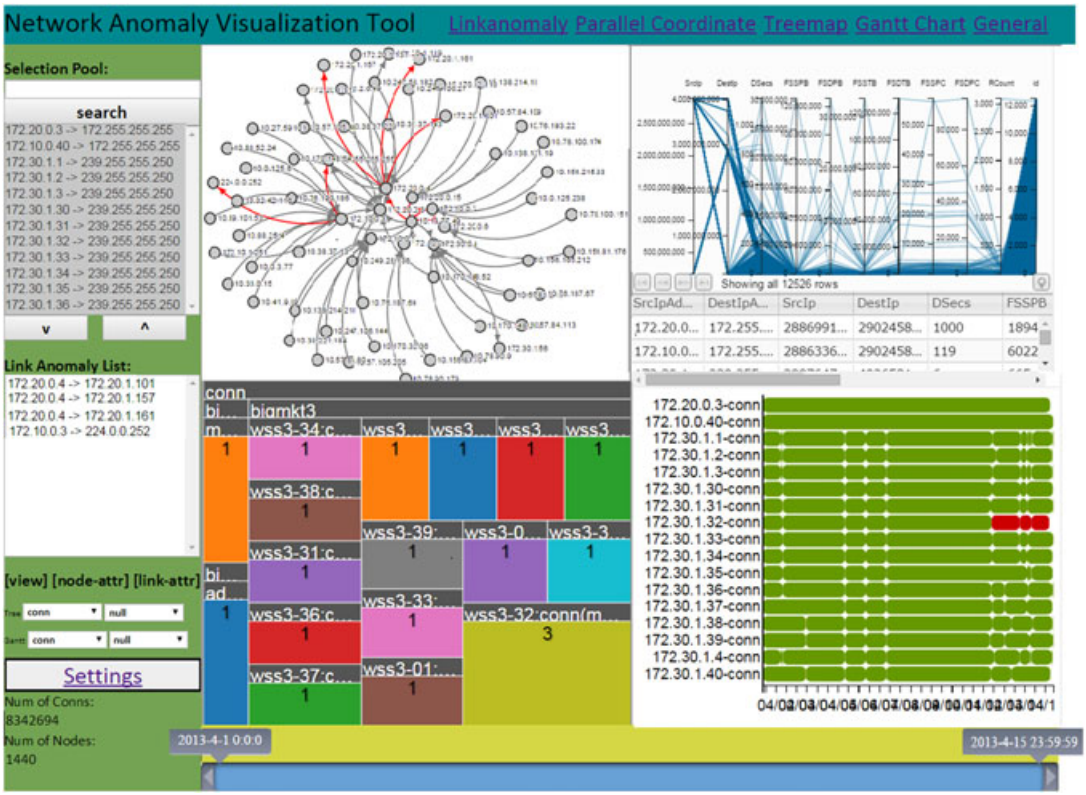



Figure 1. Overview of interactive, web-based network anomaly visualization and analytic (NAVA) tool for analyzing network and system log files.

Evaluation through case studies over public dataset shows positive results on the efficiency and effectiveness of the proposed tool in identifying and analyzing various network anomalies from standard log files.

The rest of the paper is organized as follows. Section 2 discusses the related work in data mining and visualization approaches to network security and anomaly analysis. Section 3 discusses in detail the visualization design for each view in the tool, that is, link anomaly graph, parallel coordinations, treemaps, and Gantt charts. The link anomaly detection algorithm and entropy calculation for attributes are also discussed in this section. Section 4 describes the architecture of the system and detailed implementations of data processing, anomaly extraction, database structure, and information flow in the server/client model. In Section 5, we conduct an extensive case study over two cyber-security datasets and demonstrate a few usage examples and how the tool may help to identify anomalies and analyze their causes. Finally, Section 6 concludes the paper.

## 2. RELATED WORK

In areas of security management, data mining methods for detecting intrusion and anomalies [1,2] may be applied to network flow data [11], network graphs and links [12,13], host-user-application (HUA) data [14]. While data mining and machine learning approaches have shown effectiveness in detecting zero-day attacks without predefined signatures, automated mining and learning has fundamental limitations and challenges because of the characteristics of many security domains [3], such as noisy data, lack of data for evaluation, high error rates, and semantic gaps between results, interpretation, and decision-making.

Visual data mining [4], visual mining for short, is a promising alterative for data analysis and has its applications in computer and network security, known as security visualization [5–8]. While both traditional data mining and visual mining provide knowledge, visual data exploration brings domain



DOI: 10.1002/nem

experts into the loop and allows interactive exploration, filtering, and cross-linking. One may write a program to detect anomaly pattern *if* the pattern is known ahead. Unfortunately, many times network administrators and operators may not necessarily know what anomalies to expect. In computer and network security management, the dynamic nature and multi-dimensional property of security log data make analytic task challenging. While one can use statistical or data mining processes to analyze each of those dimensions of data, it is hard to correlate them in one coherent context for situation awareness. Choosing which attributes to examine can also be a daunting task and computationally infeasible for many automatic data analysis processes.

Visualization has been proved useful in situation awareness [6], network management and security [5,7], wireless network security, and computer forensics. Visualization-based intrusion detections and network defense [15,16] can help administrators understand dynamics of graphs and find patterns of malicious activities using graph drawing [5], graph clustering [17,18], and graph hierarchies [19]. While a few general visualization tools exist (Cytoscape, Pajek, Gephi, and Titan project/The Visualization ToolKit), these programs are designed to visualize and analyze the general and/or biological network graphs. In face of the dynamics of computer networks with heterogeneous graphs involving hosts, users, and applications, many generic graph algorithms would fail to produce meaningful results (e.g., general graph visualization will not show the evolution of community of users by comparing different graphs, and cleverly suggest which changes are abnormal and need further investigation).

Gantt charts [20] are widely used in project management, job scheduling [21], and so on. It is intuitive to show trends over time among some attributes of network log data such as packet transmission and data usage. The comparison of attributes by both numbers and percentages usually helps to pin down exact start and end times of interesting events. Because networks can be naturally organized into trees and graphs, there has been a vast amount of work to design different visualization solutions addressing issues of viewability or usability. Treemap [22], as another visualization solution, has a different role in our work. Considering the comparable design of sizes and colors of treemap, we embed quantities to show significant differences between nodes' unsynchronized performances and keep them in a time-node hierarchical structure.

Other visualizations have been designed that may potentially help administrators to investigate and detect anomalies in network traffic data. For example, a parallel axes [23] view has been used to display NetFlow records and generate network traffic patterns of both normal and malicious behavior. FlowScan [24] uses one area-chart-like plotting solution to analyze flow data and provides a continuous view of the network traffic. In addition, network anomalies may also be predicted by directly visualizing the statistical result [25]. Visualization work [26,27] has been conducted on cyber-security intrusion detection in VAST challenge data [28]. While parallel axes [23] and FlowScan [24] primarily target at the flow data, networked systems also include many other important aspects such as machines status and IDS alarms. Using only one pattern layer without considering other attributes may have issues for detecting actual anomalous network activities. Visualizing statistical results [25] that have a tendency for predicting anomalies is like a double edged sword. Although accurate prediction will efficiently help investigators, wrong results will be possible to mislead administrators. A collaborative information linking method [29] may enhance the current general view by bridging visualizations. Visual links across different views could quickly lead the user to discover the relative knowledge in other views after selecting an item. Lastly, there are also commercial network management tools such as HP OpenView, IBM Tivoli, CA NSM, and Splunk. While these are more heavyweight solutions that manage network components in one central place, we focus on an alterative interactive network and system log visualization scheme that can assist managers to identify general anomalies and their causes.

Lastly, we note that modern networks such as large enterprise networks are becoming more and more complex and dynamic. The magnitude of connections and data flow as well as interrelationship among events are too complex to be analyzed using simple Structured Query Language (SQL) queries or traditional statistical analysis. In addition, the networks also become very dynamic, that is, the topologies are constantly changing when the old connections are torn down and new connections are established at any moment. The challenging part is that some changes should be considered as normal while some other changes should not. Our link anomaly algorithm used in the dynamic network anomaly visualization tool helps investigators identify the potential anomalous changes in dynamic networks.



DOI: 10.1002/nem

As mentioned earlier, one challenge of using data mining and machine learning approaches to solve security problem is the inherent error rates and semantic gap. While data mining and machine learning are generally useful as a starting point of investigation, they only provide suggestions or hints. The human investigator must be able to find out if those hints are real problems, and if so what the underlying causes are for the anomalies. Our approach combines both data mining and visualization techniques. The proposed link anomaly algorithms suggest possible anomalous network connections while the visualization tool allows the administrator to explore the dataset interactively to reconstruct the causality relationship among network events. In addition, such visualization tool may be useful for network forensics.

## 3. VISUALIZATION DESIGN FOR NETWORK ANOMALY ANALYSIS

In terms of the choice of visualizations, our basic design principle is to keep it simple and general by taking advantage of user familiarity and visualization robustness so that it introduces minimum learning curve and potentially adapts to more standard log file format. The basic layout of the visualization interface includes four primary views, that is, link anomaly graphs, parallel coordinates, treemaps, and Gantt charts. Each view may be suitable for exploring different characteristics of network and system log and may have pros and cons in the setting of network management and anomaly analysis tasks. More importantly, we study how these views, when linked together, can provide a much better situation awareness and investigation assistance to network operators.

Figure 1 shows an overview of the visualization tool. On the left panel, there are various options for users to interact with the network log data. Users can search specific nodes and select any combination of node and edge attributes on the system. Upon search and selection, the views on the right will be automatically updated.

Specifically, the top-left panel reflects the link anomaly graph, which not only shows the end-to-end connections between machines but, more importantly, shows whether a connection is anomalous through different colors and styles of edges. The top-right panel shows the parallel coordinates, defined by each source and destination IP addresses and port numbers, together with significant attributes of the connections, for example, total count, duration, packet count, and flow size. Each flow may be explicitly selected for investigation or filtered based on certain criteria. The bottom-left panel provides a scalable, hierarchial view of attribute values over selected time. This provides a quick overview based on the magnitude of attributes of each machine in relation to other computers in the subnets. The bottom-right panel demonstrates, through Gantt charts, any node or edge's attribute values across a selected time period. Different colors are used to denote the health status, for example, red color means problems. This view provides insight on overall trend of network health.

There are additional interactive visual items, for example, buttons, for settings such as which database to use for the visualization tool. With the double-end time slider bar at the bottom of the visualization tool, network administrators can select any time period, thus filtering out large amount of data to investigate only a selected range of start and end time stamps.

### *3.1. Link anomaly graph*

The network data is dependent, meaning the status information on one machine may be caused by events or connections from another node. Therefore, it is often insufficient to investigate one standalone node independently. Therefore, network graphs are helpful for administrators to investigate the interdependency relationship between nodes. To analyze anomalous events within a network in a selected investigation period, a network administrator needs to know which machines are connected on the network and whether these connections are normal or not. Link anomaly graph is designed to provide visualized view for network administrators to find out possible anomalous network connections on the network.

We begin by first introducing our similarity score functions to identify potential anomalies, which are visualized in link anomaly graphs. In theory, there are two types of link anomalies, that is, Type-I (missing connections that should have appeared) and Type-II (established connections that should not occur). In order to detect both Type-I and Type-II link anomalies, we utilize several similarity metrics



DOI: 10.1002/nem

[30] such as Jaccard's Coefficient and Katz Index. Let $G = (V, E)$ be the graph that represents the topological structure of a general connected network. Edges in the graph are denoted by $e = (u, v) \in E$, where $u, v \in V$, and $V$ is the set of nodes or vertices. For each $u \in V$, $\Gamma(u)$ represents the set of $u$'s distinctly adjacent nodes (or neighbors). In addition, the set of paths between $u$ and $v$ is defined as $paths_{u,v}^{(l)}$ where $l$ represents the exact length of the path. Each edge $e \in E$ counts 1 for unweighted graphs and $w(e)$ for weighted graphs, in which $w \in W$ is referred to as the related edge $e$'s weight and $W$ is the set of edges' weights.

The Jaccard coefficient is well known to measure the similarity between sample sets, which is defined as the size of the intersection divided by the size of the union of the sample sets [31]. Particularly, in the graph $G$, for each pairwise nodes $v, u \in V$, the coefficient is defined as the ratio between the number of their common neighbors and the number of total neighbors, namely

$$J(u, v) = \frac{\Gamma(u) \cap \Gamma(v)}{\Gamma(u) \cup \Gamma(v)} \tag{1}$$

In our approach, for instance, it indicates whether two network nodes, for example, clients, servers, and routers, have a large percentage of overlapping destinations regardless of the absolute number of connecting targets. In addition, the Katz measure [32] is a variant of the shortest-path measure [31], which considers that the more simple paths there are between two nodes and the shorter these paths are, the stronger is the relationship of two nodes. The measure is defined as

$$K(u, v) = \sum_{l=1}^{\infty} \beta^l \cdot \left| paths_{u,v}^{(l)} \right| \tag{2}$$

where $0 < \beta < 1$ is a parameter ensuring that the shorter path contributes more to the score. Finally, we apply a weighted time frequency function [18] with the temporal dimension to consider the time dynamics of network connection:

$$T(u, v) = \frac{\sum_{t=1}^{N} w(t) \cdot d}{\sum_{t=1}^{N} w(t)}, d_{t,(u,v)} \in (0, 1) \tag{3}$$

$$w(t) = e^{-\lambda\left(1 - \frac{t}{N}\right)} \tag{4}$$

The appearance probability functions can be either weighted or unweighted. The weighted form (equation (3)) takes a nonlinear time weighting function $w(t)$ (equation (4)), that is, the appearance of links at later snapshot graphs (or in other words closer to the time of investigation) should have higher weights over the earlier graphs. Both equations 3 and 4 are normalized between 0 and 1, where $T(u, v)$ represents the temporal probability of $i$th link between node $u$ and $v$. $N$ denotes the number of snapshot graphs, and $d_{t,i}$ takes a binary form to denote whether $i$th link appears or not at time $t$.

While $J$ function measures the similarity of one-hop neighbors in dynamic network graphs, $K$ function considers the nodes that are more than one-hop away. In addition to the topological dynamics, $T$ takes into consideration the temporal locality of network connections. Let $S(u, v)$ be the score of a pair of node $u$ and $v$ and be a linear combination of the $J(u, v)$, $K(u, v)$, and $T(u, v)$. Spatial–temporal dynamics are unified into one similarity score for link anomaly prediction. For each non-connected node pairs $(u, v)$, the Type-I link anomaly occurs when $S(u, v) > \Theta_u$. For each connected node pairs $(u', v')$, the Type-II link anomaly occurs when $S(u', v') < \Theta_l$, where $\Theta_u$ and $\Theta_l$ are the thresholds of upper and lower bounds.

For example, consider a time series of networks (Figure 2) containing five past snapshot graphs (Figure 2(a–e)), the union of graphs $G_{t-4}$ to $G_t$ (Figure 2(f)), and the future network graph (Figure 2(g)). $J(3, 4) = \frac{\Gamma(3) \cap \Gamma(4)}{\Gamma(3) \cup \Gamma(4)} = 0.2$. Suppose $\beta = 0.5$, all simple paths between 3 and 4 are {3,4}, {3,1,4}, and {3,1,2,4}, and $K(3, 4) = 0.5^1 \cdot |1| + 0.5^2 \cdot |1| + 0.5^3 \cdot |1| = 0.875$. The edge (3,4) appeared in the third and fifth snapshot graphs. Suppose $\lambda = 5$, $T(3, 4) = \frac{e^{-2} + e^0}{e^{-4} + e^{-3} + e^{-3} + e^{-2} + e^{-1} + e^0} = 0.72$. Assume the constants are 0.1, 0.8, 0.1, the linear combination of final score $S(3, 4) = 0.1 * 0.2 +$



DOI: 10.1002/nem

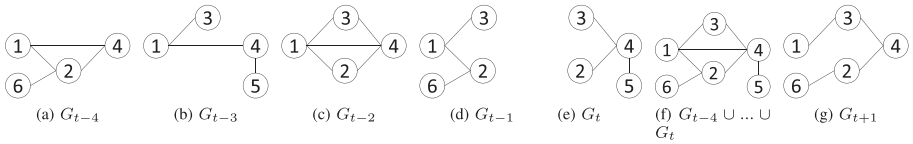



Figure 2. Example of computing graph link anomaly scores.

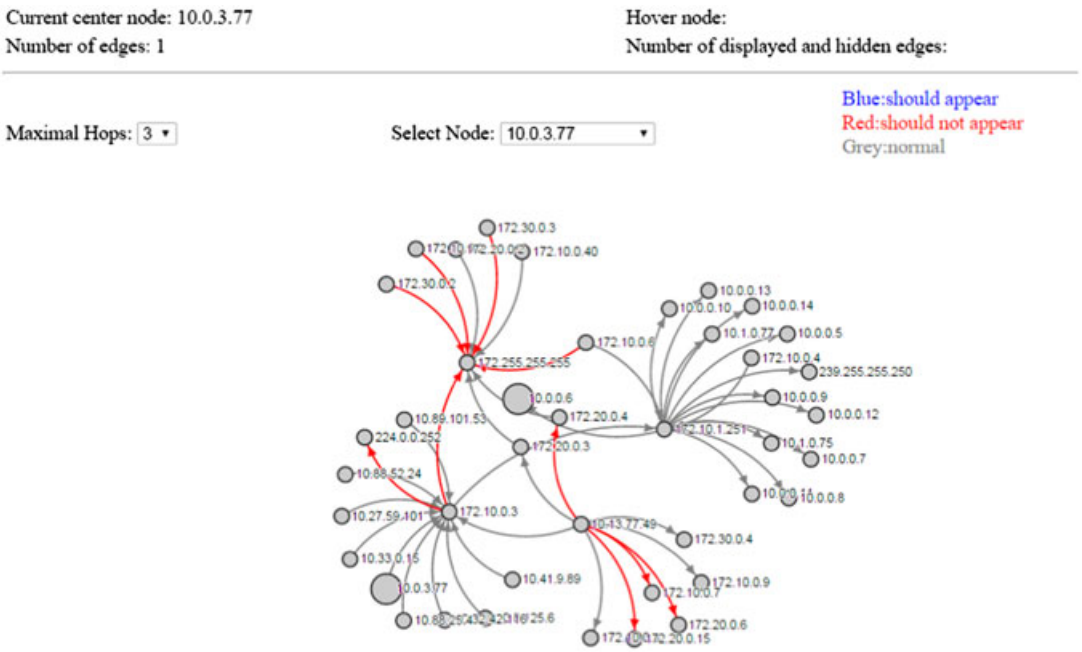



Figure 3. A link anomaly graph showing the connections between machines. The directed, curved edges with gray color mean the normal connections. Red edges indicate the connections should not appear (Type-II link anomalies).

$0.8 * 0.875 + 0.1 * 0.72 = 0.792$. If an upper threshold of 0.75 is used, the edge (3,4) of network in the next time window $G_{t+1}$ will be considered normal. If an upper threshold is changed to 0.9, such an edge will be suggested as Type-II link anomaly. While there is no-size-fits-all, depending on how many anomaly suggestions one wants to see (e.g., top $n$), parameters such as weights and thresholds may be customized by network managers depending on different networks.

Figures 3 and 4 show screenshots of such network graph link anomaly visualization. Different edge colors are used to show whether the connections are normal or not in the selected investigation period. In current implementation, blue dotted lines are used to indicate two unconnected machines that should connect (Type-I anomaly). Red solid lines indicate that two connected machines should not connect to each other (Type-II anomaly). Gray lines indicate that the connections are considered normal. After selecting an investigation period using the time bar, one can easily find out which connections are possibly anomalous on the network with link anomaly graph by observing the colors/styles of edges. On the left panel of the tool (Figure 1), there is a link anomaly list. Using $\vee$ and $\wedge$ buttons, we can add or remove anomaly connections found in link anomaly graph to or from the link anomaly list for further investigation using other views.

The graph view supports standard graph exploration functions such as zoom and pan, drag and drop, and mouse hover tips actions. Directed lines between nodes indicate that two machines are connected in the selected investigation period. *Curve shaped* edges are adopted for better perception of node relationships in larger networks. Node labels may be hidden or visible to allow more space for larger graphs. In the information panel on the top of the graph, the IP address of current center node in the graph is displayed. In addition, when mouse hovers on a certain node, the node will be highlighted



DOI: 10.1002/nem

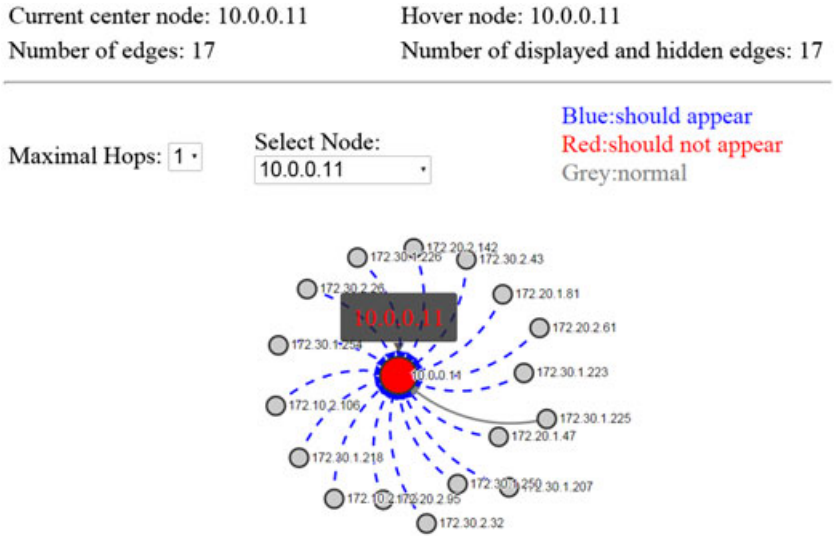



Figure 4. Blue directed edges in link anomaly graph indicate the connections should appear, but in reality, they did not appear (Type-I link anomalies).

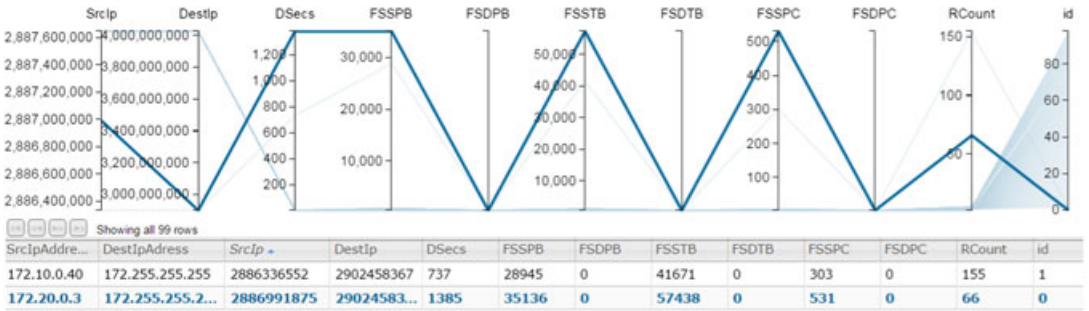



Figure 5. Attributes of network connections (edges) and their anomalies can be visually examined in detail from parallel coordinate system. Each individual connection (e.g., 172.20.0.3→172.255.255.255) may be highlighted by selection in the bottom table.

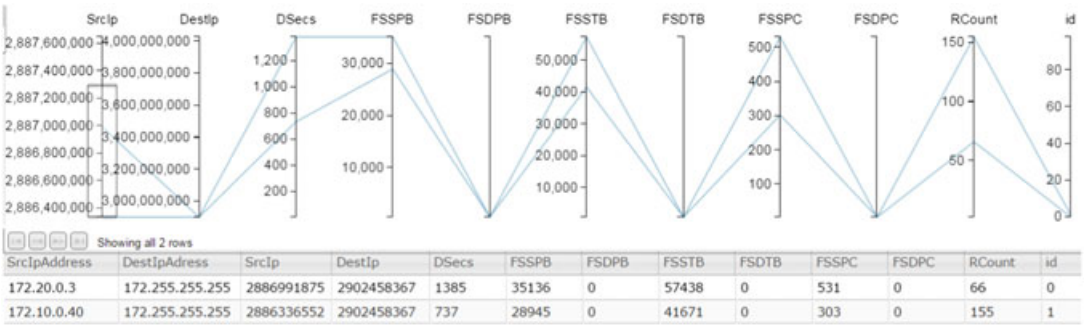
in red, and its label (e.g., IP address) will be displayed in a pop-up textbox. While one can click any node in the graph that he/she is interested in directly, nodes may also be quickly selected from a drop-down list in case there are many nodes in the graph. The selected node will become the new center node (highlighted in yellow color), and all nodes connected to it will be shown. To be more *scalable* to larger networks, the visualization tool is designed to render only a subgraph, and filtering options are available to let users control the distance in terms of the number of hops from a selected node.

### *3.2. Parallel coordinates*

After knowing some possible anomalous connections from link anomaly graphs, the network administrator may need to find out what attributes may have caused these anomalies. In parallel coordinates, we treat every connection appearing in the selected investigation period as an individual, extract important attributes of each connection from the network flow data or packet traces, and display them in the parallel coordinate. The attributes shown in Figure 5 are SrcIP (source IP address), DestIP (destination IP address), DSecs (duration seconds), FSSPB (first seen source payload bytes), FSDPB (first seen destination payload bytes), FSSTB (first seen source total bytes), FSDTB (first seen destination total bytes), FSSPC (first seen source packet count), FSDPC (first seen destination packet count), and RCount (record forceout). For simplicity, source and destination port numbers are set invisible by default but can be enabled if necessary. Attributes of flows are significant for network administrators



DOI: 10.1002/nem




| SrcIpAddress | DestIpAdress | SrcIp | DestIp | DSecs | FSSPB | FSDPB | FSSTB | FSDTB | FSSPC | FSDPC | RCount | id |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 172.20.0.3 | 172.255.255.255 | 2886991875 | 2902458367 | 1385 | 35136 | 0 | 57438 | 0 | 531 | 0 | 66 | 0 |
| 172.10.0.40 | 172.255.255.255 | 2886336552 | 2902458367 | 737 | 28945 | 0 | 41671 | 0 | 303 | 0 | 155 | 1 |

Figure 6. Large number of connections may be filtered by dragging selection boxes on any combination of coordinates. For example, a range of source IP may be selected to display only those connections involving those source IP addresses.

to analyze the connections in the network. For example, *RCount* attribute tells us the frequency of two machines connecting to each other during the selected investigation period. If two machines have an unusually high number of connections in a short period of time, there may exist anomalies such as port scans.

The connections' attributes and their related values are shown respectively in the parallel coordinate system. In the bottom table, as shown in Figure 5, every connection's attribute values are explicitly displayed. The network administrator can enlarge each table cell to check the exact value of every attribute. He or she can also choose to show the connections and their attributes in ascending or descending sequence by using the button next to every column name. The network administrator can select which connections to show by using the scroll bar at the right part of the table or using the page turning button on the top part of the table. If the network administrator moves his or her mouse over certain connection in the table, the connection will be highlighted both in the table and in the coordinate, as shown in Figure 5.

Comparing to the table view, the top coordinate system gives us a more visualized view of every connection in the selected investigation period. In the coordinate, each vertical coordinate axis represents one dimension of each connection, and each connection's attribute values are displayed on the bars. Therefore, instead of a dot in a traditional 2D space, there is a line in multi-dimensional space that represents a network connection and its attributes. We can have a general view of whether the connection is anomalous or not by comparing each of its attribute values with other connections in the coordinate. If certain connection's specific attribute values are much larger or smaller compared with other connections, then we can assume that there may be something wrong with this connection, and the corresponding anomalous attributes may be the cause. As there may be a lot of connections during a selected investigation period, we can filter data on an arbitrary data dimension by dragging a selection box on an axis. For example, we can show connections with only a certain range of IP addresses, which can significantly reduce the amount of data to be examined visually. When we find out some anomalous connections, we can show them solely in the coordinate by pitching on them on any vertical bars, as shown in Figure 6. Correspondingly, only the filtered down connections will be shown in the bottom table. We can also move the position of each vertical bar by dragging the related value near it.

### *3.3. Treemaps*

Servers and workstations are significant components of a network. Network administrators have to check these machines routinely to make sure the whole network works well. From link anomaly graphs and parallel coordinates, the administrator may have identified an anomalous-machine-candidate list from the connectivity perspective. However, the administrator does not necessarily know what is happening on a specific machine. The next target for the administrator is to find out which machine properties mainly cause the anomalous event. To solve the problem, we try to illustrate the true anomaly by using the treemap view through comparison of magnitudes of machines' representative properties corresponding to the events.



DOI: 10.1002/nem

Table 1. Examples of machine (node) properties that can be analyzed in the visualization tool

| Property | Meaning |
|---|---|
| conn | ability to connect to the system |
| cpu | percentage of CPU usage |
| disk | percentage of disk usage |
| mem | memory usage |
| pagefile | pagefile usage |
| smtp | server SMTP status |
| degree | incoming and outgoing connections |
| entropy | entropy of above values |

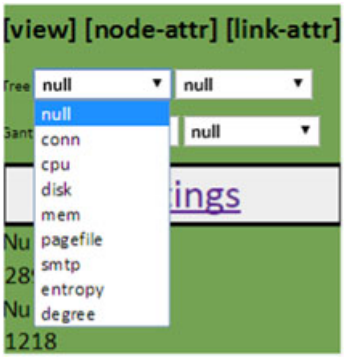



Figure 7. The option panel in the visualization tool allows users to choose any combination of node (machine) properties or link (network connection) attributes for visualizations in treemaps and Gantt charts.

The treemaps' view may be used to examine the property status values of both machines (nodes) and links between them. Depending on the logging content in different organizations, whether utilizing syslog monitoring mechanism and/or Simple Network Management Protocol (SNMP) enabled devices (routers, switches, hosts, etc), there can be several properties about the server and workstation machines. Table 1 shows an example of such machine (node) properties. The visualization tool allows the administrator to choose which machine and/or link properties to examine from the drop-down menu in the left option panel shown in Figure 7. The amount of data to be visualized can be filtered by selecting two things. First, dragging the double-end time-range slider shows only the data within the time range falling into the examination period. Second, putting the interesting connections into the anomalous candidate list shows properties of the machines in question.

Once we have selected which connections, machine properties, and time range to display, the treemap view displays the related machines' property status in rectangles of different sizes and colors. For each rectangle, the area size corresponds to the status value of each machine during the specified time window. Every rectangle has a title text above it to indicate the name of the machine and its status value of certain property (Figure 8). Neighboring machines are displayed in different colors to distinguish each other. Machines are also grouped according to which sites or subnets they belong (e.g., bigmkt1-3). Rectangles of different sizes indicate different health status of a machine. For example, when a rectangle's size equals to one, it means the machine's property status is good or normal. When a rectangle's size equals to two, it means a warning on the machine's status. Three tells that there is problem with the machine's property, and four indicates the machine does not send any status information at a certain time. There is positive relationship between machines' property status value and their anomalous situation; therefore, it is intuitive to use bigger rectangles in treemap view to indicate anomalies with machine's certain property.

Treemaps give us a general view about machines' (servers, workstations, etc.) certain property status. Sometimes, there may be many rectangles rendered at the same time, and it may not be easy to



DOI: 10.1002/nem

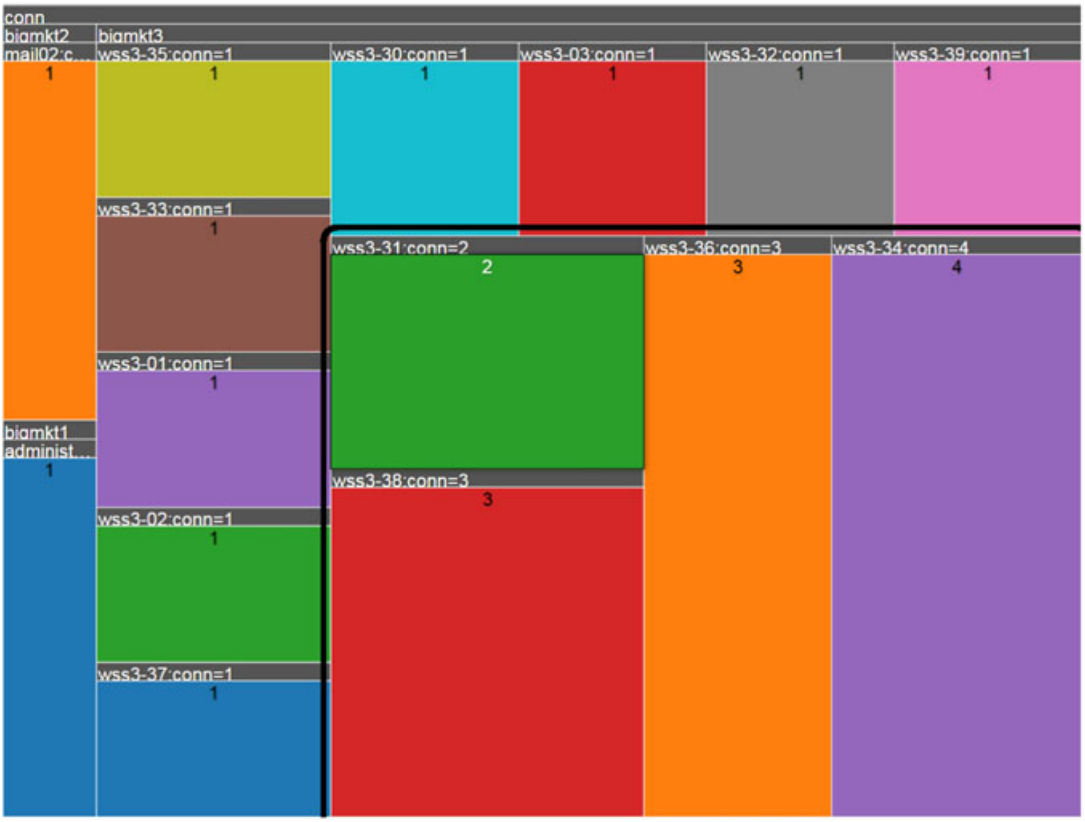



Figure 8. Hierarchical rectangles in the treemaps' view may represent properties of servers or workstations in subnets. The area size of rectangles can be used to visualize machines' status, for example, in this case, larger sizes suggest possible anomalous connectivity status of servers (highlighted).

find out which rectangle is anomalous that has the greater size. The administrator can check machines in each network respectively by clicking the (sub)network title in the treemap view, and only machines belonging to the selected subnet will be displayed. The administrator can also click a certain rectangle, and the selected rectangle will be enlarged in the view. A subsequent mouse click will quit the one rectangle view and go back to the original view. After comparing each machine property rectangle, the administrator can drill down to certain machines, which are the most possible causes of an event. As shown in Figure 8, in a selected investigation time period, the *conn* properties of machines such as *wss3-34/36/38* are greater than those of other machines, suggesting there may be something wrong with these machines' connection ability.

### *3.4. Gantt charts*

While the treemaps' view provides summary machine status within a specified time range, to further investigate and learn more precise knowledge about when the status of machines turns, the administrator needs to know the detailed changing trend of the machines' property over time, which can be achieved via the Gantt chart view.

Same with the treemap view, the tool allows the selection of connections, node for edge properties, and time range of investigation. For each selected property during an event's time period, a Gantt chart plots each machine as a single visual element and uses different colors to represent the related property at each time slice during the selected investigation period (Figures 9 and 10). The $x$-axis in the Gantt view is used to show the selected investigation period, and the $y$-axis represents machines' IP addresses; the value for selected property is rendered over time stamps.

In addition to the machine properties (Table 1) from syslog and SNMP monitoring, such as connectivity, CPU, and memory, entropies [26] of such properties may be calculated. For example, for



DOI: 10.1002/nem

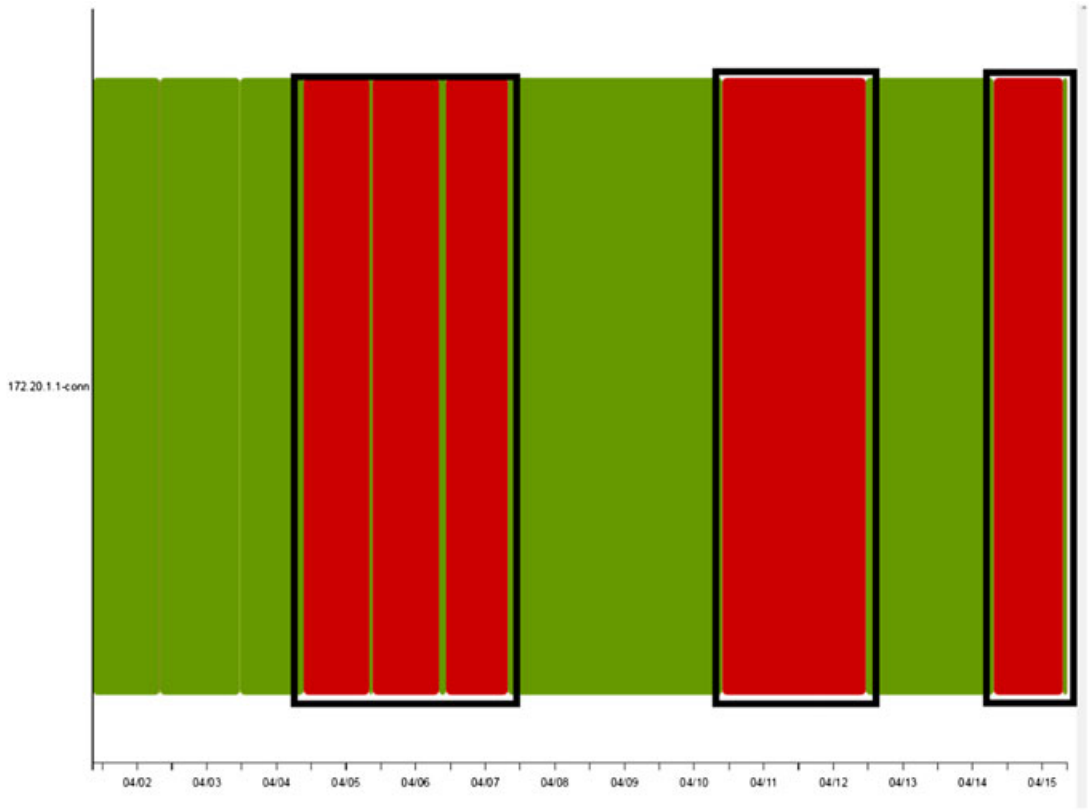



Figure 9. A Gantt chart showing the *connection* status of a server (172.20.1.1) over 2-week's time period. The green color suggests normal status while other colors indicate the severity of events. For example, the server's *conn* (connect ability) property status shows anomalies (red, highlighted) at various time points.

the network connections at each computing node, we may count the frequency of *i*th link from/to that node and calculate the appearance possibility of each connection $p_i$ by using the equation as follows:

$$p_i = \frac{|L_i|}{\sum_{i=1}^{n} |L_i|} \tag{5}$$

where $|L_i|$ denotes the occurrence times of *i*th link. A node's entropy can be computed using the following equation:

$$E = -\sum_{i=1}^{n} P_i \log_2 P_i \tag{6}$$

In order to make it easier to analyze in the Gantt chart, we normalize the entropy to the interval [0,1] as $E_s$, where

$$E_s = \frac{E - E_{min}}{E_{max} - E_{min}} \tag{7}$$

Different colors are used to represent different health status of machines. This is similar to levels of urgency in many report system such as *syslog*. For example, green color is for normal (level one), yellow means warning (level two), and red for abnormal (level three). A detailed color mapping for property values of one dataset is shown in Table 2. For the node entropy property shown in Table 1, we



DOI: 10.1002/nem

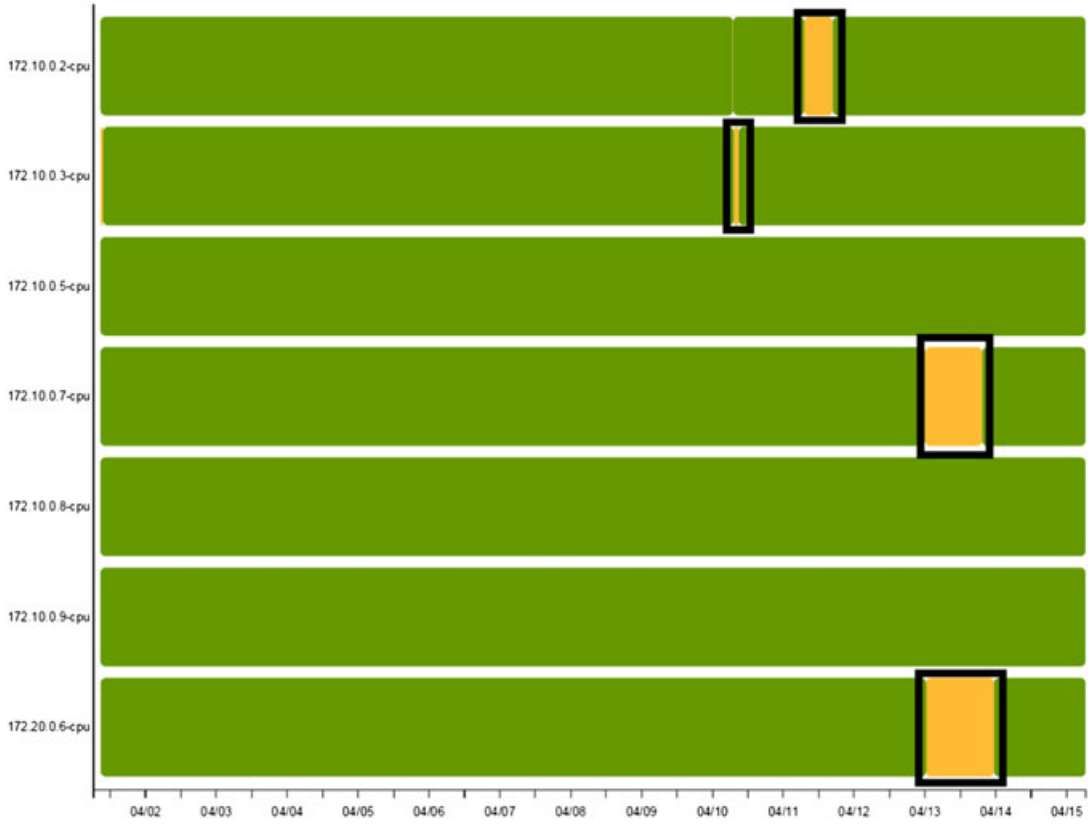



Figure 10. A Gantt chart showing the overall trend of seven machines' CPU usage status change over 2-week's time period. Yellow color (highlighted) suggests a warning level. Timeline trend view allows partial causal ordering and anomaly event correlations.

Table 2. Color legend for machine properties listed in Table 1

| Level | Status | Color |
|---|---|---|
| 1 | Good | Green |
| 2 | Warning | Yellow |
| 3 | Problem | Red |
| 4 | Did not send status info | Blue |

use different gradient colors to represent machines' entropy status value. The gradient color spectrum is defined in the following equation:

$$Color_i = Color_S + \frac{i * (Color_E - Color_S)}{n} \tag{8}$$

where $n$ depends on the temporal data size and $i$ is in the range from 0 to $n - 1$. In equation 8, $Color = \{R, G, B\}$ is a three-tuple containing values of red, green, and blue; $Color_S$ and $Color_E$ represent starting color and ending color, respectively. In our case, we use blue (*#33FFFF*) as the starting color and red (*#ff0000*) as the ending color. With this color spectrum model, we can associate machines' entropy value with a specific color and further display it in the Gantt chart view. While the normalized machines' entropy values range from 0 to 1, the 10 gradient colors change correspondingly from blue to red. If there is a radical change, either an increase or a decrease of one machine's entropy value, color changes abruptly in a selected period of time. Such changes suggest potential anomalies of health status of the underlying machines.



DOI: 10.1002/nem

From the Gantt chart view, the administrator can observe the changing trend of each machine's property status value during the investigation period. For example, from Figure 9, the administrator can learn that there are some problems with the connect ability of the machine 172.20.1.1 from April 5 to 7 and from April 11 to 12, because its *conn* property is shown in red color. Figure 10 shows an intuitive vertical comparison between machines that highlights every target whose trend of changing values is significantly different from others. For example, the *cpu* status of machines such as 172.20.0.6 and 172.10.0.7 indicates yellow warning on April 13 and 14. If there is anomalous event during this time period on the network, it is most likely that this anomaly is caused by these two machines, which need more attention from the administrator for further investigation. Another advantage of comparing the timeline status change is that it is now possible to do a partial casual ordering or, in other words, to analyze a causality relationship between anomalous events by simply observing which status changes 'happen before' others.

### *3.5. Discussion on linking and brushing and scalability*

The individual views we discussed earlier focus on creating presentation with two data structures, that is, the connectivity between nodes and the attributes of nodes and edges. Node attributes describe a network element's information, for example, system status. Network data records usually correlate to two nodes, and pairwise nodes have common attributes to describe the link. For instance, a network flow record has a source and destination IP addresses and port numbers. Their relevant attributes such as first seen packet count, total bytes transmit, and duration are also related to two nodes' properties in the current conversation. Considering every connection record as an isolated item will make the workload to increase exponentially.

Cross consulting and linking between each individual view is another important procedure during the entire network anomaly analysis task. The idea is to combine the aforementioned visualization methods to overcome the shortcomings of a single technique. We create the visualization tool by connecting the four views together and establishing the contact between visual items of each view. With the interaction such as selecting a specific node in one view or from the anomaly list menu, other views will automatically reflect the updated information with respect to the selected item. For example, if the administrator selects a specific node in the network graph chart in the visualization tool, by treating the selected node as the source node, Gantt chart will automatically plot all connections' destination IP nodes as visual elements. Node health attributes such as connect ability, CPU, and memory usage could be shown as granularity changed colors. Meanwhile, treemap uses rectangles' sizes to show which host dominates a particular attribute. In addition, parallel coordinates will provide detailed information about each connection.

The idea behind this design is that by connecting multiple views through interactive linking and brushing, the visualization tool provides more information than considering each component independently. This is especially useful for network anomaly analysis and investigation. The investigator will consider all evidence and make the conclusion such as which connection and system are anomalous. Furthermore, linking will empower the administrator to aggregate scattered anomaly into a correlated event and a bigger picture. Additional conclusion may be drawn, such as which node is the attacker in a malicious intrusion attempt. Attack types may also be inferred by the connections. For instance, if one attacker makes connections to many victims, it may be a portscan behavior, and reversely, if one victim node has many anomalous links included, it may have suffered from a DDoS attack.

Lastly, we note that we have several mechanisms implemented to address the scalability issues in our visual analytic tool. First, we include a time selection bar in the visualization tool (Figure 1). By selecting a time period of investigation, there is no need to extract the entire dataset for analysis. Only records that fall between the start and end time will be queried from the database and shown in the tool. Second, the link anomaly graphs support user selection of particular nodes and the maximal hops (Figures 3 and 4). An investigator may select one or more nodes as the focus nodes. The maximal hops $h$ limit the graph sizes by only including nodes and edges that is $h$-hop away from the focus nodes. This tremendously improves the scalability of data analysis and visualization. Third, the filtering functions in the interactive visualization tool further reduce the amount of information for analysis. For example, Figure 15 shows that a user may select a value range on any of the parallel axes to highlight



DOI: 10.1002/nem

only network connections that fall between the ranges. Fourth, the design of hierarchical views further facilitates scalable data analysis. Some views such as link anomaly graphs and parallel coordinates allow detailed analysis while some views such as treemaps allow overview, which can usually accommodate larger datasets.

## 4. SYSTEM ARCHITECTURE AND IMPLEMENTATION

The overview of architecture of the network log analysis and visualization tool is shown in the system implementation diagram in Figure 11. In the following section, we describe in details the system components, their relationships, and the data analytic flow for anomaly investigation.

### *4.1. Logging and data processing*

As introduced in previous sections, there are four major views in the tool, that is, link anomaly graphs, parallel coordinates, treemaps, and Gantt charts. Each view has its own focus of the network and system log data. Choosing which data to display in the views makes the tool more efficient in analyzing network anomalies. The data used in this paper include network flow data and network health data. In network flow data, a series of exchanged messages between two computers are combined into a single flow record with other attributes associated with the flow such as packet counts, duration, and sizes. This type of data can be collected via Cisco netflow or other flow-like data including data from firewalls or IDS system. In network health data, servers or workstations' detailed status information is also described as a record. The health status data such as connectivity, CPU, and memory can be uploaded via syslog remote server mechanism and via SNMP protocol enabled devices. The actual SNMP management information base objects and variables depend on the enterprise implementation. Because those management information base objects are treated as either node or edge attributes, the visualization tool is general enough to readily read and visualize such attributes. For link anomaly graph and parallel coordinates, we mainly utilize the network flow data. For treemaps and Gantt charts, network and system health data are mainly used.

To track network connection and dig down to the source of network anomaly, we need to extract essential attributes from the original log data. The extraction process includes selecting significant attributes, dividing into time windows, and converting to the desired format. The data processor is written in Python, which is a popular script language with good string parsing support. The extracted data from the netflow data containing the source and destination information is passed to link anomaly detection module (Section 3.1). Each connection in the time windows (5 min) is evaluated for normal or abnormal by comparing the calculated score with actual connection situation. Link anomaly scores along with all other processed data are then put into the SQL database.

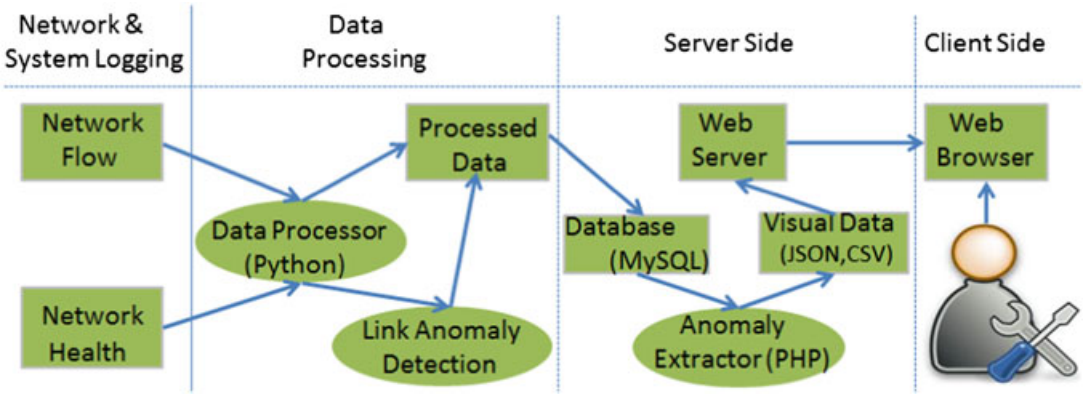



Figure 11. Architecture overview and system components diagram. The standard logging data are processed and stored in database. Anomalies are analyzed and extracted using web technologies and visualized via web browsers over network.



DOI: 10.1002/nem

### 4.2. Server side

A web server (Apache) is set up to host user visualization requests from a client machine. The web server utilizes an anomaly extractor (AE) written in PHP to query the database and generate the required JSON files for clients' visualization in web browser using JavaScript, as shown in Figure 11. Dynamic time selection is one important feature of network anomaly visualization tool. When administrators on the client side choose certain period of time, the tool will give them related information for analysis. Therefore, time selection is one of the main reasons that we design the server side's database.

The structure of SQL database is organized by three tables, that is, *anomalyGraph*, *networkFlow*, and *networkHealth* tables, which are queried and rendered by different views as shown in Table 3. The *anomalyGraph* table stores pairwise source and destination IP addresses, start and end time, link anomaly scores, and connectivity situation as shown in Table 4. The start and end time is for the range query by the administrator. The connectivity situation indicates whether each pair of nodes have been connected in the past or not and their current connectivity status. The link anomaly score is computed and used for link anomaly detection when combined with the threshold and connectivity situation.

The *networkFlow* table is used by almost all views in the tool. The table records the established connections within network and more importantly, the attributes associated with the flows, as shown in Table 5. Once the administrator moves the time bar in visualization tool, records whose time falls between the selected period will be selected, and further analyzed by the administrator for selected connections' detailed information in parallel coordinates and other views.

Table 3. Database tables and corresponding views

| **Tables** | **Views** |
|---|---|
| anomalyGraph | Link anomaly graph |
| networkFlow | Parallel coordinates, treemap and Gantt chart (degree, entropy) |
| networkHealth | Treemap and Gantt chart (conn, cpu, mem, disk, etc.) |

Table 4. Attributes in the *anomalyGraph* table

| **Attributes** | **Examples** |
|---|---|
| srcIp | 10.0.125.238 |
| destIp | 172.10.0.7 |
| linkAnomalyScore | 0.0000537739 |
| beginningTime | 2013-04-01 07:50:00 |
| endTime | 2013-04-01 08:05:00 |
| connectivitySituation | Old |

Table 5. Attributes in the *networkFlow* table

| **Attributes** | **Examples** |
|---|---|
| parsedDate | 2013-04-01 07:50:16 |
| srcIp | 172.20.0.3 |
| destIp | 172.255.255.255 |
| srcPort | 514 |
| destPort | 514 |
| durationSeconds | 29 |
| firstSeenSrcPayloadBytes | 600 |
| firstSeenDestPayloadBytes | 0 |
| firstSeenSrcTotalBytes | 1104 |
| firstSeenDestTotalBytes | 0 |
| firstSeenSrcPacketCount | 12 |
| firstSeenDestPacketCount | 0 |
| recordForceOut | 0 |



DOI: 10.1002/nem

Table 6. Attributes in the *networkHealth* table

| Attribute name | Example |
|---|---|
| id | 36017 |
| hostName | administrator.bigmkt1.com |
| hostIp | 172.10.0.40 |
| serviceName | cpu |
| currentTime | 1364832314 |
| statusVal | 1 |
| parsedDate | 2013-04-01 09:05:14 |

The *networkFlow* and *networkHealth* tables can both be used in treemap and Gantt chart views. In *networkHealth* table, each server or workstation's health status information is saved as one record, as shown in Table 6. Possible values for *serviceName* are suggested in Table 1. Based on the information, we may derive additional important attributes, such as node degrees or connectivity entropies, which may be analyzed in the treemaps and Gantt Chart views.

The AE, written in PHP language, extracts and integrates final information from the tables in the SQL database so that the information can be presented in all views of the network anomaly visualization tool. When the administrator performs some operations, such as selecting a period of time, typing keywords in the selection pool, or adding some connections into the link anomaly list, the AE will query the database and transform the result into different data format in the form of JSON or CSV files.

For link anomaly graph, the AE extracts information from the *anomalyGraph* table and decides whether one connection is normal or not, as discussed in Section 3.1, according to their *connectivitySituation* and *linkAnomalyScore*. Depending on the data distribution in different organizations, the threshold percentage may vary. After getting the information of each connection's type (normal or anomalous), the AE will write the information into JSON files.

For the parallel coordinates' views, the AE performs a range query based on the values returned by the time slider bar at the client browser and translates the *networkFlow* table into a CSV file. For the treemaps' views, the AE combines the information from both the *networkHealth* and *networkFlow* tables. Once the administrator selects the time period and what nodes and their attributes to analyze, AE extracts nodes whose *serviceName* value equals the selected attribute and then calculates total *statusVal* for the nodes in question. Based on the provided attributes in the *networkFlow* table, AE also calculates each node's degree value in the selected period. After each node's total *statusVal* and/or degree is calculated, AE writes these information into JSON files, which will be read by the treemaps' view.

For the Gantt chart view, AE performs the similar operation as for the treemap views. The difference is that instead of computing the total *statusValue* for each selected node in a certain period of time as for the treemap view, the AE records every single *statusValue* of a node at each timestamp for the Gantt chart view. The AE also computes the node connectivity entropy as discussed in Section 3.4. The AE first divides the selected time range into intervals (1 h in our case) and then calculates the network entropy for each node that appears in the network during that time interval. After getting *statusValue* and/or the standard entropy for each node, the AE writes the final information into JSON files, which will be read by the Gantt chart view.

### *4.3. Client side*

All of the four views in the client side are based on JavaScript and implemented using D3 [10] library. We also utilize a few other JavaScript packages such as JQuery and JQrangslides in HTML for data display. The administrator can simply visit a URL to visually explore the dataset from the database. We choose a server-client model because it is generic and convenient rather than adopting a desktop application approach. The administrator can access and analyze his or her data using any web browser running on an inexpensive PC and from anywhere he or she has network access.

As the administrator switches between views, making filtering and selection, the operation will be transmitted under the asynchronous JavaScript and XML (AJAX) mechanism to the server side asynchronously. For example, when the administrator moves the time bar in the visualization tool, the



DOI: 10.1002/nem

time bar's start time and end time will be transmitted to the AE on the web server via (AJAX), and consequently, the AE makes queries from the back-end database. The data finally reaches to the client side for visualization. By exploring and analyzing the data displayed in each of the views provided by the network anomaly visualization tool, the administrator can correlate events happened on his or her network, identify potential anomalies, and pin down the underlying causes for the anomalies.

## 5. CASE STUDY

In this section, we evaluate the visualization tool through several publicly available datasets, that is, VAST challenge [28] datasets, and demonstrate a few common usage of the tool. Through the interactive exploration of the dataset, we explore a variety of anomaly events and their underlying causes and further illustrate how they can be effectively identified with the visualization tool.

### *5.1. VC13 data description and processing*

In this section, we use VAST 2013 Mini Challenge 3 dataset as an example to perform the case study. The dataset contains network security data from an international marketing company (Big Marketing) that consists of around 1200 workstations and servers. Besides the common network traffic logs such as network flow data and intrusion prevention system data, the 2-week dataset also contains a commercial network monitoring program that provides network health and status data for every single workstation and server, which periodically reports status updates including conn, cpu, disk, mem, pagefile, and smtp. The meaning of these six properties are shown in Table 1. Each machine property or node status has a value range of 1–4, indicating the health status of such server or workstation. A different color (explained in Table 2) is used in Gantt charts for visualizing the trend of status change over time for each node.

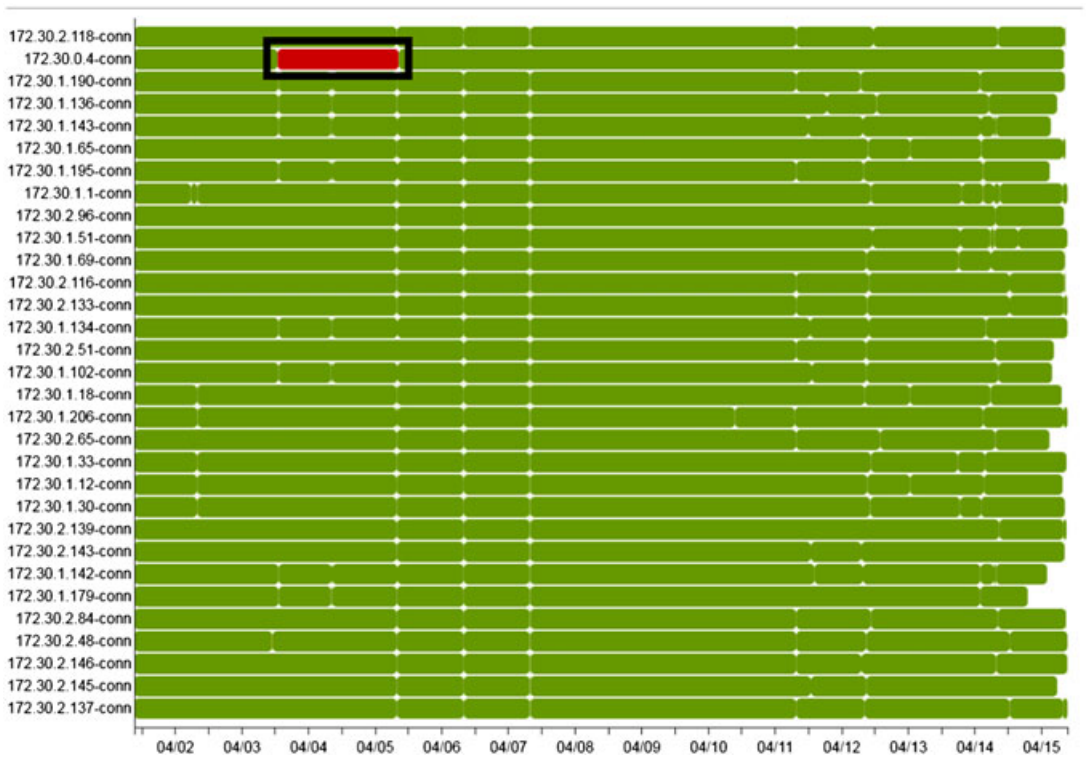



Figure 12. The Gantt chart view of *conn* property (connect ability) of servers over 2-weeks' logging period. During the period of April 2–5, web server *web03* (172.30.0.4)'s *conn* status is red suggesting possible anomaly for further investigation.



DOI: 10.1002/nem

As a starting point, we divide the records of the entire dataset into two groups based on machines' roles. Group one contains information about servers and the administrator workstations, and group two includes user workstations' records. Machines in both groups are further distributed into three different sub-groups according to which subnet they belong to. In this case, there are three subnets, they are *bigmkt1*, *bigmkt2*, and *bigmkt3*.

Suppose the company's system and network administrator, Adam, needs to examine his system log data for potential bad activities. Because anomaly events can happen at any time and any types of anomaly can appear, Adam often feels such tasks difficult as he has no clue where to begin with the dataset on hand. Therefore, Adam decides to explore his log data by using the NAVA tool for his daily network management task.

### 5.2. *DoS attack*

From April 2 to 5, some users in the Big Marketing company complain that one of the corporate web sites becomes unresponsive. Adam reboots machines in the data center and has all web sites and relevant network functionalities on other servers back online. After solving the problem, Adam realizes that the reason for the web site crash is still unknown, which might be important because similar or worse events could happen in the future if current issue is due to hardware-level hidden faults in the data center's machines or, even worse, from malicious attacks.

To find out which part of the network is the main reason that causes this anomaly and what exactly the anomaly is, Adam loads corporate network logs in the reported period from monitoring facilities into the Network Anomaly Visualization tool and moves the bottom time bar to the related time period.

Firstly, Adam refers to the Gantt chart. In this view, Adam finds server *web03*'s (IP address 172.30.0.4) *conn* status value in the selected period is anomalous compared with other machines. As shown in Figure 12, the status value of the machine(172.30.0.4) is red, while other machines are mostly

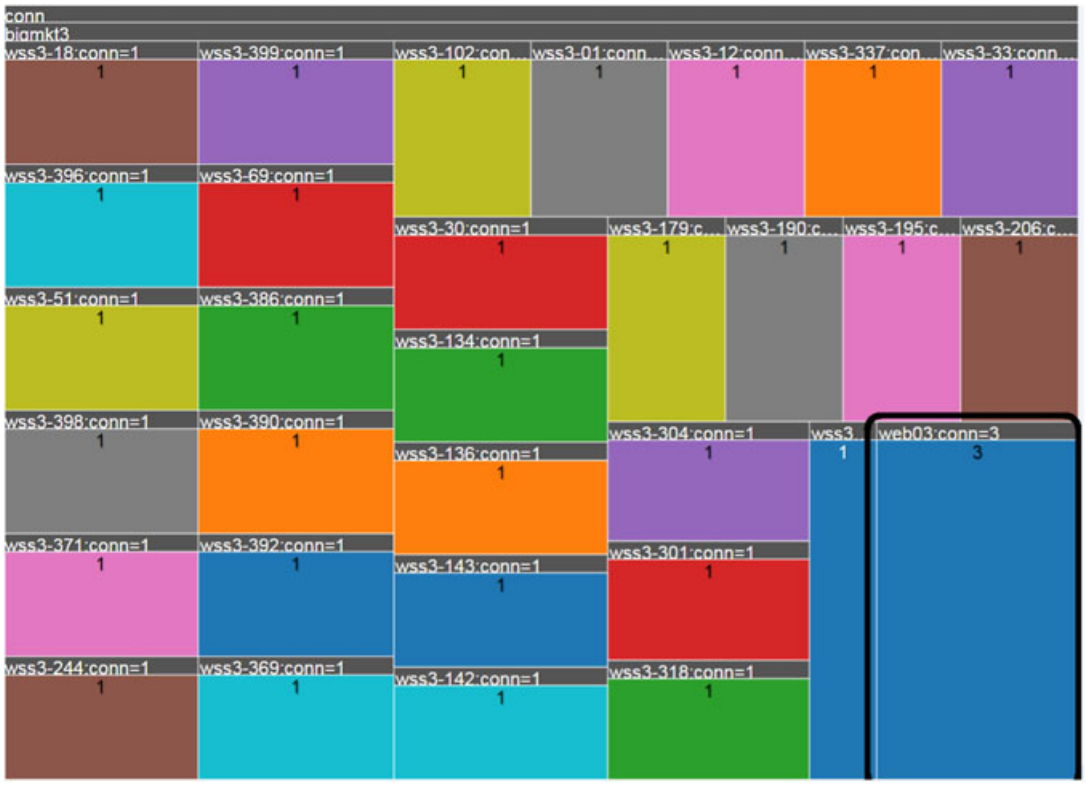



Figure 13. The treemap view of machine's *conn* property in the period of April 2–5. Web server *web03* (172.30.0.4)'s rectangle size is larger than those of other machines suggesting anomaly with the connectivity of the server.



DOI: 10.1002/nem

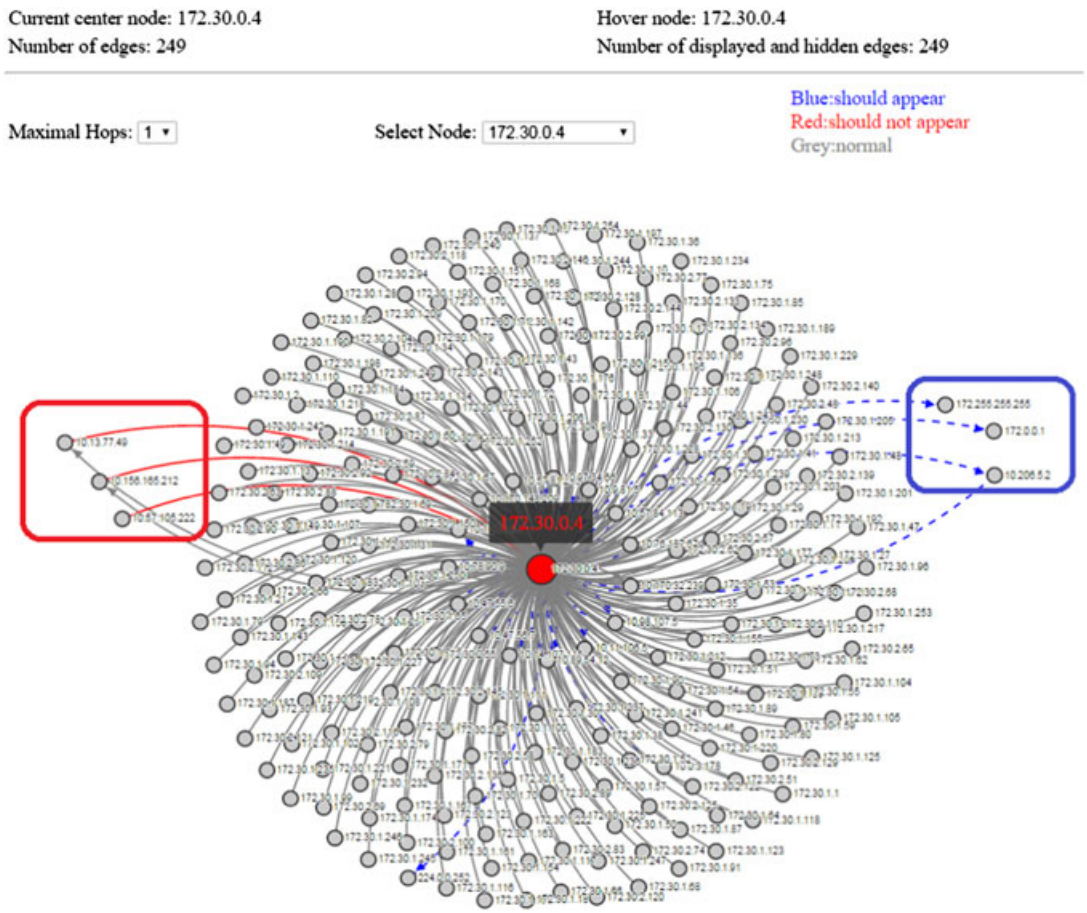



Figure 14. Link anomaly graph during the period of April 3–5, with one hop view from the selected node 172.30.0.4 (*web03*). Blue directed edges suggest Type-I link anomalies, that is, a few machines such as 172.255.255.255 should connect to *web03* but not in reality. Red edges suggest Type-II link anomalies, that is, machines such as 10.156.165.212 should *not* connect to *web03*.

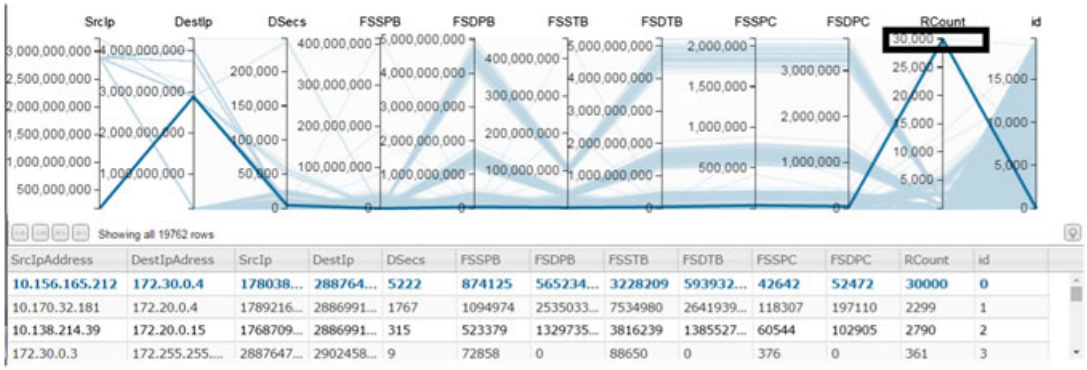



Figure 15. Parallel coordinates' view during the period of April 3–5 related to the web server *web03* (172.30.0.4). A few connections' *RCount* (frequency of connection appearance) values are much higher compared with other connections, for example, 10.156.165.212–172.30.0.4's *RCount* is as high as 30 000 (highlighted), suggesting *web03* is under DoS attack from these machines, which causes the connectivity anomaly of the server.



DOI: 10.1002/nem

green or occasionally yellow. The problem with *conn* status falls between the reported problem time period, and an inference could be made that the event may relate to actions by server *web03*.

In order to verify his hypothesis, Adam then digs into the treemap view in the reported period. In this view, Adam discovers that the rectangle representing *web03*'s *conn* property is significantly larger than those of other machines, shown in Figure 13. While other machines' *conn* status value is 1 (normal),

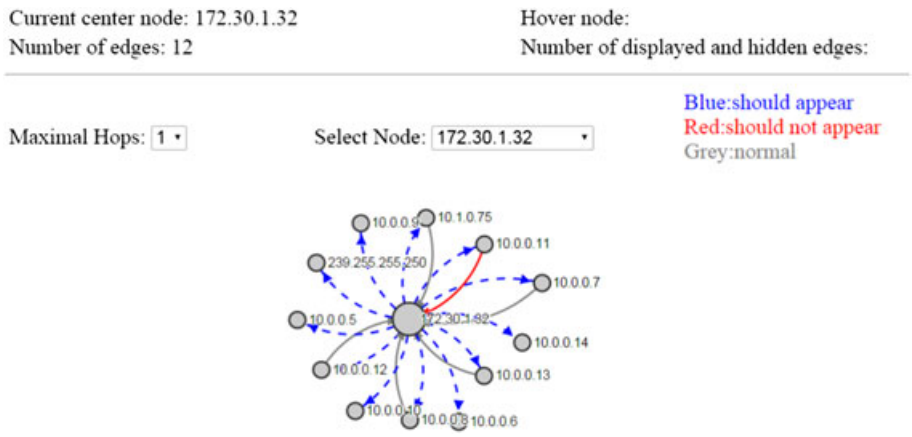



Figure 16. A user workstation (172.30.1.32) as the center node of link anomaly graph in the period of April 13–15. The dotted blue lines suggest possible Type-I link anomalies, and the node may have connectivity issue with many other nodes such as 10.1.0.75 and 10.0.0.11.

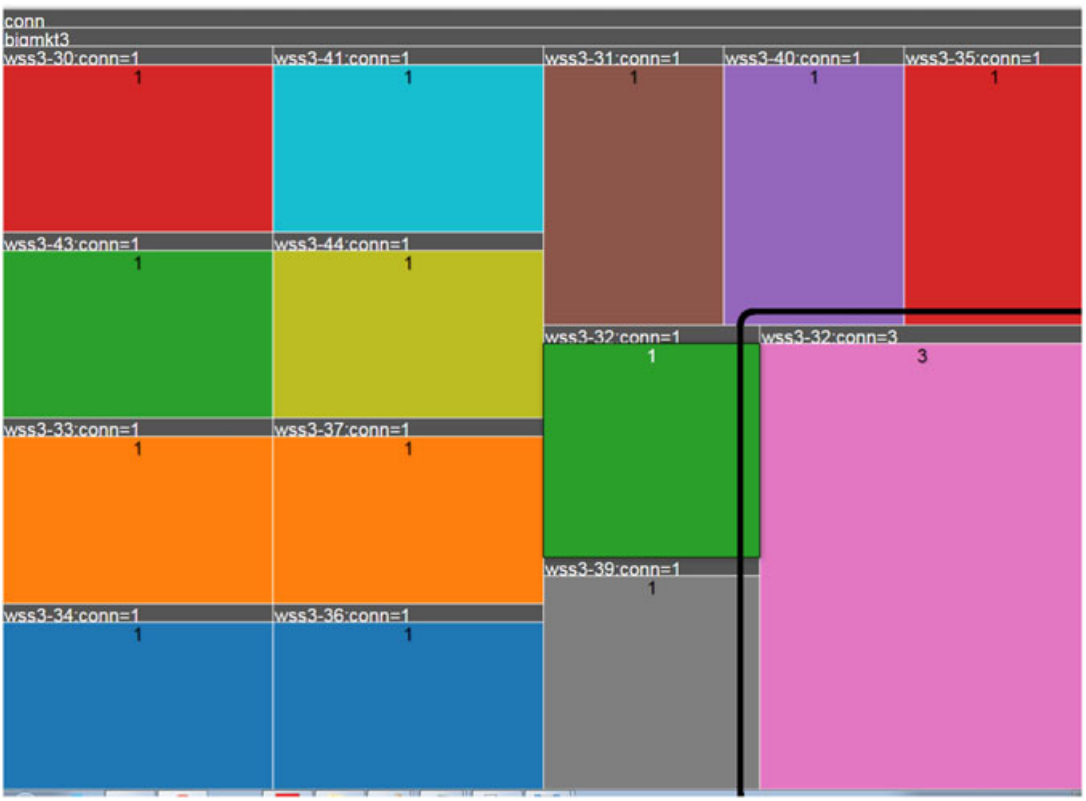



Figure 17. The treemap view of machines' *conn* (connect ability) property during the period of April 13–15 within *bigmkt3* subnet. The rectangle representing the problem workstation *wss3-32* (172.30.1.32) has larger size suggesting anomaly.



DOI: 10.1002/nem

server *web03*'s *conn* status value is 3 (problem). Verified by both the Gantt chart and the treemap view, Adam can draw a conclusion that there is something wrong with *web03*'s connectivity, which causes the anomaly on the network.

After finding the server, which causes the network anomaly, Adam has to further investigate the reason for the machine's anomaly. In link anomaly graphs, Adam selects 172.30.0.4 (*web03*) as the center node, and all connections made with this node are displayed in the graph, as shown in Figure 14. Adam notices that there are a few nodes such as 172.255.255.255, 172.0.0.1, and 10.206.5.2 that should connect to 172.30.0.4 according to link anomaly detection algorithms but actually are not connected to the server (dotted edges shown in blue color). On the other hand, there are outside machines such as 10.156.165.212, 10.13.77.49, and 10.57.105.222 that should not connect with 172.30.0.4 but in fact are connected (solid edges shown in red color).

Adam also checks a few 172.30.0.4-related connections' attributes in the parallel coordinates during the reported period. The connection information is shown in Figure 15. Adam finds that some suspicious machines such as 10.156.165.212 have unusually large number of connections to 172.30.0.4 during the period, as the *RCount* (frequency of the connection appearing in the selected period) value is significantly higher compared with most of other connections. All of the supporting evidences from different angles suggest to Adam that the event is not caused by normal functional failure of the sever itself, but rather, server *web03* is under DoS attack that causes the connectivity anomaly on the network.

### 5.3. *Functional failure*

Adam receives a user trouble report that a user's computer (IP address 172.30.1.32) does not have connection access to certain machines from April 13 to 15. Not knowing what may be the reason that causes this problem, Adam opens NAVA tool and moves the bottom time slider bar to match the time

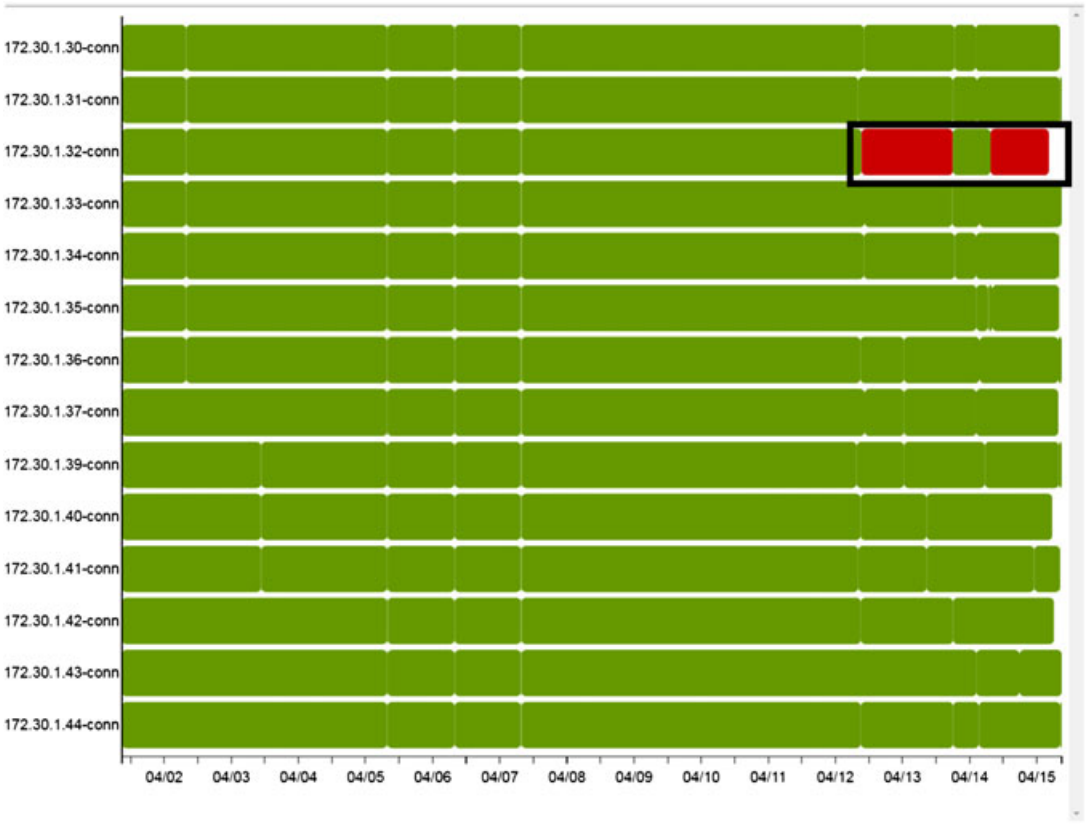



Figure 18. The Gantt chart view of machines' *conn* (connect ability) property over the entire logging period. The workstation 172.30.1.32 shows red meaning there is connectivity problem during the period of April 13–15.



DOI: 10.1002/nem

range of the user's trouble report. In the link anomaly graph (Figure 16), Adam chooses 172.30.1.32 as the center node and sets maximal hops to 1, so all connections with 172.30.1.32 are displayed in the graph. Adam finds that the user's machine does not connect to many machines during the time period, as most of the edges from 172.30.1.32 are dotted blue lines, which indicates Type-I anomaly, that is, this machine should connect to those machines, but in fact, it does not. Adam concludes that there must be something wrong with this machine. To find out the exact reason, Adam decides to take advantage of the treemaps and Gantt chart views.

Adam enters the user's IP address 172.30.1.32 in the textbox, clicks the search button in the Selection Pool, and selects all connections appearing in the list. He then selects the *conn* attribute in attribute selection drop-down menu, and consequently, both the treemaps and Gantt chart views are automatically updated. As shown in Figure 17, the rectangle representing the workstation *wss3-32* (which refers to machine with IP address of 172.30.1.32) has the value of 3 in the scale of 1–4 and is significantly larger than those of other machines within the subnet *bigmkt3*. According to the treemap view, Adam reasons that the machine's bad connection ability causes this anomaly.

To further confirm his assumption, Adam turns to the Gantt chart view (Figure 18) and traces the selected machine (172.30.1.32)'s *conn* property and its changing trend over the entire logging period. Adam notices that during the reported problem period, the workstation 172.30.1.32's *conn* status color is red while those of other machines are green (normal). After linking visualization and investigation, Adam draws the conclusion that the machine 172.30.1.32's anomalous connection ability is due to functional failure.

### 5.4. *Port scan*

On April 13, Adam gets reports from users who complain their computers are not working properly. The user trouble tickets are quite vague and confusing; therefore, Adam turns to NAVA trying to find out which part of the network is going wrong and what are the reasons.

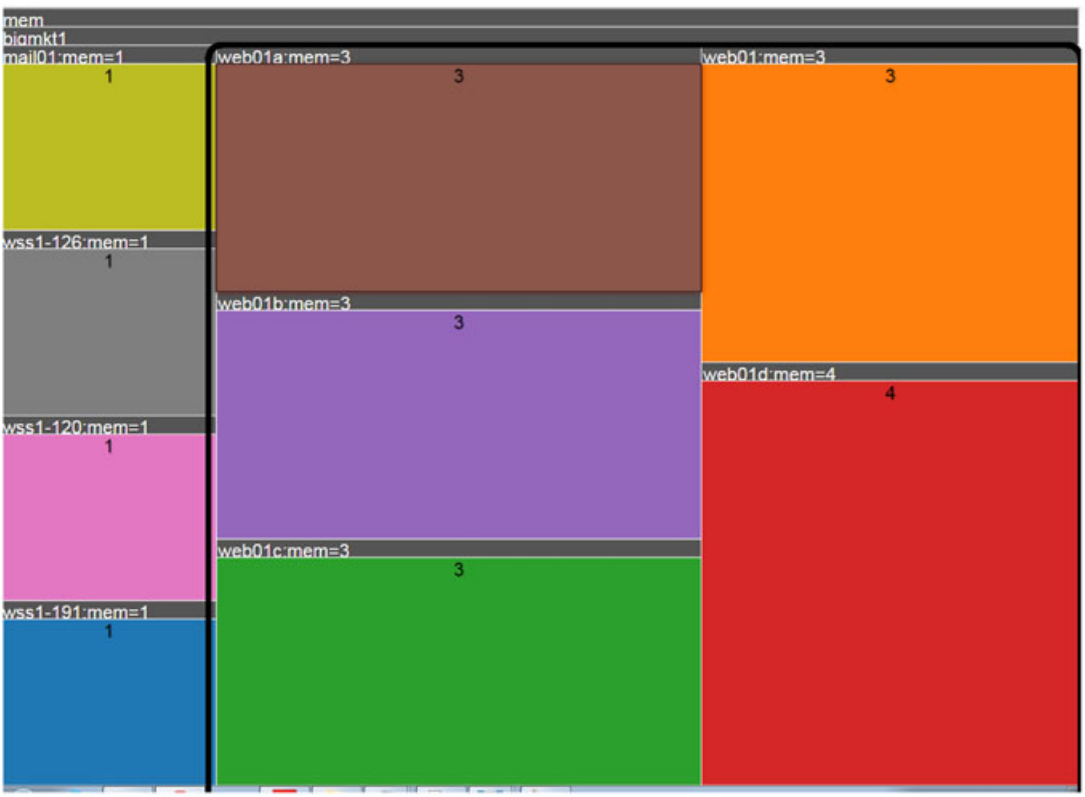



Figure 19. The treemap view of machines' *mem* (memory usage) on April 13, when several web servers ($web01*$) within *bigmkt1* network exhibit anomalous behavior.



DOI: 10.1002/nem

Having no clue what causes this anomaly, Adam refers to the treemap view first. He moves the time bar to the reported period and selects connections in the Selection Pool. Adam checks each of the selected machines' property respectively. He chooses the *mem* (memory usage) property for the treemap view (Figure 19). Adam notices that there are several servers, such as *web01*, *web01a*, and *web01b*, whose rectangle areas are significantly bigger than those of other machines within the *bigmkt1* site. While other normal machines' *mem* values are 1, these servers' *mem* values are 3 (problem) and 4 (does not send status). To investigate his assumption that these server might be under attack, Adam goes through the Gantt chart view and gets similar results. These servers' status values are yellow (warning), red (problem), or blue (no status) in the selected period.

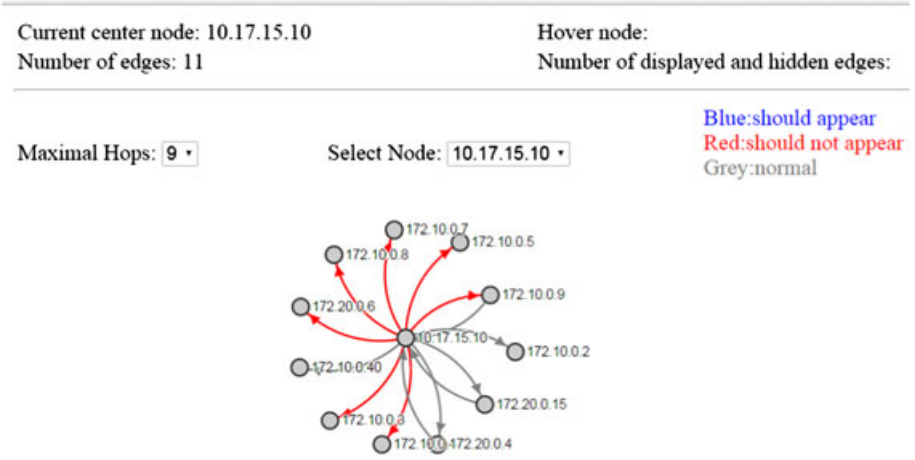



Figure 20. In the link anomaly view on April 13, a suspect machine 10.17.15.10 is identified via Type-II link anomaly suggested by red edges connecting to the anomalous web servers 172.10.0.5 (web01a) and 172.10.0.9 (web01b).

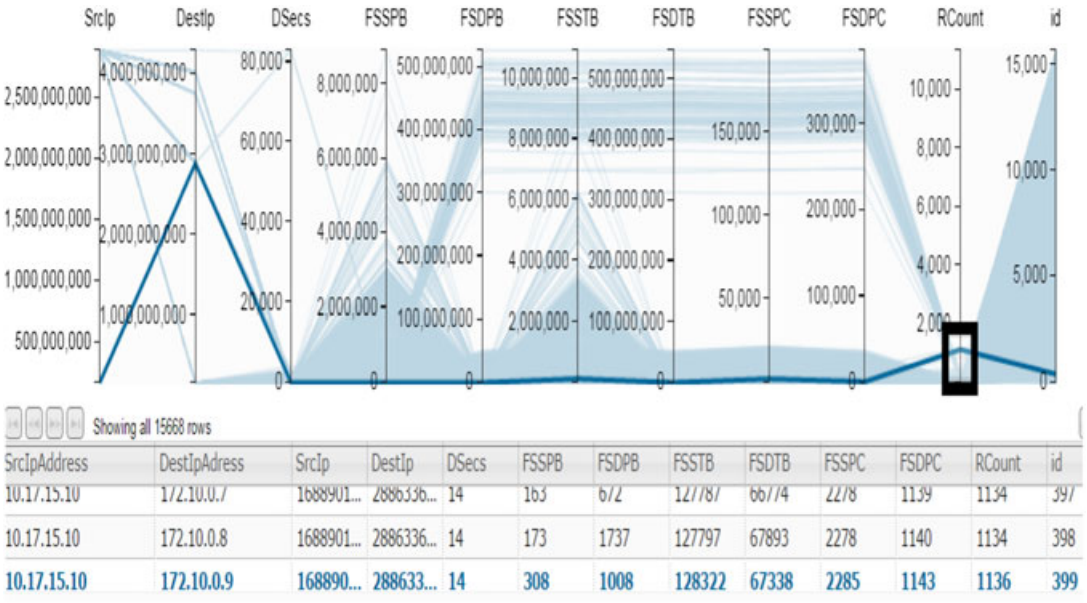



Figure 21. In the parallel coordinates view on April 13, source and destination pair 10.17.15.10→172.10.0.9's *RCount* is much higher than those of other machines suggesting the malicious machine 10.17.15.10 conducts port scan attack on the web servers in the network.



DOI: 10.1002/nem

To further investigate the reasons that lead to these anomalies, Adam takes advantage of the link anomaly graph view and the parallel coordinates view. In the link anomaly view (Figure 20), Adam identifies there is a node (10.17.15.10) that should not connect to the servers 172.10.0.5 (*web01a*) and 172.10.0.9 (*web01b*), but actually, there are connections established among them, suggested by the red connection lines. Based on the information, Adam infers that the anomalies may be caused by the node 10.17.15.10. He tries to verify his hypothesis in parallel coordinates (Figure 21). Adam notices that these connections appear more often during the selected period compared with other connections. The source and destination pairs 10.17.15.10 → 172.10.0.9 appear as high as 1136 times (*RCount* value). Based on all the information learned from the four views, Adam concludes that on April 13, exterior machine 10.17.15.10 has conducted a port scan attack on the Big Marketing network. Servers such as *web01*, *web01a*, and *web01b* are all under attack. Port scan is another type of anomaly that happens often on the network. It is commonly used by hackers to probe secured system for available access points in order to gain access to the system or start other types of attacks including DoS attack.

### 5.5. *DDoS attack*

On April 14, from 6 to 9 AM, Adam receives a phone call stating that one of the important exterior web sites (the IP address of which is 10.1.0.100) becomes unresponsive. He tries to visit the web site himself, and he also has no access. To find out the exact reason for this anomaly, he goes to NAVA tool and moves the bottom time bar to the related time period and most recent data uploaded to the database is loaded into the tool. Adam first examines the link anomaly graph, and he chooses the web site as

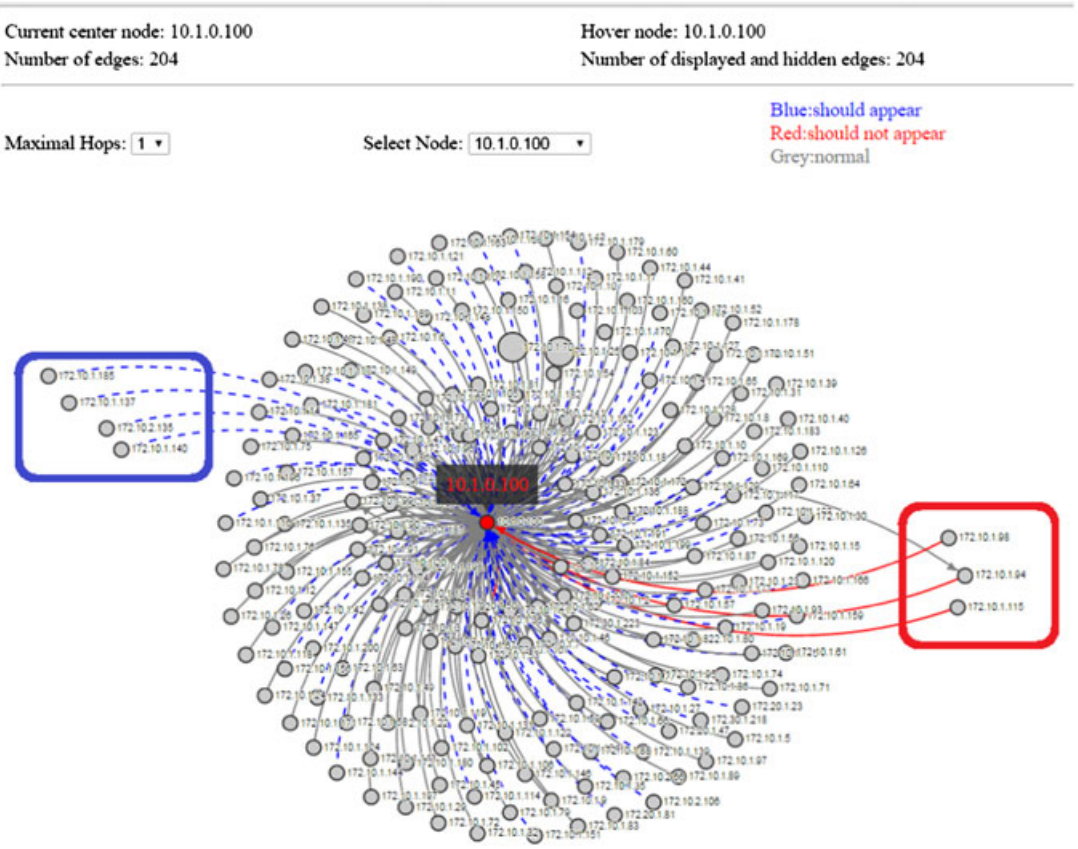



Figure 22. In the link anomaly graph on April 14, from 6 to 9 AM, machines 172.10.1.98, 172.10.1.94, and 172.10.1.115 launch DoS attacks on the web server 10.1.0.100 as suggested by Type-II link anomalies shown in red edges. Consequently, many users, suggested by Type-I link anomalies shown in blue edges, are unable to access the web site.



DOI: 10.1002/nem

the center node. Adam notices that there are some anomalous connections in this graph, as shown in Figure 22.

A few machines with IP addresses such as 172.10.1.98, 172.10.1.94, and 172.10.1.115 should not connect to the web site (10.1.0.100) but, actually, are connected, shown by red solid edges. Many other machines, such as 172.10.1.185, 172.10.1.137, 172.10.1.140, and 172.10.2.135, should connect with the web site in that period, but in fact, they do not, shown by dotted blue edges. Adam hypothesizes that the unresponsive webserver (10.1.0.100) might be under DDoS attack by the nodes of red lines (172.10.1.98, 172.10.1.94, and 172.10.1.115) so that many of his users' machines of blue lines are unable to connect. To confirm his hypothesis, Adam goes to the parallel coordinates view, and he verifies that the *Rcount* attributes of these pairs of source and destination IP addresses are all unusually high compared with other connections.

### 5.6. *VC12 data description*

We further evaluate the tool using another publicly available dataset as case study. The VAST 2012 Mini Challenge 2 contains 40-hour (from April 5 2012,17:00:00 to April 7 2012,09:00:00) network security data. This dataset focuses on issues involving operational concerns of the global enterprise: Bank of Money (BoM). BoM has many offices of various size across the BankWorld. Each of these offices has many computers active throughout the day. To illustrate the 'big data' challenge, BoM network has nearly one million computers, with approximately 4000 workstations and 1000 servers in each regional office. Network security data provided by this enterprise includes network flow and IDS logs. In our case study, we mainly take advantage of the network flow log and use only minimum information (i.e., who connects to whom).

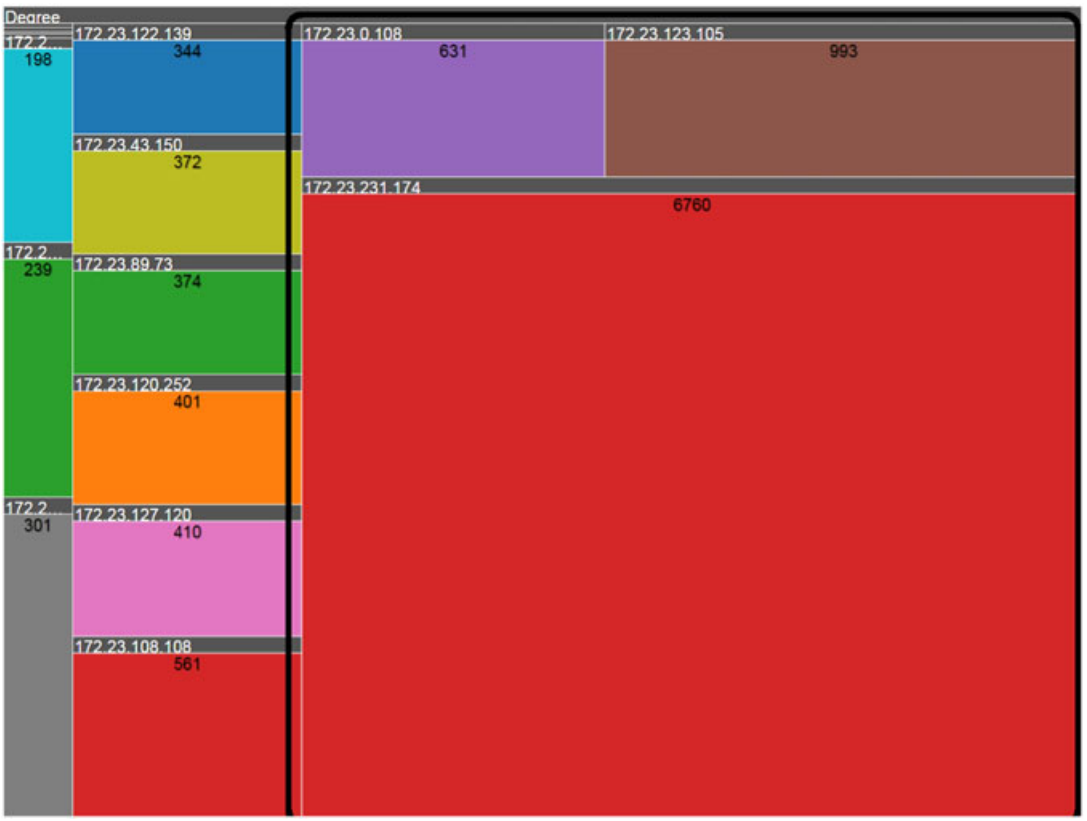



Figure 23. In the treemap view from 5 PM April 5 to 9 AM April 7, machines 172.23.231.174, 172.23.123.105, and 172.23.0.108 exhibit anomalous behavior with the connectivity *degrees* significantly larger than others.



DOI: 10.1002/nem

### 5.7. *Botnet*

On April 7, BoM network administrator, Adam, receives reports from the enterprise's staff that their machines have slowed down compared with prior days and have an excessive number of pop-ups, which contain ads. IP addresses of these reporting anomalous machines include 172.23.0.108, 172.23.123.105, and 172.23.231.174. With suspicion in mind that these machines may be infected with malware, Adam opens NAVA tool and sets the time slider bar to examine the log data during the problem time period. He types the aforementioned three machines' IP addresses, 172.23.0.108, 172.23.123.105, and 172.23.231.174, and clicks the 'search' button in the left panel. All connections involving those IP addresses are shown in the Selection Pool. Adam chooses these connections and then goes to the treemap view and the Gantt chart view trying to check the machines' degree and entropy values. In the treemap view (Figure 23), Adam finds that during the selected period of time, these three machines' rectangle sizes are indeed larger compared with other machines, with 631, 993, and 6760 number of connections.

Adam then observes these three machines' entropy changing trend in the Gantt chart view (Figure 24). He finds that the three machines' entropy values all have dramatic changing points at certain time. Based on information provided in the treemap view and the Gantt chart view, Adam learns that not only their connection quantities are comparatively greater but they have more dynamic connectivity situations. He infers that these machines could be port scanned by other machines. To find out the source machines that may cause these anomalies, Adam plans to utilize the other two views in the tool.

In the link anomaly graph, Adam chooses the IP address 172.23.231.174 as the current center node and sets the maximal hops to observe the subgraph during the investigation time period. He

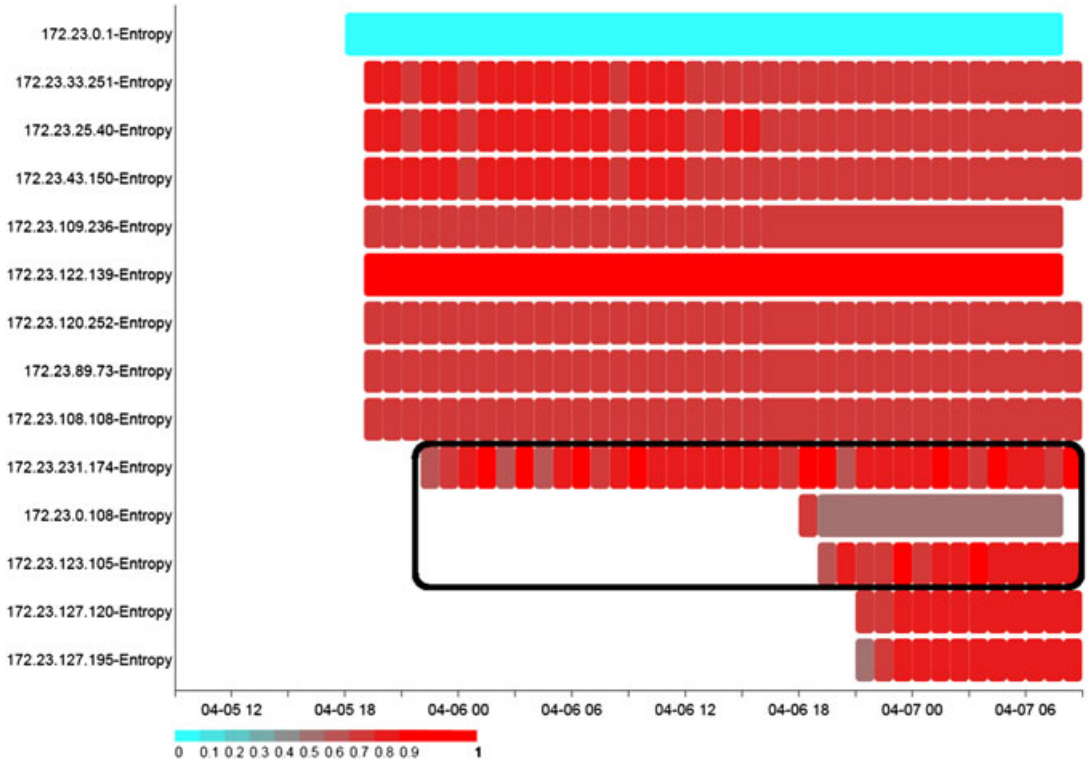



Figure 24. In the Gantt chart view of the same time period from 5 PM April 5 to 9 AM April 7, machines 172.23.231.174, 172.23.123.105, and 172.23.0.108 are not only having significantly larger connectivity degrees but also showing higher dynamics, as suggested by the timeline trend of their entropy values. For example, the entropy value of 172.23.0.108 changes dramatically between colors at 7 PM on April 6, indicating there is anomaly with this machine at that time.



DOI: 10.1002/nem

observes the node 172.23.231.174 connects with 10 different web sites, whose IP addresses range from 10.32.5.50 to 10.32.5.59. The red connection links indicate Type-II anomalies that these connections should not appear in the first place, as shown in Figure 25. Consequently, Adam also selects nodes 172.23.0.108 and 172.23.123.105 as the center nodes, and he gets the same results as machine 172.23.231.174, that is, machines 172.23.0.108 and 172.23.123.105 are also connected with the aforementioned 10 IPs.

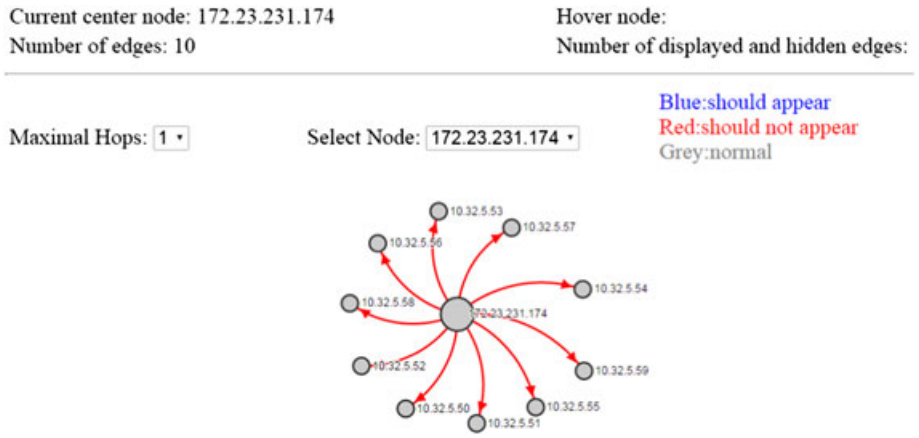



Figure 25. During the whole selected period, machine 172.23.0.108 only connects with 10 machines with IP addresses ranging from 10.32.5.50 to 10.32.5.59, and connection lines among them are all red, indicating these connections are anomalous.

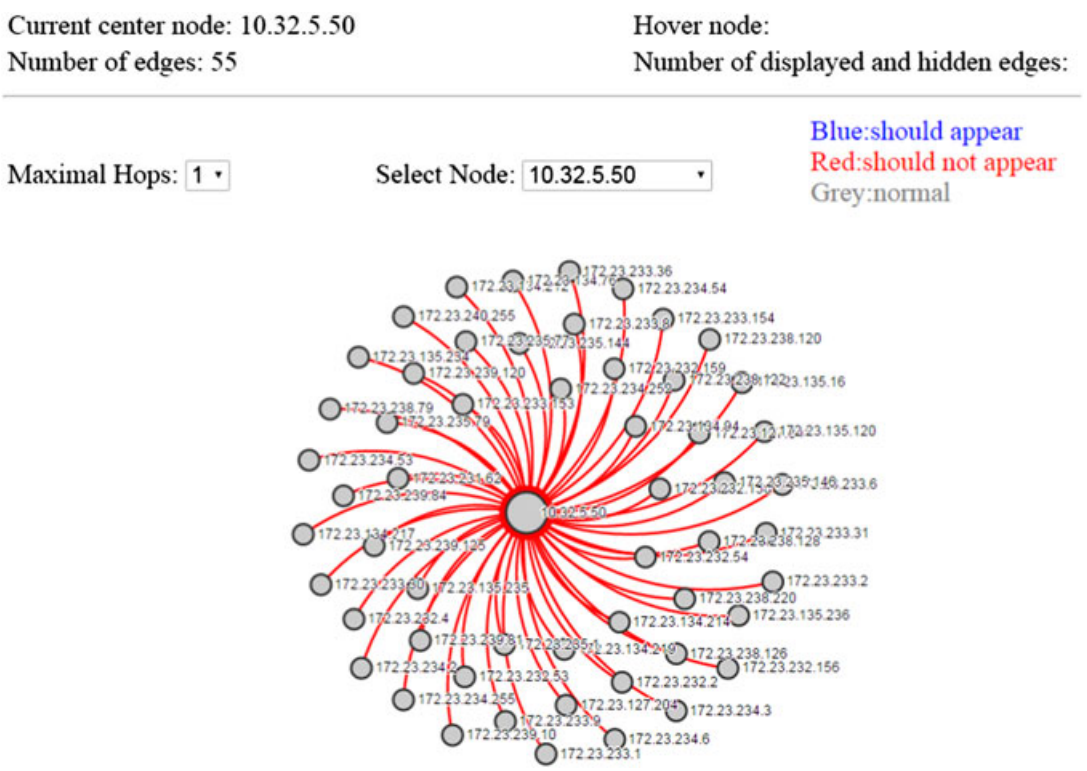
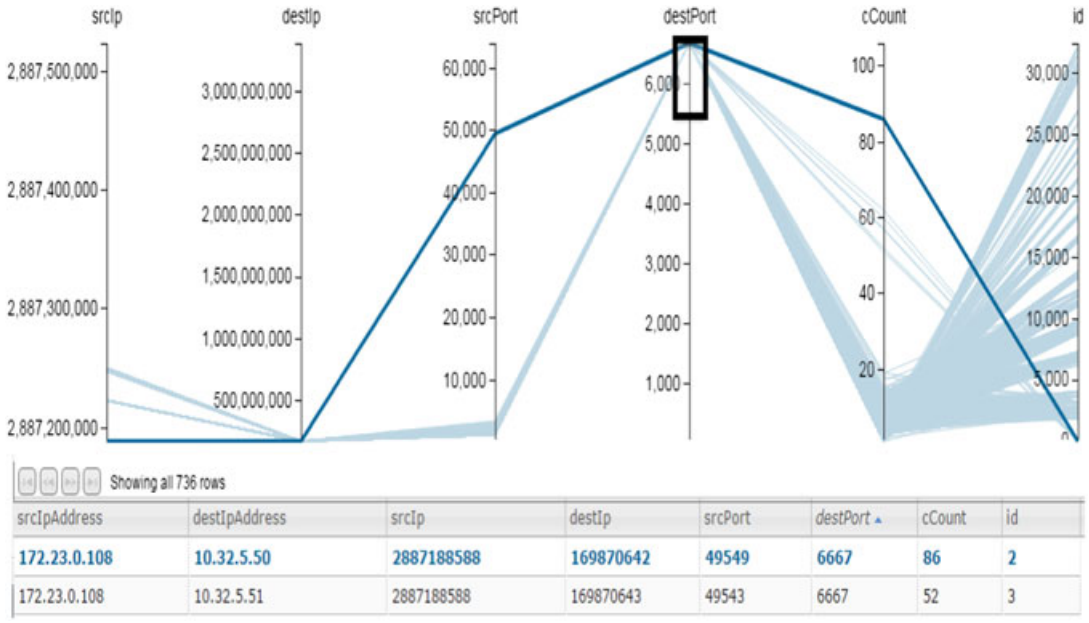



Figure 26. Link anomaly graph suggests one external web site 10.32.5.50 (the center node) has Type-II anomalous connections with many internal machines, possibly compromised and became part of a bonet.



DOI: 10.1002/nem




Showing all 736 rows

| srcIpAddress | destIpAddress | srcIp | destIp | srcPort | destPort | cCount | id |
|---|---|---|---|---|---|---|---|
| 172.23.0.108 | 10.32.5.50 | 2887188588 | 169870642 | 49549 | 6667 | 86 | 2 |
| 172.23.0.108 | 10.32.5.51 | 2887188588 | 169870643 | 49543 | 6667 | 52 | 3 |

Figure 27. The parallel coordinates view examines the anomalous connections' destination ports are 6667 in association with Internet Relay Chat (IRC) traffic. This is strong evidence that the 10 external web sites (10.32.5.50–59) act as command and control centers to remotely control many internal machines managed by Adam, including 172.23.0.108, 172.23.231.174, and 172.23.123.175.

To explore the data more, Adam chooses one of the 10 IPs, that is, 10.32.5.50, as the current center node and sets the maximal hops to 1. He finds that in addition to the three machines referred earlier, the external web site 10.32.5.50 also has connections with other internal machines (172.23.*) in his managed network. The red lines indicate those connections should not be established (Figure 26). Adam repeats the step for the nine web sites from 10.32.5.51 to 10.32.5.59, and he gets the same result. Based on all of the information, Adam concludes that anomalies within his network are caused by these 10 sites.

In order to find the underlying cause, Adam further refers to the parallel coordinates view and tries to get the detailed information of the 30 connection pairs (between 172.23.0.108, 172.23.231.174, 172.23.123.175, and 10.32.5.50–59). He finds out that the destination ports of these connections are 6667, which reminds him of Internet Relay Chat (IRC) traffic (Figure 27). He also learns that these 30 source/destination pairs are connected more often according to the *cCount* attribute in the parallel coordinates. It turns out that the 10 web sites 10.32.5.50–59 are botnet servers controlled by botnet-masters, who use the servers to distribute malware and to compromise the machines. The servers act as the command and control centers to send commands to the compromised machines within Adam's network, including machines 172.23.0.108, 172.23.123.105, and 172.23.231.174.

## 6. CONCLUSION

Network log data has become not only increasingly larger in data volume but also highly dynamic in nature. To investigate and analyze the anomalous events has become computationally intractable and a daunting task for modern day network operators and managers. In this paper, we study how various visualization methods may be used to address different aspects of network management, and when these views are linked and combined, what information gains might be particularly valuable for network anomaly analysis tasks. We design and implement a general, interactive, web-based network anomaly visualization and analytic (NAVA) tool. The tool utilizes visualizations such as link anomaly graphs, parallel coordinates, treemaps, and Gantt charts, combined with spatial–temporal analytic algorithms for dynamic network, such as link anomaly detection based on the graph topology evolution.



DOI: 10.1002/nem

Through extensive case studies over public cyber-security datasets, we demonstrate that the NAVA tool provides a time-efficient alternative for system administrators and network managers to analyze their network and system log data and can effectively identify a variety of network anomalies and analyze their underlying causes.

## ACKNOWLEDGEMENTS

This work is supported in part by US Air Force SFFP fellowship and VFRP Extension Grant program (R926249).

## REFERENCES

1. García-Teodoro P, Díaz-Verdejo J, Maciá-Fernández G, Vázquez E. Anomaly-based network intrusion detection: techniques, systems and challenges. *Computers & Security* 2009 February-March; **28**(1-2): 18–28.
2. Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. *ACM Computing Surveys (CSUR)* 2009 July; **41**(3, article 15): 1–58.
3. Sommer R, Paxson V. Outside the closed world: On using machine learning for network intrusion detection. In *IEEE Symposium On Security And Privacy,* Oakland, CA, 2010 May 16-19; 305–316.
4. de Oliveira MC F, Levkowitz H. From visual data exploration to visual data mining: a survey. *IEEE Transactions on Visualization and Computer Graphics* 2003 July-Sept; **9**(3): 378–394.
5. Marty R. *Applied Security Visualization*, Addison Wesley Professional, 2008. ISBN: 9780321510105.
6. Livnat Y, Agutter J, Moon S, Foresti S. Visual correlation for situational awareness, In *IEEE Symposium On Information Visualization (Infovis'05),* Minneapolis, MN, Unknown Month 23; 95–102.
7. McRee R. Security visualization: what you don't see can hurt you. *Information Systems Security Association (ISSA) Journal* June 2008; **6**(6): 38–41.
8. Schwartz M, Liebrock L M. A term distribution visualization approach to digital forensic string search. In *Proceedings of 5th International Workshop On Visualization For Computer Security (Vizsec'08),* Cambridge, MA, 15 September 2008; 36–43.
9. Teoh S T, Ma K-L, Wu S F, Jankun-Kelly TJ. Detecting flaws and intruders with visual data analysis. *IEEE Computer Graphics and Applications* September/October 2004; **24**(5): 27–35.
10. Bostock M, Ogievetsky V, Heer J. $d^3$ data-driven documents. *IEEE Transactions on Visualization and Computer Graphics* Dec.2011; **17**(12): 2301–2309.
11. Lakhina A, Crovella M, Diot C. Characterization of network-wide anomalies in traffic flows. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement ser.IMC '04,* New York, NY, USA: ACM, 2004; 201–206.
12. Rattigan M J, Jensen D. The case for anomalous link discovery. *ACM SIGKDD Explorations Newsletter* 2005; **7**(2): 41–47.
13. Takahashi T, Tomioka R, Yamanishi K. Discovering emerging topics in social streams via link-anomaly detection. *IEEE Transactions on Knowledge and Data Engineering* Jan. 2014; **26**(1): 120–130.
14. Liao Q, Striegel A, Chawla N. Visualizing graph dynamics and similarity for enterprise network security and management. In *ACM Proceeding of the 7th International Symposium on Visualization for Cyber Security (VizSec'10), in conjunction with RAID'10,* Ottawa, Ontario, Canada, 14 September 2010; 34–46.
15. Komlodi A, Goodall J R, Lutters W G. An information visualization framework for intrusion detection. In *Proceeding of ACM Human Factors In Computing Systems (chi'04), Extended Abstracts,* Vienna, Austria, 2004; 1743–1746.
16. D'Amico A D, Goodall J R, Tesone D R, Kopylec J K. Visual discovery in computer network defense. *IEEE Computer Graphics and Applications* 2007; **27**(5): 20–27.
17. Tolle J, Niggemann O. Supporting intrusion detection by graph clustering and graph drawing. In *Proceedings of Raid 2000 Third International Workshop on the Recent Advances in Intrusion Detection*: Toulouse, France, April 2; 51–62.
18. Liao Q, Striegel A. Intelligent network management using graph differential anomaly visualization. In *2012 IEEE Network Operations and Management Symposium (NOMS),* Maui, Hawaii, USA, 2012; 1008–1014.
19. Kumar G, Garland M. Visual exploration of complex time-varying graphs. *IEEE Transactions on Visualization and Computer Graphics* September 2006; **12**(5): 805–812.
20. Wilson J M. Gantt charts: a centenary appreciation. *European Journal of Operational Research* 2003; **149**(2): 430 –437. http://www.sciencedirect.com/science/article/pii/S0377221702007695.
21. Jia HZ, Fuh JYH, Nee AYC, Zhang YF. Integration of genetic algorithm and gantt chart for job shop scheduling in distributed manufacturing systems. *Computers & Industrial Engineering* 2007; **53**(2): 313 –320.
22. Abello J, Kobourov S G, Yusufov R. Visualizing large graphs with compound-fisheye views and treemaps. In *Proceedings of the 12th International Conference on Graph Drawing,* New York, NY, Oct.2 2004 Sep.29; 431–441.
23. Yin X, Yurcik W, Treaster M, Li Y, Lakkaraju K. Visflowconnect: Netflow visualizations of link relationships for security situational awareness. In *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security (VIZSEC/DMSEC '04),* Washington DC, USA, 2004; 26–34.
24. Plonka D. Flowscan: A network traffic flow reporting and visualization tool. In *In USENIX LISA,* New Orleans, Louisiana, USA, 2000; 305–317.
25. Celenk M, Conley T, Willis J, Graham J. Predictive network anomaly detection and visualization. *IEEE Transactions on Information Forensics and Security* 2010; **5**(2): 288–299.



DOI: 10.1002/nem

26. Zhao Y, Zhou FangFang, Fan XiaoPing, Liang X, Liu YongGang. Idsradar: a real-time visualization framework for ids alerts. *Science China Information Sciences* August 2013; **56**(8): 1–12.
27. Zhang T, Liao Q, Shi L. Bridging the gap of network management and anomaly detection through interactive visualization. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis),* Yokohama, Japan, 4-7 March 2014; 253–257.
28. Cook K, Grinstein G, Whiting M. The vast challenge: history, scope, and outcomes: an introduction to the special issue. *Information Visualization* October 2014; **13**(4): 301–312.
29. Waldner M, Schmalstieg D. Collaborative information linking: Bridging knowledge gaps between users by linking across applications. In *IEEE Pacific Visualization Symposium (PacificVis),* Hong Kong, China, 2011; 115–122.
30. Han K J, Zhang T, Liao Q. Data stream mining based dynamic link anomaly analysis using paired sliding time window dataApproved for Public Release; Distribution Unlimited, U.S. Air Force Research Laboratory (AFRL). *Technical Report 88ABW-2014-5054*, 2014.
31. Fire M, Tenenboim L, Lesser O, Puzis R, Rokach L, Elovici Y. Link prediction in social networks using computationally efficient topological features. In *Socialcom/Passat IEEE,* Boston, MA, 2011; 73–80.
32. Katz L. A new status index derived from sociometric analysis. *Psychometrika* March 1953; **18**(1): 39–43.



DOI: 10.1002/nem