Detecting Flaws and Intruders with Visual Data Analysis

Soon Tee Teoh, Kwan-Liu Ma, and Soon Felix Wu
University of California, Davis

T.J. Jankun-Kelly
Mississippi State University

Keeping computer and network systems secure and stable requires collecting vast amounts of data and analyzing how the systems perform dynamically. No matter how rigorous a system's design process, runtime factors can compromise performance. Even network protocols with strong theoretical bases can suffer security flaws and instability when deployed. Furthermore, few systems are designed with perfect security. Intrusion detection and response are thus important components of any computer system.

In general, computer security aims to prevent, detect, and respond to flaws and intrusions. Intrusion detection plays a critical role in system security because prevention methods, such as passwords and access control, often fail. Robust systems thus require an intrusion-detection mechanism to identify unauthorized use of the system that threatens resource availability, integrity, or confidentiality. Analyzing the detected intrusion also lets administrators take corrective or punitive action. Intrusion detection typically involves examining user activity logs for suspicious or anomalous behaviors that may indicate an attack (see the "Data Mining and Visualization for Network Security" sidebar on the next page).

The task of sifting through large amounts of data to find useful information spawned the field of data mining. Most data mining approaches are based on machine-learning techniques, numerical analysis, or statistical modeling. They use human interaction and visualization only minimally. Such automatic methods can miss some important features of the data. Incorporating human perception into the data mining process through interactive visualization can help us better understand the complex behaviors of computer network systems. This article describes visual-analytics-based solutions and outlines a visual exploration process for log analysis. Three log-file analysis applications demonstrate our approach's effectiveness in discovering flaws and intruders in network systems.

Visual analytics

This issue's guest editors describe visual analytics as a "contemporary and proven approach to combine the art of human intuition and the science of mathematical deduction to directly perceive patterns and derive knowledge and insight from them" (p. 20). Visual analytics helps users discover new and useful knowledge in data, leading to improvements in system security and stability. Over the past decade, various applications have used visual analytics. Ahlberg and Shneiderman,[1] for example, promote visual-based methods to support information seeking because humans recognize features in visual displays and identify anomalies by recalling related images. According to Girardin,[2] humans perceive even unexpected features. We believe this is visual analytics' biggest advantage over primarily algorithmic data mining methods.

By incorporating human perception into the data-mining process, researchers can detect patterns in data missed by traditional automatic data mining methods.

Our work is based on the premise that we can glean valuable knowledge from large data sets—the same premise behind knowledge discovery. Most data mining methods use algorithms or statistical or mathematical models, often adapted from machine-learning techniques. Although these methods have successfully performed tasks such as classification, regression, and clustering, each method discovers only limited knowledge.

Because visual data mining differs in essence from automated methods, visual methods can discover information that complements the knowledge found by more commonly used statistical approaches. This is particularly useful when users want to explore the data only to learn about it—that is, they don't know exactly what they need to discover from the data. Many real-world problems, including intrusion detection, fit into this category.

Our visual analytics process is as follows. Starting with a large data set (in our case a log file), we design an appropriate visual representation and an intuitive interaction method with which the user explores the data. The design of the visualization and interaction method is critical. Good visual metaphors bring out interesting

 0272-1716/04/$20.00 © 2004 IEEE

Data Mining and Visualization for Network Security

Security experts introduced data mining methods into intrusion detection when they realized that signature-based methods were too rigid to discover novel attacks. These intrusion-detection approaches use different data mining methods and analyze different data. Schultz et al.,[1] for example, use Naive Bayes algorithms to detect malicious executables. Ghosh and Schwartzbard,[2] on the other hand, use neural networks on the DARPA network connections data set.

Signature-based methods also require time-consuming human input. To solve this problem, Lee et al.[3] use data-mining methods to learn rules to accurately capture behavior from network connection and host session features.

Jiang et al.[4] present a pattern-extraction algorithm and a method to compare the extracted patterns to find intra- and interpattern mismatches. Mahoney and Chan[5] apply association rules (another data mining technique) to intrusion detection. The rich data mining techniques for finding frequent sequences are also helpful for computer security. Michael,[6] for example, uses suffix trees to find frequently occurring sequences of system calls.

Much less work has applied visualization to computer security. Examples include Erbacher et al.'s[7] use of glyphs to visualize intrusion-detection data and the Tudumi[8] visualization system designed to monitor and audit computer logs to help detect anomalous user activities.

Visualization-based data mining methods are also scarce. One example is PBC,[9] a visual classification tool. Because classification can be applied to intrusion detection, this visualization tool is particularly relevant to computer and network security. Furthermore, PBC successfully incorporates visualization and machine-learning techniques to improve data mining.

References

1. M.G. Schultz et al., "Data Mining Methods for Detection of New Malicious Executables," *Proc. IEEE Symp. Security and Privacy*, IEEE CS Press, 2001, pp. 38-49.
2. A.K. Ghosh and A. Schwartzbard, "A Study in Using Neural Networks for Anomaly and Misuse Detection," *Proc. 8th Usenix Security Symp.*, Usenix Assoc., 1999, pp. 141-152.
3. W. Lee, S.J. Stolfo, and K.W. Mok, "A Data Mining Framework for Building Intrusion Detection Models," *Proc. IEEE Symp. Security and Privacy*, IEEE CS Press, 1999, pp. 120-132.
4. N. Jiang, K. Hua, and S. Sheu, "Considering Both Intrapattern and Interpattern Anomalies for Intrusion Detection," *Proc. IEEE Int'l Conf. Data Mining* (ICDM 02), IEEE CS Press, 2002, pp. 637-640.
5. M.V. Mahoney and P.K. Chan, "Learning Rules for Anomaly Detection of Hostile Network Traffic," *Proc. 3rd IEEE Int'l Conf. Data Mining* (ICDM 03), IEEE CS Press, 2003, pp. 601-604.
6. C.C. Michael, "Finding the Vocabulary of Program Behavior Data for Anomaly Detection," *Proc. 3rd DARPA Information Survivability Conf. and Exposition (DISCEX 03)*, IEEE CS Press, 2003, pp. 152-161.
7. R.F. Erbacher, K.L. Walker, and D.A. Fincke, "Intrusion and Misuse Detection in Large-Scale Systems," *IEEE Computer Graphics and Applications*, vol. 22, no. 1, Jan./Feb. 2002, pp. 38-48.
8. T. Takada and H. Koike, "Tudumi: Information Visualization System for Monitoring and Auditing Computer Logs," *Proc. 6th Int'l Conf. Information Visualization*, IEEE CS Press, 2002, pp. 570-576.
9. M. Ankerst, M. Ester, and H.-P. Kriegel, "Toward an Effective Cooperation of the User and the Computer for Classification," *Proc. 6th Int'l Conf. Knowledge Discovery and Data Mining* (KDD 00), ACM Press, 2000, pp. 179-188.

data features in a logical way. Applications can use different visual metaphors, and each can lead to different discoveries, some of which will lead to a better understanding of the system. Other discoveries might reveal system architecture flaws and weaknesses or detect intruders.

There is a close collaboration between human and computer in our visual analytics process. On one hand, we use computations to process and project the data onto the display and to transform the data based on user input. On the other hand, the process uses human intelligence in two ways:

- in designing the visualization and interaction techniques; and
- in using the interactive visualization to discover, analyze, and draw conclusions from the data.

Visual-based applications and techniques

We've developed three visual-based methods for detecting network flaws and intrusions. All three applications require appropriate visualization tools and interaction techniques. Because the nature of the data, the task, and the desired knowledge are all different, the visual metaphors used in each of the applications also differ. In fact, we use different visualization tools in the same application.

Anomalous origin autonomous system changes

In the Internet, *autonomous systems* (ASs) control groups of hosts with consecutive IP addresses. This simplifies the packet routing problem to routing data between these larger entities (see the "Border Gateway Protocol" sidebar).

To maintain network stability, the AS associated with a group of IP addresses (their *origin AS*) must correspond to their true owner. Network analysts study the dynamics of origin AS changes (OASCs) to distinguish normal behavior from faults or suspicious activity. Earlier work presented a visual analysis tool to address this problem.[3] Here, we discuss an augmentation to the browsing portion of the tool and its application to a set of anomalous OASCs.

Visual exploration of the OASC data is a two-phase process. First, the user browses a sequence of visualizations summarizing the OASCs over time. If the user discovers an anomaly, the user can drill down into the data to determine the anomaly type and the ASs involved. In this step, the user is essentially performing visual pattern matching, using the visual system to separate normal from abnormal behavior. This visual classification is based on the rendering of OASCs for a given date.

Figure 1 (p. 30) demonstrates our rendering method. We map each IP address to a pixel using a quadtree decomposition, iteratively mapping pairs of bits from the

32-bit IP address. We map ASs along the four edges of the display area, and represent an OASC by a colored line connecting the previous AS owner to the new owner (O-type events have no previous owner). The color designates the type of change.

The original browsing tool successfully separated normal from abnormal behavior in the OASC data, as Figure 2 shows. However, occlusion occasionally masked some events. To rectify this issue, we developed a new browser that displays one image for each change type and one image for all changes together. It also shows the previous and next date's events. A focus + context radial layout[4] manages screen real estate, arranging the images in a circle around a larger focal image, as Figure 3 shows. The new layout solves the old browser's line-occlusion problem by decomposing event types while retaining the same mode of rapid, iterative exploration. The user can select the previous or next date's image or one of the eight event type images as the new focus. To assist in the analysis, we color the circle's sectors according to event type, using white for the combination image.

We can use the new event browser in several ways. Our previous study, for example, noted several sequential CSM/CMS events (a CSM event is a C-type—that is, when an AS claims ownership of another AS's prefix—change from a single origin AS to multiple ASs; a CMS event is a C-type change from multiple ASs to a single origin AS; see the "Border Gateway Protocol" sidebar for a detailed explanation of these OASC event types), indicating a misconfigured router and its subsequent correction. To determine how common these paired events are, a user could select either the CSM or CMS event and then explore the data (as in Figure 3). Comparing the previous and next day's images for the given type (at the 5 or 7 o'clock positions around the circle) against the image for its complementary type for the current date (at either 10 or 11 o'clock) reveals paired events. This analysis shows that the events are common, occurring almost once a month over the 480 days sampled.

As a further example, a user browsing the data could note some interesting behavior on 31 March 2001, as Figure 4 shows. The browser shows several coordinated CMS (light yellow), CMM (red; a CMM is a C-type

Border Gateway Protocol

The Internet resembles a set of clusters, with each cluster representing an organization's network. These *autonomous systems* (ASs) manage traffic within AS clusters. To communicate between systems, routers on an AS edge use the Border Gateway Protocol (BGP).[1]

BGP assigns each AS a unique identifier and set of IP prefixes. These prefixes identify which subset of IP addresses correspond to hosts in the AS. An IP prefix consists of an IP address and a mask—for example, the IP prefix 128.120.0.0/16 represents all IP addresses sharing the same first 16 bits 128.120. Using BGP, edge routers communicate network reachability information to transmit packets. These BGP routes consist of the destination IP prefix and a list of the ASs through which data will be routed to reach the destination. The BGP route 128.120.0.0/16: (7, 23, 92), for example, means that packets for the IP prefix 128.120.0.0/16 must sequentially pass through AS 7 and AS 23 before reaching their destination system (AS 92). The AS responsible for an IP prefix is known as the prefix's origin AS.

We visualize two aspects of BGP routing information in the article's main text:

- *Origin AS changes* (OASCs)—changes due to IP prefix ownership changes, valid network operation, network faults, or attacks.
- *BGP route changes*—changes of the BGP route to a particular IP prefix.

An OASC event consists of the IP prefix affected, a list of ASs associated with the change (generally the prefix's new origin AS), the change's date, and the change type. A change can narrow the mask of addresses an AS already owns (a B type) or another AS owns (an H type), claim ownership of another AS's prefix (C type), or claim ownership of an unowned prefix (O type). Further classification of the last two changes depends on whether a single AS or multiple ASs claim prefix ownership. Because usually only one AS should claim ownership, multiple origin AS conflicts may indicate faults or attacks. Some OASCs are complementary: A CMS event (a C-type change from multiple ASs to a single origin AS) could correct a CSM event (a C-type change from a single AS to multiple origin ASs). Eight OASC types (OS, OM, CSM, CMS, CMM, CSS, H, and B) exist. Our tool visualizes them all.

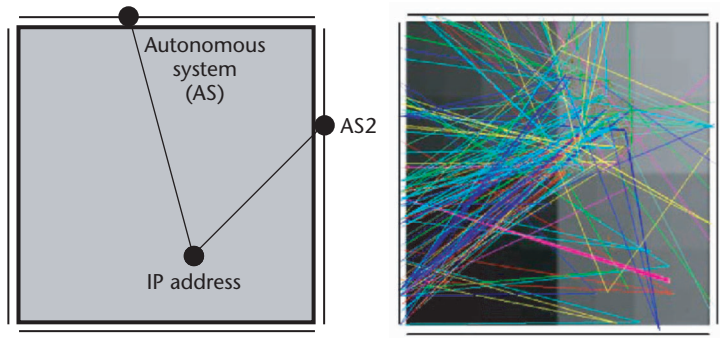
Our second visualization examines BGP route dynamics. As a host's availability changes, an AS along a route or the origin AS could become unavailable. Whenever routing information changes, edge routers exchange BGP announcements. An announcement is either a new BGP route or a withdrawal event such as 128.120.0.0/16: WD (in this example, the IP prefix 128.120.0.0/16 has become unavailable). A sequence of BGP announcements reveals Internet routing behavior.

Routing instabilities can seriously disrupt network traffic. Previous studies have identified three forms of instability events: slow convergence to a stable path,[2,3] oscillations between paths,[4] and repeats of the same path.[4] The visualization discussed in this work makes it more convenient to analyze these events.

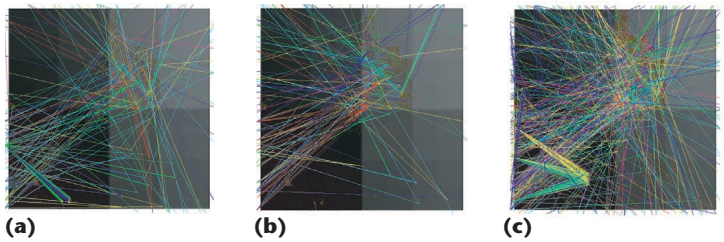
This article uses routing data from the Oregon Route Views server (http://www.antc.uoregon.edu/route-views). The data consists of BGP events from 480 days in 2000 and 2001.

References

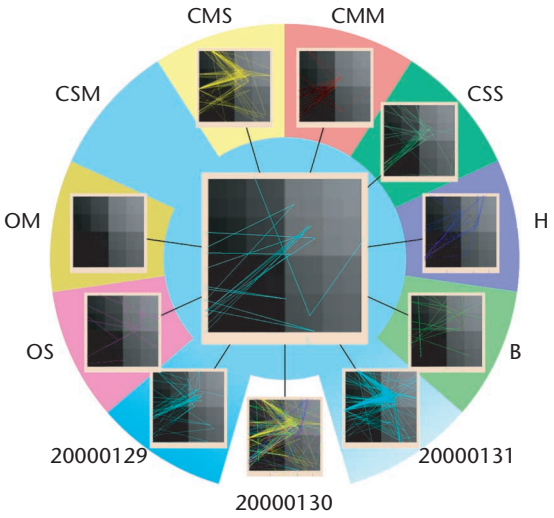
1. Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," IETF RFC 1771, 1995; http://www.rfc-editor.org/rfc/rfc1771.txt.
2. T. Griffin and G. Wilfong, "An Analysis of BGP Convergence Properties," *Proc. ACM Sigcomm*, ACM Press, 1999, pp. 277-288.
3. D. Pei et al., "Improving BGP Convergence through Consistency Assertions," *Proc. IEEE Infocom*, IEEE CS Press, 2002, pp. 902-911.
4. L. Gao and J. Rexford, "Stable Internet Routing without Global Coordination," *Proc. ACM Sigmetrics*, ACM Press, 2000, pp. 307-317.



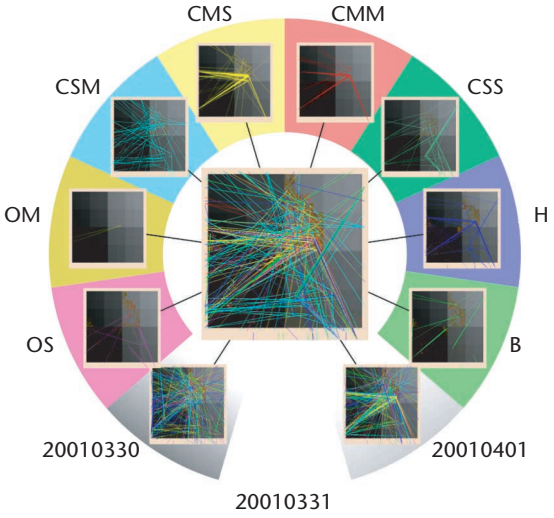
1 Origin autonomous system change (OASC) visualization. (a) A line connecting the affected IP prefix and ASs represents the event. (b) Actual data with color denoting event type.



2 OASCs from three different dates: (a and b) normal OASCs, (c) the high concentration of lines from the same ASs indicates an anomaly.

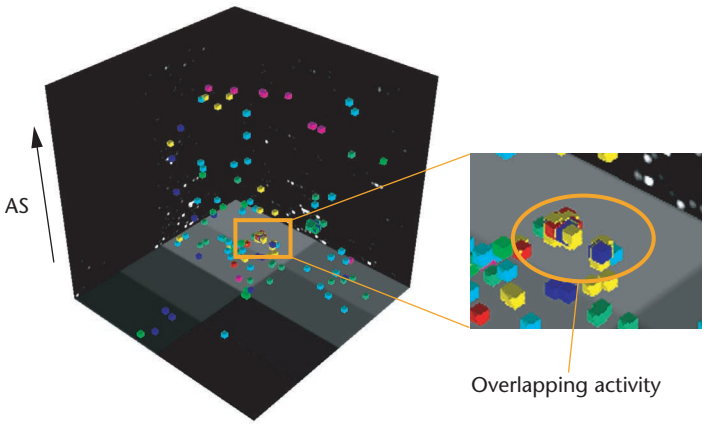


3 The new OASC browser. We arrange the eight change types radially around the center; another image shows all events and images for the neighboring dates.



4 Correlated anomalous OASCs on 31 March 2001.

change from multiple ASs to a multiple origin AS), and H (blue) events. The image for 1 April (light gray) shows a similar pattern consisting of different events: CMS, CSM (cyan), and CSS (green; a CSS is a C-type change from a single AS to another single origin AS). This coordination is difficult to see in the combined image (center) and might have been missed without the other images. With the browser, the user could step through the subsequent dates to determine the anomaly's length. The behavior is strongest in the first four days following the 31 March event, with smaller corrections continuing after that. To determine which ASs were involved, we use the drill-down module described in our previous work.[3]



5 Further investigation into the 31 March anomaly. The two clusters involve AS 703, one of the ASs at fault.

Figure 5 is a 3D representation of events using the IP address quadtree as its base, AS identifiers for height, and colored cubes for events. It shows two clusters of overlapping events, meaning that several OASCs are associated with the same IP address. Both clusters are in the same vertical plane, suggesting the same AS (AS 703) was involved. After using the analysis tool, the user concludes that AS 703 and AS 4740 claim different portions of AS 17561's addresses (the H events), causing conflicting correcting events over the rest of that day and the following days (the other events). It's easy for a user to notice correlated visual patterns without training; a fully machine-based method requires a significant number of ad hoc event signatures to do the same.

Routing instability

The Internet is a complex distributed system running on a large number of nodes using various protocols. Studying the Internet's operational behavior is fundamental to increasing its stability, robustness, and security. In the past, routing analysts have monitored and examined network behavior mainly by browsing the raw data or looking at simple plots of statistical analysis results. Our suite of visualization techniques improves understanding of Internet routing data. Using our system, we detected and analyzed a problem in the routing to Google's IP address.

We first observed the problem from the event shrubs[5] module, which shows each instability event as a circle on a time line. We colored segments of the circle according to the instability type matching the event, such as oscillation and repeat. The number of Border Gateway Protocol (BGP) update messages included in the event determines the circle's size. Obviously, the more messages received in a short period of time, the more unstable routing is, and large circles immediately draw the user's attention. The shrubs' height has no inherent meaning; we use different heights simply to prevent shrubs from occluding each other. However, the presence of tall shrubs indicates that many different instability events occur around that period of time, thus providing an additional visual cue.

Figure 6 shows the event shrubs visualization of the routing instability events of the Google IP address for the year 2001. You can see that a severe instability event occurred around July of that year.

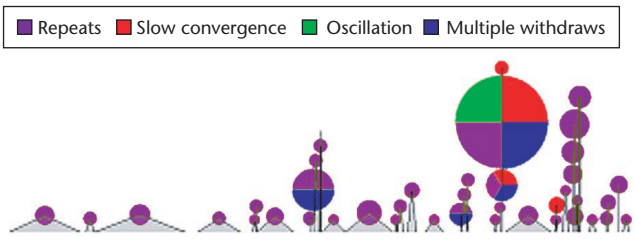
Using the visualization system's browsing feature, the user quickly locates the period of instability and looks at the text visualization of the update messages, as Figure 7 shows.

To investigate further, the user looks at the BGP update messages from a different peer. Each peer connected to the node from which we're collecting the data has a path to the destination IP address, in this case Google. The visual displays from Figures 6 and 7 show the messages from peer AS 2914. The user now looks at the messages from another peer, AS 3333. AS 3333 messages show no corresponding instability in this same time period; rather, they show a long withdraw message, meaning that no path is available to the IP address. As Figure 8 (next page) shows, we can visually compare the messages from the two peers by telling the visualization program to display them side by side. The instability probably occurred because the Google server was experiencing some major problems at the time and hence working sporadically. This caused frequent availability changes, leading to many BGP announcements and withdrawals. Most likely, route dampening caused an AS along the path from AS 3333 to filter these messages.

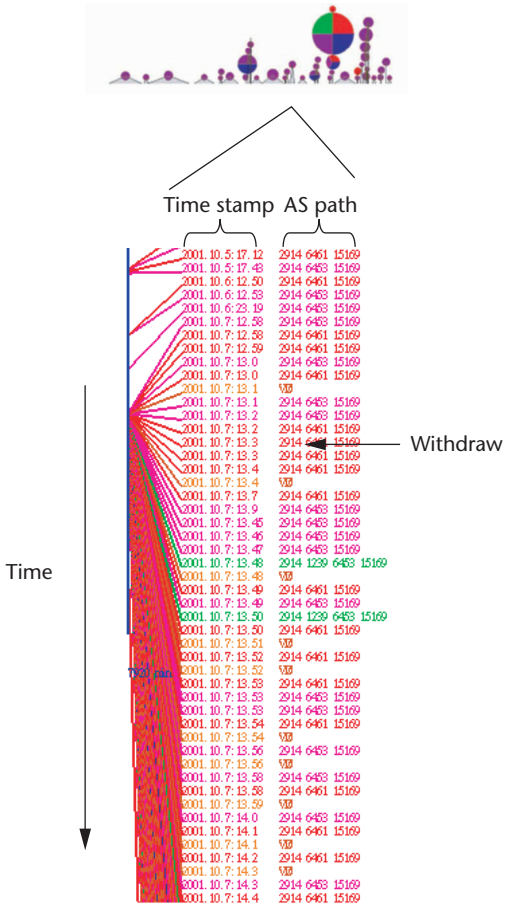
Intrusion detection

Users can examine user activity logs, system calls, or network connections to detect most intruders in a computer network system. We let users interactively explore logs so that they can detect intruders to computer network systems.

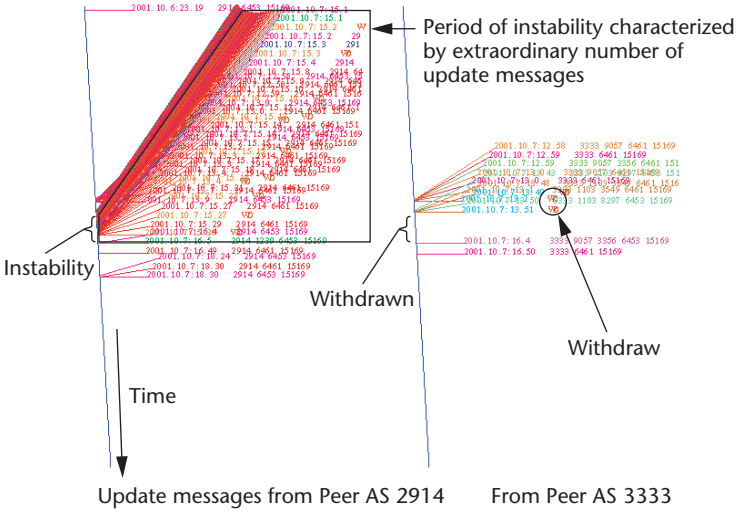
We treat intrusion detection as an event-classification problem. In classification problems, objects are defined by their attribute values in a multidimensional space; furthermore, each object belongs to one class among a set of classes. The task is to predict, for each object whose class is unknown, the class the object belongs to. Users typically train classification systems using a set of data with known attribute values and classes. After building a model based on the training, the system uses it to assign classes to unclassified objects. A classification-based intrusion-detection system (IDS) therefore takes network connection samples labeled "attack" or "normal" and uses them to construct a model to which it can compare future connections, and thus detect attacks. We use visualization not only for classification, but also for anomaly detection and cluster analysis.



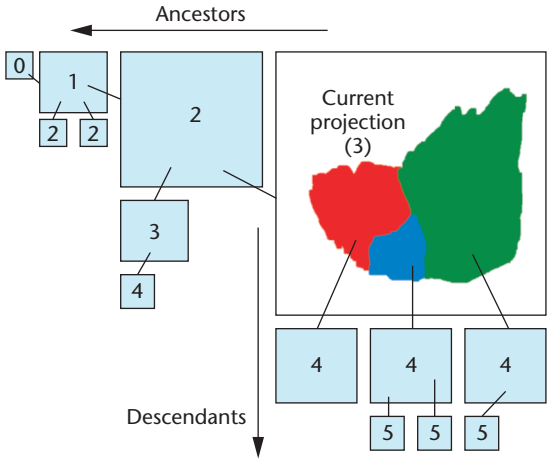
6 **Event shrubs visualization of the route path to the Google IP address. One instance of severe instability occurred around July 2001.**



7 **Visualization of the update messages for Peer AS-2914 during the Google instability event. Each update message is written horizontally across, with a line drawn to the vertical time line to indicate when the message was received.**



8 Visualizing update messages from peers AS 2914 and 3333 shows that damping along the AS path from AS 3333 to the origin AS filters out the instability. Visual analysis discovered the lack of damping from AS 2914.



9 PaintingClass decision tree visualization layout. We number each projection by its distance to the root—for example, the current projection has three regions, and each has been reprojected to a child projection.

Visual classification. We use PaintingClass,[6] a user-directed visual-classification program, to display data. PaintingClass uses decision trees, a popular and well-known classification approach. A decision tree classifier constructs a decision tree by recursively partitioning the data set into disjoint subsets. It then assigns one class to each leaf of the decision tree.

Interactive decision tree construction starts by visualizing the training set, which consists of connections with known attribute values and classes. In the decision tree's root, PaintingClass projects and displays every object in the training set visually. Each nonterminal node in the decision tree corresponds to a projection—that is, a mapping from multidimensional space to 2D display. We use the star coordinates[7] projection method, in which the position of a point representing a data object is given by

$$\left(\sum_{i=0}^{n} a_i x_i, \sum_{i=0}^{n} a_i y_i\right)$$

where a_i refers to the data object's attribute value in dimension i, and (x_i, y_i) are the screen coordinates of the endpoint of axis i. In other words, a point's position is influenced by its attribute value in each dimension and that dimension's user-defined axis position. By moving the axes in a star coordinates projection, the user creates a projection that separates the data objects belonging to different classes. The projection therefore determines each data object's position and assigns it color according to its class.

The user partitions each projection into regions by painting (to paint, the user left-clicks and drags the mouse cursor over the screen), which colors the cursor's path with the selected color. The user can reproject any region in the projection to form a new node—that is, the user creates a projection for the new node in a way that separates the data objects in the region leading to this node.

The user can partition each new node's associated projection into regions. The user recursively creates new projections (nodes) until he or she has constructed a satisfactory decision tree. Each projection thus corresponds to a nonterminal node in the decision tree, and each unprojected region corresponds to a terminal node. Thus, for each nonroot node, PaintingClass projects and displays only the objects projecting onto the chain of regions leading to the node. PaintingClass displays the decision tree according to the schematic in Figure 9, letting the user switch the focus to different projections, move the star coordinates axes, paint regions, and build the decision tree.

In the classification step, PaintingClass projects each object to be classified, starting from the decision tree's root and following the region-projection edges to an unprojected region, which is a terminal node (or leaf) of the decision tree. PaintingClass predicts the class with the most training set objects projecting to this terminal region for the object.

We applied PaintingClass to the Knowledge Discovery and Data (KDD) Mining Cup 1999 intrusion-detection data set. Each object in the data set is a network connection. Each object is defined in 41D space, and belongs to one of five classes: normal, probe, denial-of-service (DOS), unauthorized access to root (U2R), and unauthorized access from remote machine (R2L). Objects in the normal class are harmless connections, whereas objects in the other four classes are different types of attacks. The training set contains 494,021 connections; the text data includes 311,029. The KDD Cup 1999 data set is the only large-scale, publicly available data for evaluating intrusion-detection tools. A detailed description of the data set is available at http://kdd.ics.uci.edu/databases/kddcup99/task.html.

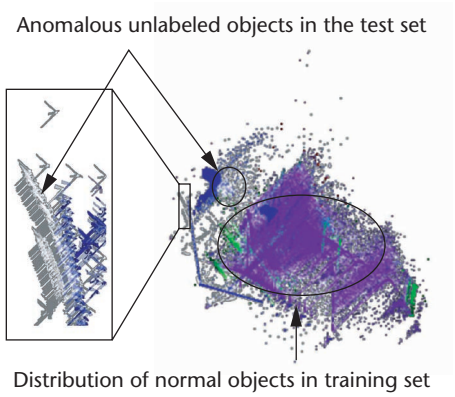
We visualize the KDD Cup training set and interactively construct a visual decision tree by painting regions and specifying projections. We use this decision tree to classify the test data, omitting the class labels.

Anomaly detection. Anomaly detection complements standard decision tree classification. Given examples of only normal activity, an anomaly detector creates a model of normal activity characteristics. When it sub-

sequently receives unlabeled activity, the anomaly detector compares the data against the normal pattern and flags the new activity as an anomaly if it deviates from that pattern. The advantage of anomaly detection lies in discovering previously unknown attacks.

To use the visual IDS for anomaly detection, we display the connections to be classified with the training set, coloring objects in the training set according to their class, and color data to be classified gray. For testing purposes, we color objects in the test set gray, as we would if they were new connections to be classified as intrusion or normal. We don't want to reveal the test set objects' class labels because we're simulating the visualization of as yet unclassified connections.

Displaying test set objects (with class labels omitted) with training set objects lets the user identify regions where the density distribution in the two sets differs. In particular, the user looks for regions in which the training set has a low density of normal data but the test set has a high-density cluster. We consider such a region an anomaly because the density distribution deviates from normal. Figure 10 is an example. Using PaintingClass, the user paints and labels the region as an attack class based on neighborhood information, and predicts all test objects projecting to the region belong to the labeled class.



10 **Distribution of normal objects (purple points) in a training set. Gray-scale objects belong to the unclassified test set. Two clusters of test set data are highlighted. The absence of normal data in the training sets in**

Table 1. Cost score of visual intrusion-detection system compared with the top five KDD Cup 1999 contest entries.

System	Score
Visual classification and anomaly detection	0.2087
KDD Cup first place	0.2331
KDD Cup second place	0.2356
KDD Cup third place	0.2367
KDD Cup fourth place	0.2411
KDD Cup fifth place	0.2414

Accuracy. We measured the accuracy of our visual IDS using the KDD Cup 1999 test data. Using the visual IDS, the user first visualized only the training data and built a decision tree classifier (this corresponds to using PaintingClass purely as a visual classification tool). Next, the user visualized test data (with class labels omitted) with training data and constructed a new decision tree (this corresponds to using PaintingClass as both a classifier and an anomaly detector).

A description of the contest's scoring system along with the contest results are available at http://www.cs.ucsd.edu/users/elkan/clresults.html. Table 1 shows the cost score of the top five contest entries. Visualizing only the training data using the visual IDS incurred a cost of 0.2551, placing it in the middle of the 24 entries. Incorporating anomaly detection into the visual IDS reduced the cost to 0.2087. This cost is significantly lower than that of the contest's winning entry, considering that the threshold for statistical significance used in judging the contest is 0.0060. The improvement of our present results over those of actual participants in a past contest certainly doesn't indicate our method's superiority. Our intention is merely to show that visual classification and anomaly detection work.

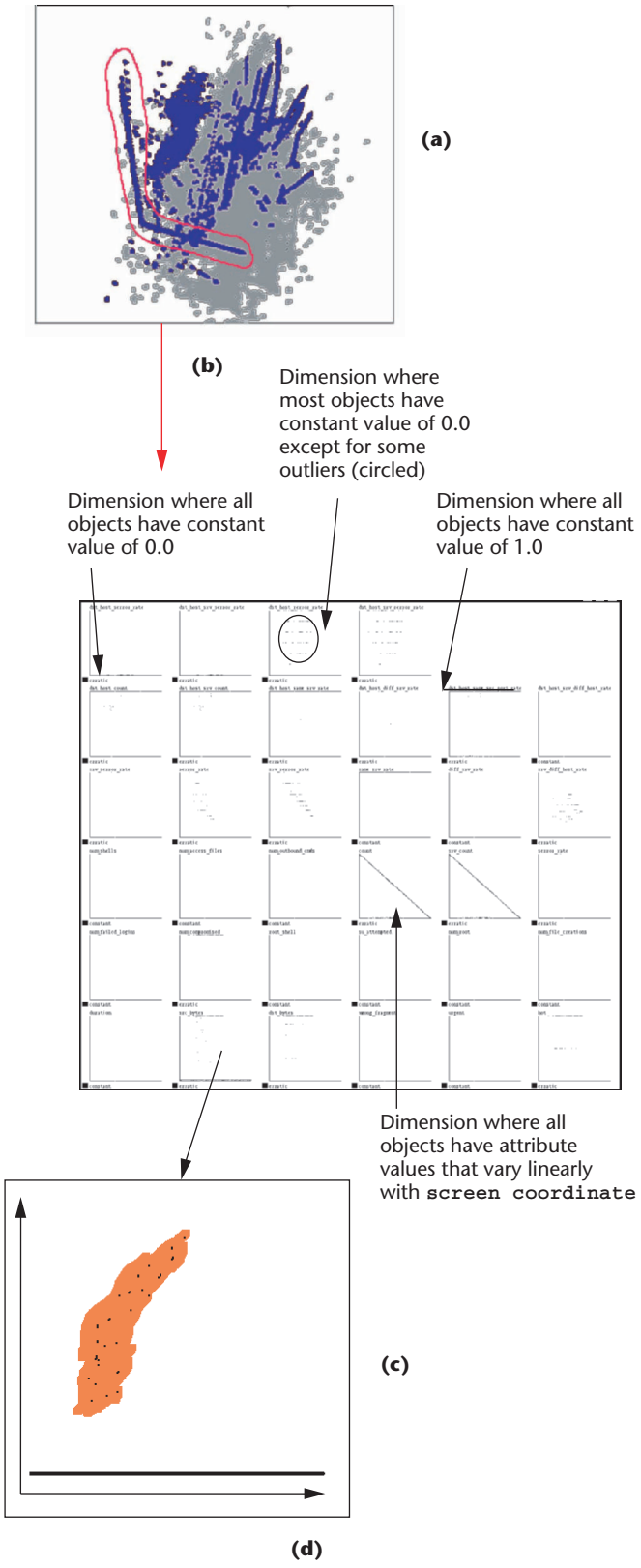
From patterns to knowledge. Perhaps the most important contribution of visual intrusion detection is that it reveals new patterns during interactive use of the visualization system.

We created an interactive visual mechanism to let users analyze such patterns in higher-dimensional space. After painting on an observed pattern, the user specifies the class to be examined and clicks "correlate." The program lists all objects that both belong to the specified class and are projecting to the painted region. From this list, the program finds the minimum and maximum values of the screen's x and y coordinates. The `screen coordinate` is the coordinate with the larger range (maximum to minimum). The program displays a table of plots, one plot for each dimension in the original attribute space. In each plot, an object is displayed as a point whose attribute value in that dimension determines its y coordinate. The `screen coordinate` determines the x-coordinate.

Figure 11 (next page) shows an L-shaped cluster of DOS connections and a table of plots. Most points follow a trend in each plot. For example, in some dimensions, the object's attribute values are constant; in others, they vary linearly with the `screen coordinate`. The user thus hypothesizes that the trends observed describe a cluster, and that the anomalous points don't belong to the cluster. The user paints on these points and clicks "remove." The system removes the objects represented by these points, and labels the remaining points as belonging to this cluster.

This process lets us define clusters of attack types precisely. Each constant dimension d_i results in an equation $d_i = c$. Each pair of linearly varying dimensions d_i and d_j results in an equation $d_i = k \times d_j + c$. For example, we can define the L-shaped DOS cluster in Figure 11 as a 25D cluster. It contains the entire L-shaped cluster, consisting of 280,795 objects with 3,649 false positives. (A false positive is a non-DOS object classified as a DOS object.)

If we consider additional dimensions, such as srv-count and count, the two parts of the L shape become separate clusters, as painting the entire L shape and viewing each dimension in the matrix of plots reveals.



11 Visual cluster definition process (DOS attack connections are blue, all other connections are gray): (a) User notices an L-shaped cluster of DOS attacks and paints over it. (b) The program makes a matrix of plots of all DOS attacks in the painted region. (c) User gets rid of outliers. (d) Cluster is defined.

The lower-dimensional description is therefore a generalization of the two higher-dimensional clusters, and represents characteristics the two clusters share. We can thus use this signature to detect new variants of this attack type. We also found lower-dimensional generalizations of other clusters.

Future work

Although we made some key discoveries using visualization, other useful information in the data sets is yet to be uncovered. For example, many OASC events remain to be analyzed. Although our visual representations have revealed some patterns, additional algorithmic filtering and processing may reveal other patterns, such as the correlation between days or temporal patterns pertaining to specific ASs or IP prefixes. Such patterns, if they exist, aren't easily observable using current visualization techniques. Similarly, we can apply machine-learning algorithms to routing instability analysis to generate better instability definitions, or to intrusion detection to help the user find useful projections. We are thus working to efficiently combine interactive visual methods with algorithmic methods to make further analyses and discoveries.

We can analyze the cluster definitions described in the previous section to extract attack type characteristics. We're also looking at real-time BGP monitoring and real-time deployment of the visual IDS.

Our intention isn't to replace conventional machine-learning methods with visual data mining. Just as different machine-learning methods can complement one another in discovering knowledge, adding visualization to the suite of data mining methods can help discover missed patterns. The full power of visual analytics for log-file analysis thus remains to be exploited. ■

Acknowledgments

The US National Science Foundation partially funded this work under contracts ACI 9983641 (PECASE award), ACI 0222991, and ACI 0220147 (ITR). We thank Ke Zhang for preparing some of the data files.

References

1. C. Ahlberg and B. Shneiderman, "Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays," *Proc. CHI 1994: Human Factors in Computing Systems*, ACM Press, 1994, pp. 313-317.
2. L. Girardin, "An Eye on Network Intruder-Administrator Shootouts," *Proc. Workshop on Intrusion Detection and Network Monitoring* (ID 99), Usenix Assoc., 1999, pp. 19-28.
3. S.T. Teoh et al., "Case Study: Interactive Visualization for Internet Security," *Proc. IEEE Visualization Conf. 2002*, IEEE CS Press, 2002, pp. 505-508.
4. T.J. Jankun-Kelly and K.-L. Ma, "MoireGraphs: Radial Focus+Context Visualization and Interaction for Graphs with Visual Nodes," *Proc. 2003 IEEE Symp. Information Visualization*, IEEE CS Press, 2003, pp. 59-66.
5. S.T. Teoh, K.-L. Ma, and S.F. Wu, "Visual Exploration Process for the Analysis of Internet Routing Data," *Proc. IEEE Conf. Visualization 2003*, IEEE CS Press, 2003, pp. 523-530.

6. S.T. Teoh and K.-L. Ma, "PaintingClass: Interactive Construction, Visualization, and Exploration of Decision Trees," *Proc. 9th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, ACM Press, 2003, pp. 667-672.
7. E. Kandogan, "Visualizing Multidimensional Clusters, Trends, and Outliers Using Star Coordinates," *Proc. 7th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, ACM Press, 2001, pp. 107-116.

Soon Tee Teoh *is a PhD candidate at the University of California, Davis. His research interests include network stability, security, and the discovery and analysis of patterns, anomalies, and knowledge in large data sets. Teoh has a BS in electrical engineering and computer science from the University of California, Berkeley, and an MS in computer science from the University of California, Davis.*

Kwan-Liu Ma *is a professor of computer science at the University of California, Davis. His research interests include visualization, user interfaces, computer graphics, and high-performance computing. He is the editor of the VisFiles Column in ACM Siggraph's* Computer Graphics Quarterly. *Ma has a PhD in computer science from the University of Utah. He is a senior member of the IEEE and a member of the ACM.*

Shyhtsun Felix Wu *is an associate professor of computer science at the University of California, Davis. His research interests include fault-tolerant networks, Internet Protocol Security/virtual private network security policy, attack source tracing, wireless network security, and intrusion detection and response. Wu has a BS from Tunghai University, Taiwan, and an MS and a PhD from Columbia University, all in computer science.*

T.J. Jankun-Kelly *is an assistant professor in the Department of Computer Science and Engineering at Mississippi State University, where he is part of the Visualization, Analysis, and Imaging Laboratory in the Engineering Research Center. His research interests include visualization (both information and scientific), computer graphics, user interfaces and models for visualization, Web-based visualization, graph visualization, time-varying volume visualization, and transfer function generation for volume visualization. Jankun-Kelly has a BS in physics and computer science from Harvey Mudd College and an MS and a PhD in computer science from the University of California, Davis. He is a member of the IEEE Computer Society, the ACM, and Siggraph.*

Readers may contact Soon Tee Teoh at the Dept. of Computer Science, Univ. of California, One Shields Ave., Davis, CA 95616; steoh@ucdavis.edu.

SCHOLARSHIP MONEY FOR STUDENT MEMBERS

Lance Stafford Larson Student Scholarship best paper contest

✶

Upsilon Pi Epsilon/IEEE Computer Society Award for Academic Excellence

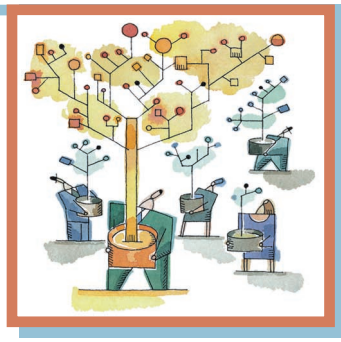
Each carries a $500 cash award.

Application deadline: 31 October

IEEE COMPUTER SOCIETY

Investing in Students

www.computer.org/students/



JOIN A THINK TANK

Looking for a community targeted to your area of expertise? IEEE Computer Society Technical Committees explore a variety of computing niches and provide forums for dialogue among peers. These groups influence our standards development and offer leading conferences in their fields.

Join a community that targets your discipline.

In our Technical Committees, you're in good company.

www.computer.org/TCsignup/