EG UK Theory and Practice of Computer Graphics (2013)
Silvester Czanner and Wen Tang (Editors)

NetVis: a visualization tool enabling multiple perspectives of network traffic data

J. Nicholls, D. Peters, A. Slawinski, T. Spoor, S. Vicol, J. Happa, M. Goldsmith and S. Creese

Department of Computer Science, University of Oxford

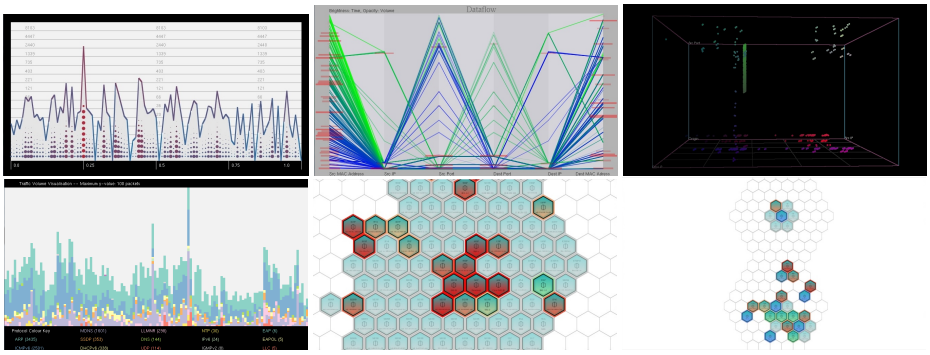


Figure 1: *The suite of visualizations in NetVis. In order of appearance from top-left to bottom-right: Attribute Distribution, Data Flow (Parallel Coordinate Plots), Spinning Cube of Potential Doom, Traffic Volume, Heat Map and Activity Groups.*

Abstract

Computer network traffic visualizations deliver improved understanding of pattern-of-life for networks, and via such enhanced awareness can facilitate the detection of malicious traffic. Existing tools often opt for graph or plot-based visualizations to detect patterns or outliers in the data, but they still largely provide a segmented view of any data feed. In this paper we present a novel framework designed to support multiple heterogeneous visualizations of network traffic data. NetVis enables different visualizations that work in tandem to provide different perspectives of the same data in real-time. As each visualization is modularly tied together it enables a user to investigate on-going activity, or any subset of it, at their pace and based on their priorities for further exploration. We currently support six visualizations, three are new and three are based on existing literature (parallel coordinate plots, flowscan and spinning cube of potential doom). Our results show that it is possible to use NetVis to detect unusual activity and cyber attacks on a network. The framework is written to allow future visualizations to be added straightforwardly.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications—K.6.5 [Management of Computing and Information Systems]: Security and Protection—

1. Introduction

Monitoring and analysing network traffic is an important part of network security as it provides a means to detect malicious activity on network systems. This is typically by detecting known attack patterns or by seeking to understand normal pattern-of-life activity for the network and then detecting deviations which might indicate presence of malicious events. As the amounts of data to process increase substantially each year, it is becoming more and more difficult to make effective analysis of on-going activity on a network. Visualizations offer a very useful solution: they give a compact representation of data to a human analyst who can use them to detect patterns in the network [**?**]. Effective visualizations makes it easy to identify outliers while making

clear what general patterns are arising, facilitating detection of malicious activity.

In this paper, we present a framework implemented as a software application. Specifically, we are interested in harnessing the ability of the human analyst to *detect* unusual and possibly malicious activity, and in supporting that analyst through the use of visualisations; in particular, through the use of multiple visualisations from a single platform. We envisage an environment where the analyst can easily switch between visualizations, exploring the dataset until a particular visualisation resonates as *showing* something of interest. Current approaches to supporting analysts with visualizations are typically configured around proprietary Security Incident Event Management (SIEM) environments, and so are by their nature limited in scope to a finite set of visualizations. In contrast, our view is that the visualization should be working in tandem to improve situational awareness and that we should not assume that there is any single optimal set of visualizations which will work for all analysts in all operating conditions. For this reason we have developed a framework which can easily support new visualizations as well as the tailoring of existing ones.

We enable interactive analysis, we allow a deeper and more straightforward understanding of the data by enabling an analyst to explore their raw datasets in a way that they deem appropriate given any current situation. Time series data, in the form of packet captures, are processed in real-time and simultaneously rendered in multiple connected visualizations. The user can switch between the available representations and change both the visualization's layout and the amount and type of the data displayed. The software is designed to make it easy to spot irregular activity and investigate it using multiple perspectives. Three of the visualizations are based on existing literature (*Parallel Coordinate Plots* [**?**], *Flowscan* [**?**] and *Spinning Cube of Potential Doom* [**?**]). Some of their limitations have been addressed in our implementations. The three remaining visualizations are novel to this paper. These are referred to as: *Attribute Distribution, Heat Map* and *Activity Groups*.

New visualizations can complement the existing ones: The underlying data processing engine provides a standard data basis which is shared by all displays. The tools emphasize exceptions, show comparisons, and answer a wide variety of questions in a concise fashion. We want the user to generate good hypotheses in response to the visualized information. To achieve this, we have implemented a filtering system which can be adjusted in response to changing situations.

Though NetVis's main purpose is to monitor current activity in the network, the same architecture could be used to forensically analyse recorded data. This allows us to exploit the important dimension of time, making it easier to understand recorded activity in the context of a digital investigation after the fact.

The remainder of the paper is divided into the following sections: Section 2 summarises related work. Section 3 provides an overview of NetVis's architecture design. Section 4 describes each of the visualizations in detail, and Section 5 describes our workflow paradigm with some example scenarios to demonstrate the usefulness of our framework. A reflection on the utility of our methods and future work directions can be found in Section 6 and the conclusion is presented in Section 7.

2. Related Work

Visualization approaches provide different perspectives view of a dataset based on criterias of interest. Some of the techniques favour graph-based representations, port-scanning activity characteristics, network traffic patterns, payload characteristics or event-log forensics. More specifically, these could include use of parallel coordinate plots [**?**], treemaps [**?**], geolocation [**?**] and visual firewall [**?**].

Rumint [**?**] and Wireshark [**?**] are examples that can be used for traffic forensics. For analysing network traffic patterns a variety of strategies have been proposed including traffic volume [**?**] and raw data relationships [**?**]. Best et al. [**?**] also demonstrated the usefulness of tools such as Traffic Circle, CLIQUE and MeDiCi to improve situational awareness by exploring raw dataset through visualization.

Conti [**?**] and Marty [**?**] provide a detailed discussion on this topic, while Zhang et al. [**?**] presented a survey overviewing the techniques. Other tools and visualization techniques can be found in the SecViz.org [**?**] gallery.

2.1. Motivation

While many tools exist to improve aspects of network traffic data understanding, few works have discussed how multiple visualization techniques can work together to deliver an improved understanding of network activities. NetVis addresses this problem by showing how multiple heterogeneous of visualizations approaches can work together. Three are improvements on known visualization techniques (Parallell Coordinate Plots, FlowScan and Spinning Cube), while the latter three are new visualizations for network traffic.

3. NetVis Architecture

NetVis is a Java application which uses OpenGL (JOGL) to draw visualizations and Swing to provide a Graphical User Interface (GUI). In designing the application, maintaining extensibility and keeping a modular programming style was a large focus. It is now straightforward to add support for additional input data formats, or to develop a new visualization that utilises the same data processing engine.

Packet data is transferred to the application as a CSV packet capture file. NetVis uses a file format that can be directly obtained from the packet analyser Wireshark [**?**].

NetVis will process each packet and analyse its fields which includes: *timestamp, information about the packet's source (IP, hardware (MAC) address, port), information about the packet's destination, communication protocol, and the packet length in bytes*. The current version of NetVis does not process the content of packets.

A convenient feature of the data input system is a time control system. The CSV file is processed as if its packets would arise in a live network. The user can choose to speed up or slow down the speed with which packets are fed into the application, can pause processing, or skip towards the end of the data record. This is helpful in analysing the data since critical time intervals can be analysed in more detail. The application is set up in such a way that it is also easy to use the activity of a live network as its input source.

The input data is processed in a data controller that supplies visualizations and the user interface with packet data. If the user has chosen to apply filters to the data, the data controller only directs the filtered data stream to the rest of the application, so that all parts share a common data source.

NetVis includes a data filtering system that allows users to select a subset of processed packets which exhibit features which the user specified to be of interest. The application supports two types of filters: filters that the user explicitly defines, and filters applied on-the-fly from within the visualizations. Both types are applied to all visualizations and information displayed, and they can be adjusted at any time without losing prior data.

There are a variety of filter controls. First, there is a menu for filtering by transit protocol. By default, all protocols are selected and therefore included. Protocols are sorted into menus by protocol family, appearing in multiple places where appropriate. Second, there is a control to select the range of ports the application uses, which defaults to the maximum port range. The source and destination ports can be set separately, enabling a user to view all data entering/exiting a port as desired. Third, there are IP and MAC address filters. These work on a blacklist/whitelist system, allowing a user to only view packets to or from a particular set of addresses, or to ignore packets going to or from a different set. This enables a user to, for instance, ignore traffic from sources they know are irrelevant or focus only on an address that is causing concern.

The packet attributes that are processed by NetVis do not share a common scale, nor are they necessarily orderable. It is, however, desirable to map some of these properties into a shared representation which allows a more intuitive grasp of the distribution of attributes. We achieve this by processing the packets in a 'normalising' system. This system keeps track of the used values and maps them into the interval $[0,1]$. In this way all normalised packets can be displayed in relation to a number axis. This representation is used in three different visualizations. Each normaliser is able to create a temporary filter in the application that will filter its corresponding attribute on a certain range. Since the normalising class is in control of its filter this creates a zooming effect based on the range of the filter: the lower bound is normalised to 0, the upper bound to 1.

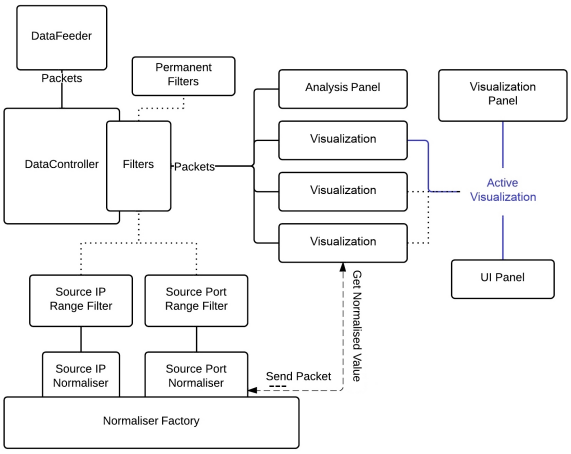


Figure 2: *Diagram overviewing the architecture of NetVis.*

The application is designed to be extensible. Adding a new visualization is a matter of adding it to the application's list of those available, which will keep it up-to-date as packets come in and are filtered. The same is true for filters and normalisers. A diagram overview of the system's architecture can be found in Figure 2.

4. NetVis Visualizations

4.1. Graphical User Interface

The displayed (current) visualization occupies most of the user interface and can be maximised to fill all screen space. A displayed visualization may use both multi-dimensional systems (such as parallel coordinates) to give an overview and lower-dimensional visualizations to provide more detail. This encourages an understanding of network activity that is both broad and detailed. Other panels of the GUI occupy the right and bottom sections of the window to allow user input and detailed information to be viewed, see Figure 3. **Key**: 1) Visualization, 2) Analysis Panel, 3) Context Panel, 4) Visualization-specific options, 5) General data filters and 6) Data source identifier.

4.1.1. GUI: Right Panel

The right panel is aimed at providing the user with options to adapt the shown data. Choice of visualization is presented to the user by a list in the top of the right panel, followed underneath by options specific to that visualization, and then by general filters to refine the processed data. If a recorded data source is active, time controls are displayed on the right panel to allow the user to adjust the speed at which data is

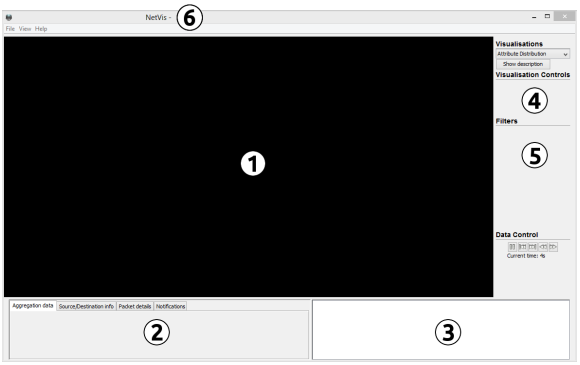


Figure 3: *An overview of the GUI layout of NetVis.*

processed and visualised by pausing, doubling or halving the current data rate.

4.1.2. GUI: Bottom Panel

The bottom panel shows fine and aggregate details of the processed data. To fit the large amount of data available into such a limited space, a tabbed pane is used on the left to categorise distinct types of data. The right section of this panel contains a 'Context Pane', which shows further detail of selected data on demand. The left 'Analysis' panel and the right 'Context' panel are split in two by a split pane, so the user can modify the proportion of each they wish to see.

Window tools, an exclusive-mode full screen option and the facility to select a new data source are all accessed by the main menu bar. Messages to the user come in the form of warning dialogues for user or program errors, notifications in the bottom 'Analysis' panel for program notifications, and text messages in the bottom 'Context' panel for help messages and suggestions.

4.2. Visualization: Attribute Distribution

The most common need in the analysis of network traffic is information on the distribution of certain features of the packets. For the graph constructed here, the user selects such a feature (e.g., ports or destination IPs). A logarithmic line graph is drawn and (as with all visualizations in the application) updated in real time. A logarithm is applied to assist in detecting spikes. A screenshot of the Attribute Distribution visualization can be seen in Figure 4.

A second layer of the visualization is presented simultaneously: circles under the line graph indicate the actual distribution (without a logarithm applied) through their area. Furthermore, all lines and circles are colour-coded to give extra signals of the volume of traffic. Red elements are under heavier load. The visualization makes active use of the aforementioned normalisers. It also allows users to click-and-drag along interesting data to zoom into. Internally this

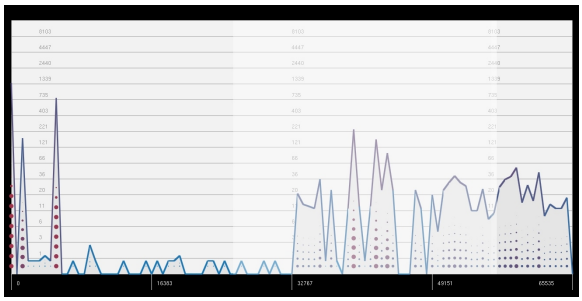


Figure 4: *Attribute Distribution visualization.*

means that a filter is applied and all visualizations show only the selected packets for closer examination.

4.3. Visualization: Data Flow

The Data Flow visualization applies the idea of parallel coordinates proposed by [**?**] to network traffic, see Figure 5. Following the principles in [**?**] the visualization shows the 'flow' of each packet, representing each as a line through the parallel coordinates. This provides an informative view of the whole traffic in the network. It visualises the distribution of various packet attributes while also giving an intuitive insight into relationships and correlations between distinct aspects of the packets.

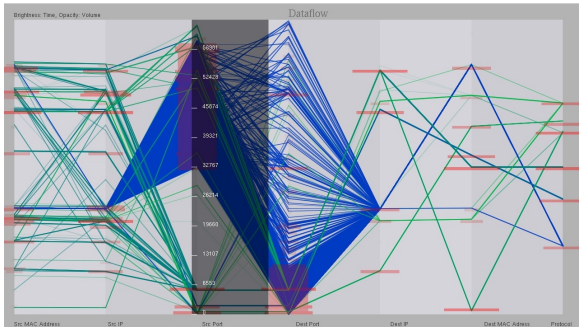


Figure 5: *Data Flow visualization inspired by parallel coordinate plots [**?**].*

Lines representing packets are coloured based on their value in some coordinate. They fade out based on how old the packet is, thereby placing focus on newer developments in the network.

A common criticism of parallel coordinate plots is that they make it hard to interpret data that is uniformly distributed between coordinates or that is very concentrated around few values [**?**]. Our implementation addresses these concerns using animation. Lines representing new packets are randomly perturbed and move slightly. This makes it easier to spot whether a line represents one or more packets.

The animation allows the user to distinguish between single packets and concentrated groups of packets with the same characteristics.

Suppose that multiple requests to a server come from a single MAC address through the same ports and using the same protocol. A non-animated visualization would represent all these requests as a single line. A user would erroneously interpret this as a single request. Adding randomness makes the pack of requests more visible without significantly impacting the accuracy of the data representation.

The Data Flow visualization is closely tied to the previously described Attribute Distribution visualization. If a coordinate shows an unusual distribution (say a uniform use of ports in a certain range), then a single click on the packets in this coordinate will show the Attribute Distribution log plot which allows direct access to filtering option. Hovering over packets also displays further contextual information.

4.4. Visualization: Spinning Cube

The Spinning Cube is a three-dimensional scatter plot that displays packets as dots inside a rotating cube. The position of the dot is determined by its packet's attributes. This could be the combination of source port, destination port, and source IP but also any other choice of packet attribute, see Figure 6 for a screenshot.

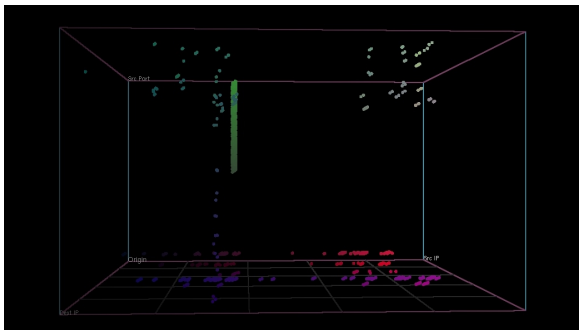


Figure 6: *Spinning Cube of Potential Doom [?].*

The visualization is an implementation of an existing visualization tool known as the Spinning Cube of Potential Doom [**?**], but it offers additional controls for the user. In particular, the axes can be chosen to represent arbitrary packet attributes that the user specifies.

To ensure good distinguishability of individual packets, we use the same principle of randomness as in the Data Flow visualization. Every point is animated and perturbed. In this way, multiple packets with the same attributes do not occupy the same pixel in the cube. It is thus easier to spot the packet density of an area in the cube.

4.5. Visualization: Traffic Volume

The Traffic Volume visualization is inspired by the 'FlowScan' graph [**?**]. The objective is to allow users to see at a glance if a particular protocol is being exploited in the network. It is realised as a stacked bar chart which displays the volume of data arriving in each time interval. Each column is segmented into sections with heights proportional to the total number of packets transmitted for each protocol. In addition, the column segments are colour-coded and can be cross-referenced with a protocol key underneath the visualization, see Figure 7 for a screenshot.

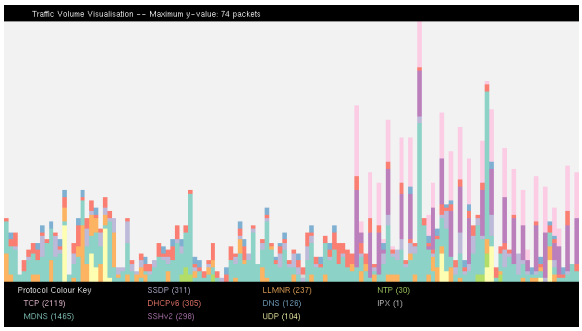


Figure 7: *Traffic Volume visualization, inspired by FlowScan [?].*

To help distinguish between different protocols, colours are selected from a palette which provides colours from a qualitative colour scheme. The increase in both column height and colour proportion should draw attention to any protocol which becomes overburdened. For instance, if a network saturated with TCP traffic suddenly becomes flooded with DNS requests, the drastic change in colour will alert the user to the change in circumstances.

A control panel is provided in the right panel for adjusting the number of time intervals displayed at once on the x-axis. The y-axis scales automatically to fit the current data.

4.6. Visualization: Heat Map

The Heat Map visualization uses a hexagonal map to display a compact cluster of nodes. Each machine found to be communicating in the analysed network will be represented as a node. The activity of a specific machine in the network is represented by the colour of the corresponding hexagon. As more and more traffic arises, nodes representing the machines that send and receive data "heat up" and, as the time passes, "cool down". The hexagonal grid is filled in a compact manner - new nodes are place outside of the already displayed nodes. The relative position of the map elements in this visualization does not play a role on default. There is a way to sort the nodes so that they form a radial gradient - the more "heated" will be sorted to the middle. See Figure 8 for a screenshot.

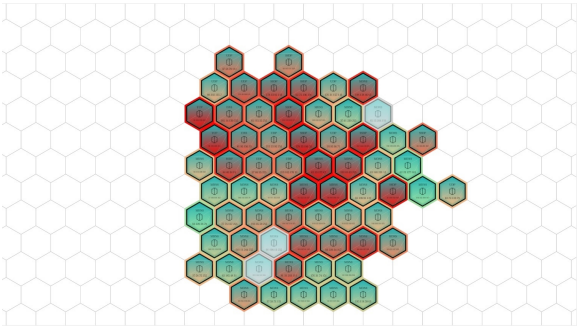


Figure 8: *Heat Map visualization.*

This visualization helps a user find machines that are most active in the network. It gives a quick overview of the number of the clients and how many of them are currently active. Additionally every node presents some information, such as the machine's most commonly used protocol. This lets users do some basic data selection choices easily. For example: if some other visualization is being obscured by too much data from one specific machine, or too much data of some protocol, the user can look at the heatmap and find out what filters to apply to make the overview of the situation more clear.

4.7. Visualization: Activity Groups

This visualization also makes use of the hexagonal grid. In the attempt to give a best possible insight it uses two visual factors: size and proximity. As the traffic in the network rises, the nodes representing machines are put on the map. All the machines sending packets to one specific device are being grouped together. As more machines communicate with a specific address, more nodes appear around the hexagon representing that destination, see Figure 9.

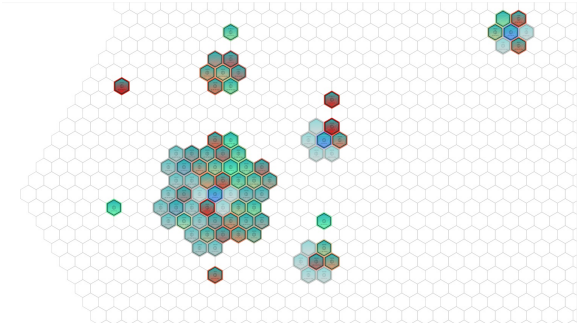


Figure 9: *Activity Groups visualization.*

The colour of the communicating nodes represents how much data they send. More heavily used destinations will thus have more orange and red nodes around them, and destinations used by a lot of machines will have a greater number of nodes around them. Node placement is automatic. The procedure ensures that there is enough space around a centre and that no two nodes overlap.

This visualization helps detect the machines that perform a server role in the network. It also allows the user to quickly guess what type of service the device is providing. Each node specifies the most commonly used protocol, thus revealing the possible role of the machine in the network. Once the typical pattern of connections is identified, it can be filtered out to show possible anomalies.

5. User Workflow

Our workflow can be considered an extension to the visualization workflow paradigm proposed by Ben Shneiderman [**?**]: *'Overview first, zoom and filter, then details-on-demand.'*. The core difference being that even an overview may have several perspectives of the same data. By allowing several perspectives to visualize the same data simultaneously enable analysts to explore a dataset based on their requirements and where they see patterns emerging.

A typical workflow for an analyst would begin by considering all activity from a broad overview, using any number of the visualizations. Once a network attack has been identified, the analyst to "zoom in" to the data of interest by applying successive filters to the source and changing the representation to suit the particular attack. Finally, when the exceptional behaviour has been singled out, the analyst may explore details of the intrusion. A diagram of the typical workflow for an analyst using NetVis can be found in Figure 10.

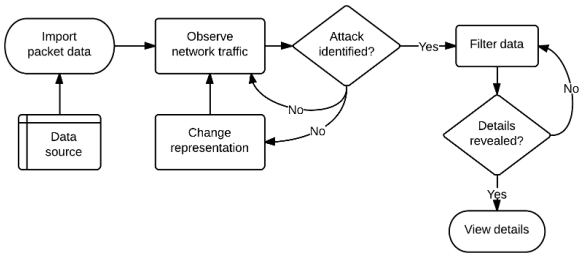


Figure 10: *A diagram outlining the workflow paradigm for using NetVis.*

To assist with the process of singling out suspicious activity, certain visualizations allow filtering by mouse interaction. Where appropriate, the analyst may click or drag their mouse cursor over sections of the active visualization to create a new filter or to switch to a different representation of the selected data. Navigation between complementing visualizations in this fashion is highly intuitive and allows the analyst to tweak the active filters quickly and effectively.

5.1. Example Scenarios

To obtain a complete understanding of a network's activity, a single visualization will not typically be sufficient. An analyst will need to see different approaches to interpret the

data, and will need to customise the visualizations and the data displayed in them. This is where the design principles of interactivity and interconnectedness of NetVis provide powerful assistance, and ensure that the visualizations in combination achieve both effectiveness and expressiveness [**?**]. An analyst can train for detecting attacks by using packet captures. By using a different filter combinations they are able to isolate the attack an learn more about it. Below follows two examples of the workflow in use.

5.1.1. Example Scenario: Portscan

In Figure 11(a) it is possible to see a tall spike in the attribute distribution. By visualizing the distribution of source IP's the analyst can distinguish a spike in the network traffic for a small IP range. Clicking the spike changes the current visualization to the DataFlow visualization, see Figure 11(b). It can then be noted that many sources may go to the same range of ports. A filter can then be applied to isolate who is responsible for the majority of the port activity. Figure 11(c) shows the same DataFlow visualization after applying a range filter to isolate the port scan which they can explore further in all the visualizations. The filtering shows a single IP is responsible for the portscans.

5.1.2. Example Scenario: SSH Brute Force Attack

In Figure 12(a) an analyst can observe some irregular activity in the traffic volume visualization. The ongoing volume of SSH is roughly the same amount at most of the intervals. After exploring with a couple of filters, see Figure 12(b) the analyst finds out that a single IP is responsible for all the irregularities. A quick look at the analysis panel (in the swing GUI) provides information about the attacker and the destination. By isolating the packets from the suspicious IP he can determine the type of attack.

(a) Suspicious SSH (b) Exploring SSH data

Figure 12: *Example scenario of an SSH brute forcing.*

6. Discussion

Important aspects of network traffic include the network's topology, the flow of the traffic, and the type of the traffic. The graphical visualizations presented here ensure that the available data can be interpreted from all three of these aspects. Moreover, we believe better insight into the activity in the network can be obtained by combining perspectives.

Each visualization fulfills a specific purpose. The Heat Map visualization provides a very quick impression of overall network load and size. The Activity Groups visualization provides the same information on a server-by-sever basis, providing more data at the expense of simplicity. The Data Flow visualization provides the ability to see what the traffic of the network currently 'looks' like, and the Spinning Cube and Attribute Distribution visualizations allow the user to detect patterns in the attributes of the packets.

The visualizations developed here differ in their approach to handling the inherent multi-dimensionality of traffic data. An effective visualization framework needs to make clear how traffic changes as time progresses, and needs to show interesting developments in diverse attributes such as port use, source and destination machines, protocol use or traffic volume. Showing all this information simultaneously risks obstructing the simplicity needed for ready understanding.

6.1. Advantages

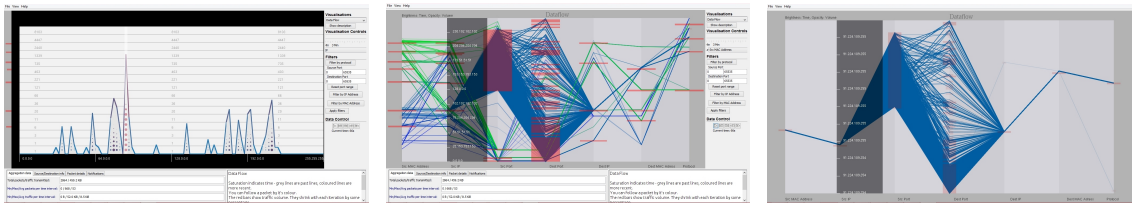
Existing network visualization solutions are typically single-purpose programs which present given data in one specific way (see for instance the catalogue in [**?**]). The analyst is expected to choose in advance which kind of visualization will provide the most insight, and is then limited to what the specific application is able to show. Existing applications also require different input file formats which makes simultaneous use complicated, and sometimes impossible when a SIEM uses proprietary middleware not commonly supported. NetVis takes a single data input stream and simultaneously visualizes it in different presentations. Since these visualizations are built upon a common base, they are able to complement and inform each other. This improves the user's ability to investigate anomalies.

Network visualizations have to process and display a large amount of data. Often, this can cause a visualization to become cluttered. An obvious solution is to provide filters which allow the analyst to focus on phenomena of interest. The filters we provide apply application-wide and can be defined in an intuitive manner from within the visualizations themselves in a "click-and-zoom" fashion. This is a significant improvement over existing workflows.

The framework developed here can be used both for real-time monitoring and for historical analysis, which makes it applicable to a wide variety of use cases. Furthermore, the application can be used as a learning tool: by observing archived data captures, new users can familiarise themselves with the general patterns of network usage and can see how suspicious patterns show up in a visualization.

6.2. Limitations

Our visualizations currently only support the analysis of time series data of packet captures at the network layer. However, a multi-layer approach combined with with security event information from intrusion detection systems,



(a) A tall spike in the attribute distribution. (b) Activity suggests a portscan. (c) Filters show a single IP is responsible.

Figure 11: *Example scenario of network portscan.*

routers and firewalls will likely help inform the analyst more accurately about the state of the network. The application in its current state does not analyse the content of packets. NetVis assumes domain expert users. If a user is unfamiliar with common network protocols it may be difficult for them to understand the meaning of emerging patterns.

6.3. Future Work

We have a number of avenues for future work. We will continue to explore the utility and effectiveness of the framework through the growth of visualizations tools it supports and experiments with expert users. We will extend the data types we support (to allow for the consideration of data from other layers and security appliances) and explore the relative benefits and limitations, seeking to identify any plateau effects (i.e. to keep vigilance levels high). We will also explore a more powerful data preprocessing system. Further, we will experiment with the application of statistical inference to the data with historical observations incorporated. Finally, it would be valuable to assess our workflow paradigm by testing it using data from outside laboratory settings and obtain comments from analysts in real environments.

7. Conclusion

We have presented a network visualization tool to analyse and monitor network traffic. Instead of presenting single perspectives (visualizations) of the data very well (but with a limited scope), our framework delivers a variety of visualizations that complement each other and work in tandem to help an analyst obtain improved awareness of ongoing network activity through user interaction. We overviewed its suite of visualizations and demonstrated its usefulness with example scenarios. In the future we intend to add more visualizations and further improve the tool's usability.