EG UK Theory and Practice of Computer Graphics (2013), pp. 1–5
Silvester Czanner, Wen Tang (Editors)

NetVis: A network traffic visualization tool

James Nicholls, Dominik Peters, Albert Slawinski, Thomas Spoor, Sergiu Vicol, Jassim Happa

Department of Computer Science, University of Oxford

Abstract

Computer network traffic visualizations attempt to deliver improved understanding of traffic on a network to an observer. Many existing tools opt for graph or plot-based visualizations to detect patterns or outliers in the data, but still largely provide a segmented view of any data feed. In this paper, we present a novel network traffic visualization framework that makes use of a variety of complementary visualizations to obtain better situational awareness. Our proposed solution is to look at different..

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Analysing and monitoring network traffic is essential for ensuring network security. Since the amounts of data involved in this task are extraordinarily large, it can be hard to make effective use of them. Visualizations offer a solution: they give a compact representation of data to a human analyst who can use them to detect patterns in the network [War12]. An effective visualization of network traffic will make it easy to identify outliers while making clear the general patterns of network usage, and it will facilitate detection of intrusions and of malicious activity.

We present here an application which provides a framework for analysing network activity in an interactive and holistic **[better word?]** manner. Time series data in the form of packet captures are processed in real-time, and simultaneously rendered in multiple interconnected visualizations. The user can switch between the available representations of the data and dynamically influence 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 then to investigate it from multiple angles.

The framework developed here aims to provide an environment in which situational awareness can more effectively be obtained. It is therefore built to be easily extendable to fit the demands of the specific situation. New visualizations will in a natural way complement the existing ones: The underlying data processing engine provides a standard, or *normalised*, data basis which is shared by all displays.

The purpose of the tools presented in this paper is to assist a human observer to make sense of what is happening in a given network. They emphasize exceptions, show comparisons, and try to answer a wide array of question in a concise and simple fashion. We want the user to generate good hypotheses in response to the visualized information. To achieve this, many aids are given to the user in order to avoid cluttering displays with irrelevant data. In particular, we have implemented a powerful filtering system which can be dynamically adjusted in response to changing situations.

The application's main purpose is to monitor the current activity in the network. However, the same architecture can be used to forensically analyse recorded data. The system will simulate the data records as if they represented a live network. This exploits the important dimension of time, and makes it easier to understand what is actually happening.

2. Related Work

JH

3. NetVis Architecture

Normalisers:

A normaliser is a class responsible with taking a packet and returning a "normalised" value of a certain attribute of that packet and the other way around. Each normalising class 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 it creates a zoom-

ing effect based on the range of the filter: the lower bound is normalised to 0, the upper bound to 1.

Flow of Data through the application: The data feeder creates Packet objects based on the traffic in analyses and feeds it to the Data Controller. We implemented a CSV data feeder that reads CSV files (/* maybe here a little more about the structure of the CSV's? */) created with Wireshark but the simple interface of the data feeder makes it easy to extend the application for reading live data.

The Data Controller uses the listener pattern and feeds new packets as they arrive to its listeners (visualizations, analysis panel). If there are any active filters then the data controller will apply those filters to the data and supply only what is relevant.

When a filter is applied or changed a reset signal is sent from the data controller informing the visualizations that all the data has changed. Along with this signal a list of all the past packets (with filters applied) is provided.

Filtering: **High-level: What do filters do?** The application supports two types of filters - filters that the user explicitly defines, and filters applied on-the-fly from within the visualisations. Both types are applied to all visualisations and information displayed, and can be adjusted at any time without losing prior data. The user has access to a variety of different filter controls. First, there is a menu for filtering by transit protocol. By default, all protocols are selected and therefore included. Protolcols 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. Next, 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 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 second type of filter will be dealt with in more detail in sections 4.2 and 4.3.

Modularity: **[Make this better.]** The application was designed to be very extensible. Adding a new visualisation is simply a matter of adding it to the application's list of those available, which will cause it to be included and kept up-to-date as packets come in and are filtered. The same is true for filters and normalisers. **[wrong place:]** Adding a filter would result in its controls automatically being included in the right panel and all packets would then be filtered according to the criteria it specifies. Adding a normaliser would cause it to be integrated with the Spinning Cube, Dataflow and Attribute Distribution visualisations. Ths modularity extends even so far as data input. If a class were written to accept packets from a different source, it would be trivial to switch the application to use this class.

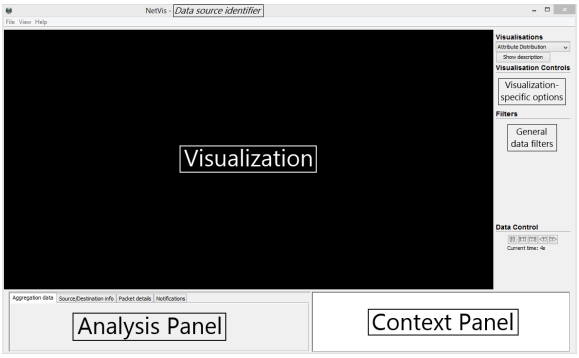
4. NetVis Visualizations

The user interface of the application focuses on the displayed visualization, which takes up the majority of the window, and can be maximised to fill the available space. Other panels occupy the right and bottom sections of the window, and are as follows.

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 visualisation, 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 fine-tune the speed at which data is processed and visualised, by pausing, doubling and halving the current data rate. The right panel is ultimately aimed at providing the user with options to adapt the shown data.

The function of the bottom panel is to show 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', the purpose of which is to show 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. A diagram of the application layout is included below.



4.1. Attribute Distribution

Displays a graph of the distribution of some packet attribute using its normalised value. It improves upon existing distribution visualizations by offering a dual representation. There is a logarithmic line graph layer useful for detecting spikes in the data and a layer of circles that indicates the actual distribution (no logarithm applied) through their area. Everything

is color coded to give an extra indicative of the volume of traffic (red means bigger than blue). All these ideas have a single purpose: make the analyst understand what is going on. // maybe you can change this into something that makes more sense (combine different ideas into one visualisation...) In this visualisation you can select a range for an attribute and the visualisation will apply a removable data filter that will only allow data in that range to get to the visualisations. **[make nicer]**

4.2. Data Flow

The Data Flow visualization applies the idea of parallel coordinates proposed by [Ins85] to network traffic. Following the principles layed out in [FM] 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.

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 [Mar09]. Our implementation addresses these concerns using animation. Lines representing new packets are randomly pertubed and move subtely. This makes it easier to spot whether a colourful 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.

For example, if multiple requests to a server came from a single MAC address through the same ports and using the same protocol, in a non-animated visualization they would all be represented by the same line. A user would erroneuosly interpret this as a single request. Adding randomness makes the pack of requests more visible without significantly impacting the accuracy of the data representation.

Hovering over one coordinate displays a scale for the attribute it represents. Clicking transfers you to the distribution visualisation showing the distribution of that particular attribute.

4.3. Spinning Cube

The Spinning Cube is a three-dimensional scatter plot. It 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.

The visualization is an implementation of an existing visualization tool known as the Spinning Cube of Potential Doom [Lau04], but offers additional control for the user. In particular, the axes can be chosen to represent arbitrary packet attributes.

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 pertubed. 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.

4.4. Traffic Volume

The Traffic Volume visualization is inspired by the 'FlowScan' graph [Plo00]. It is realised as a stacked bar chart which displays the volume of data arriving in each time interval. Each column is segmented into segments 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 itself.

To help distinguish between different protocols, colours are selected from a colour palette which provides colours from a qualitative colour scheme. The objective of this visualisation is to allow users to see at a glance if a particular protocol is being exploited in the network. The increase in both column height and colour proportion should draw attention to any protocol which becomes overburdened.

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

4.5. Heat Map

The Heat Map visualization uses a hexagonal map to display a set of communicating nodes. The activity of a specific machine in the network is represented by the colour of the corresponding hexagon. As the packets arrive, nodes representing the machines that send them "heat up" and, as the time passes, "heat down". The hexagonal grid is being filled in a compact manner - node by node.

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. Further information is presented for every client, such as its most commonly used protocol. This lets users do some basic data selection choices easily.

4.6. Activity Groups

Acting on the hexagonal grid, this visualization puts to use two important visual factors: size and proximity.

As the traffic in the network arises, the visualization groups together all the machines sending packets to a one specific other device. As more and more machines communicate with a specific address, more and more nodes appear around the hexagon representing that destination.

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. Therefore, both the size of the segment and its color transmit important information.

The placement of the nodes is automatic. The procedure employed always assures that there is enough space around a center and that no two nodes overlap.

This visualization helps detecting the machines that perform a server role in the network. It also allows a 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 server node.

5. Discussion

The main determinants of traffic in a network are 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 perspectives. Moreover, the underlying idea of our design is that better insight into the activity in the network can be gained by combining these perspectives.

Each visualization was chosen to fill a specfic purpose. The Heat Map visualization fills the need to gain 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 visualisations allow the user to detect pattern in the attributes of the packets.

Note that the visualizations combined 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 runs at the risk of obstructing the simplicity needed for ready understanding of a human observer. Our visualizations both use multi-dimensional systems (such as parallel coordinates) to give an overview, and use visualization with fewer dimensions but more detail. This encourages an understanding of network activity that is both broad and detailed.

To arrive at a complete understanding of the network, a single visualization will typically not be sufficient. An analyst will need to see different approaches to interpret the data, and will need to customize 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 [Mac87].

5.1. User Workflow

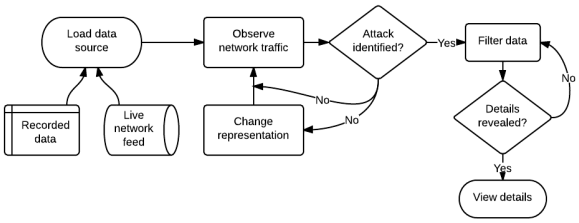
From the perspective of an analyst examining a network, a typical workflow would begin with considering all activity from a broad overview, using any number of visualizations. Once a network attack has been identified, it should be easy for the analyst to "zoom in" to the data of interest, by applying 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 on demand.

This workflow illustrates the visual design framework suggested by Ben Shneiderman [Shn98]:

Overview first, zoom and filter, then details-on-demand.

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 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.

The following is a graphical representation of the typical workflow expected of an analyst studying a network using NetVis.



5.2. Advantages

Existing network visualization solutions are typically single-purpose programs which present given data in one specific way (see for instance the catalogue in [Mar09]). 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

it complicated to use them simultaneously (for a discussion of these problems see).

The program presented here 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 be cluttered and lose its effectiveness. 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, but also for historical analysis, which makes it applicable for 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 also see how suspicious patterns show up in a visualization.

5.3. Limitations

The tools in our framework support only the analysis of time series data of packet captures. A multi-layer approach might help inform the user more concretely about the state of the network. In particular, information from intrusion detection systems, routers, and firewalls could be incorporated. The application in its current state does not analyse the content of packets.

5.4. Future Work

Our main avenue of future work is a more powerful data preprocessing system. Our current visualizations focus on giving the user a complete view on the network while allowing interactive filtering. However, if statistical inference is applied to the data, and historical observations are incorporated, this can help make visualizations more effective in pointing out suspicious patterns and anomalies in the data.

Further future developments will also give advanced tools for assisting users in investigating phenomena and better observing specific forms of network attack. For this, a library of applicable data filtering configurations should be available, and an adaptive data selection procedure should be offered.

6. Conclusion

In this paper we have presented..

References

[FM] FLIGG K., MAX G.: Network security visualization. 3

[Ins85] INSELBERG A.: The plane with parallel coordinates. *The Visual Computer 1*, 2 (1985), 69–91. 3

[Lau04] LAU S.: The spinning cube of potential doom. *Communications of the ACM 47*, 6 (2004), 25–26. 3

[Mac87] MACKINLAY J. D.: *Automatic design of graphical presentations*. Tech. rep., Stanford Univ., CA (USA), 1987. 4

[Mar09] MARTY R.: *Applied security visualization*. Addison-Wesley, 2009. 3, 4

[Plo00] PLONKA D.: Flowscan: A network traffic flow reporting and visualization tool. In *USENIX LISA* (2000), pp. 305–317. 3

[Shn98] SHNEIDERMAN B.: *Designing the user interface*. Pearson Education India, 1998. 4

[War12] WARE C.: *Information visualization: perception for design*. Morgan Kaufmann Pub, 2012. 1