Open access • Proceedings Article • DOI:10.1109/IAW.2005.1495940

Visualizing network data for intrusion detection — Source link

Kulsoom Abdullah, Christopher P. Lee, Gregory Conti, John A. Copeland

Institutions: Georgia Institute of Technology

Published on: 15 Jun 2005 - Systems, Man and Cybernetics

Topics: Intrusion detection system

Proceedings of the 2002 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY, 17–19 June 2002

Visualizing Network Data for Intrusion Detection

Kulsoom Abdullah, Chris Lee, Gregory Conti, John A. Copeland

Abstract— **As the trend of successful network attacks continue to rise, better forms of intrusion detection and prevention are needed. This paper addresses network traffic visualization techniques that aid an administrator in recognizing attacks in real time. Our approach improves upon current techniques that lack effectiveness due to an overemphasis on flow, nodes, or assumed familiarity with the attack tool, causing either late reaction or missed detection. A port-based overview of network activity produces a improved representation for detecting and responding to malicious activity. We have found that presenting an overview using stacked histograms of aggregate port activity, combined with the ability to drill-down for finer details allows small, yet important details to be noticed and investigated without being obscured by large, usual traffic. Due to the amount of traffic as well as the range of possible port numbers and IP addresses, scaling techniques are necessary to help provide this overview. We provide graphs with examples of forensic findings. Finally, we describe our future plans for using live traffic in addition to our forensic visualization techniques.**

I. Introduction

We hear about new network worms regularly in the news media and rather than slowing down, the worms are getting smarter and spreading more rapidly. One example is the Witty worm, which spread only the day after the vulnerability was announced [1]. This targeted an intrusion detection systems (IDS) product and deleted data from hard drives. The Sasser worm spread 17 days after the patch was released [2] and was similar to other new worms because it spread quickly and shut down major infrastructure components. There are continually new vulnerabilities available to target and there is no sign that this trend will slow down. There are also ongoing "zero-day" exploits, those without a software patch, that are particularly difficult to stop. Thus more needs to be done than solely relying on signature-based methods or waiting for the next patch. Compounded by increasing bandwidth due to higher capacity links, and the general growth of the Internet, especially home users with limited security, attacks are proliferating. IDS have more traffic to filter through limiting the number of signatures that can be checked. The Internet is the fastest growing information medium [3]. Also, processing power is surpassed by traffic bandwidth speed, thereby making it easier for an attacker to get through without being noticed, like the Sasser worm [4]. The biggest challenge is for an administrator to decide what is normal and what is abnormal. There are many signatures available, but this does not help for novel attacks. The field of information visualization can help to deal with processing this influx of information. According to Card, Mackinlay and Shneiderman, [5] information visualization is "the use of computer-supported, interactive visual representations of data to amplify cognition." Visualization tools are used to convey information from a set of data. A properly designed visual representation, as opposed to a textual representation, allows one to understand a greater amount of data in shorter time. Researchers in psychology have shown that humans can process pictures, a parallel process, faster than text, a serial process. Images facilitate understanding and insights that one would not have made if that same data were presented in other textual ways. It is also easier to remember as humans think and learn visually [6]. Here we present a prototype to aid in intrusion detection by visualizing network packet header data over time. We created a tool that is easy to understand, and allows those with minimal knowledge of network security to use it effectively. We have used scaling techniques on the data to reduce occlusion and used stacked histograms to efficiently visualize the data. The background information for our work is explained in section II, where an explanation of the data and analysis is given along with current related work in security visualization. Section III describes the system design, which includes the capture and graph process, how axis parameters are chosen, how the data is scaled and the threat models we use to illustrate the design. Results using forensic captures are given in section IV. Section V presents our future work and section VI discusses our conclusions.

II. Background and Motivation

First, we will give the background on the data available by packet capture and examine the fields under consideration. Next, we will explain the motivation for real-time and forensic analysis. Finally, related work in network visualization is given.

Abdullah, Lee, Copeland: Communications Systems Center, Electrical and Computer Engineering, Georgia Institute of Technology
Conti: Georgia Tech Information Security Center, College of Computing, Georgia Institute of Technology
This work was partially supported by a grant from Lancope, Inc.

ISBN 0-7803-9814-9/$10.00

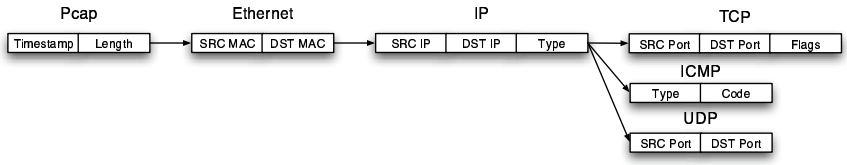

Fig. 1. Packet header fields parsed for statistics gathering

A. Packet Capture and Header Fields

Tcpdump and pcap [7] (and the functionally equivalent winpcap for Windows) are general purpose packet capture libraries. Any packet that is visible on the observed network segment while pcap is passively listening is collected and a pcap header is added to it. The pcap header includes the time the packet was received, and the total size. This data, along with rest of the data of the entire network packet is collected.

We carefully selected fields to maximize effectiveness and minimize processing overhead (figure 1). Timestamps are important for calculating the data statistics over time. Port numbers allow us to track the activity on the ports and help to tie attacks to vulnerable services. The packet sizes and Internet Protocol (IP) addresses give detailed flow information for the network.

B. Forensic and Real-Time Analysis

Forensic analysis of an attack takes much of the analyst's time. Any attack takes much longer to examine than it took to occur (a few seconds vs. a few days). Analysis of Honeynet captures have been time consuming but successful in detecting worms, and in finding compromised machines on the Georgia Tech network [8]. Our primary goal is to lessen the time of analysis. Specifically port activity is easier to observe with our visual prototype, as opposed to reading the packet capture text which is how forensic analysis usually occurs. General network monitoring is a combination of both forensic and real-time analysis. Production networks contain large percentages of legitimate traffic, but Honeynets are different because they include primarily illegitimate or suspicious activity. Honeynet captures are being used initially because they give a good benchmark to test the effectiveness of the tool. We will discuss how our monitoring techniques of forensic data relate to general network monitoring in section III-A.3.c which discusses time scaling.

C. Related Work

Originally visualizing network data was prepared with map layouts. The work by Becker [9] and Cox [10] is representative of earlier work done using glyphs to represent nodes and lines to represent links. Recent works with network layout methods to aid in security are as follows. Erbacher [11] uses glyphs and nodes where line type and circle attributes show information about the system and link. Haptics were used in the NIVA system [12] to visualize layout, and used a node placement algorithm based upon gravitational theory, electromagnetics, and fluid dynamics. VisFlowConnect uses parallel plots to show the connections between the inside network and the outside network [13]. Visualization can also be specific to an organization, such as Flodar in [14] for their server activity and SecureScope [15] which visualizes activity on network resources. Our prototype does not visualize the layout of a network. More emphasis is placed on identifying statistics of the network data and its trends over time. There are systems that use flow-based analysis for network activity visualization. Therminator is used with Lancope's IDS, StealthWatch [16]. Therminator is a non-signature based real-time visual tool using output from StealthWatch. Therminator is based on thermodynamic theories of energy, entropy and temperature [17] and keeps track of network state over time [18]. Together these tools provide a visual and text representation of the network flow. NVisionIP uses three views, in combination, to provide an overview of the network in [19]. An entire class B IP space is shown via a 256x256 grid matrix where each point represents the flows from each IP address. A user can select a subset of the grid to show histogram port plots for each address where well-known and dynamic ports are shown. Aggregate packet statistics are shown instead of flow information to be able to plot quantities in real time, as packets are received on the network, and without having to wait for flows to finish. We give an overview for all IP addresses and traffic in the network, giving the user the choice to filter for detail after the fact for real-time visualization and detailed analysis.

III. System Design

The system uses tcpdump to capture packets by listening in promiscuous mode on the network. These packets are then parsed for the needed values in the headers. Calculations are then made over time and these values are plotted to the graph. Open source Java libraries are used for generating the graphs [20].

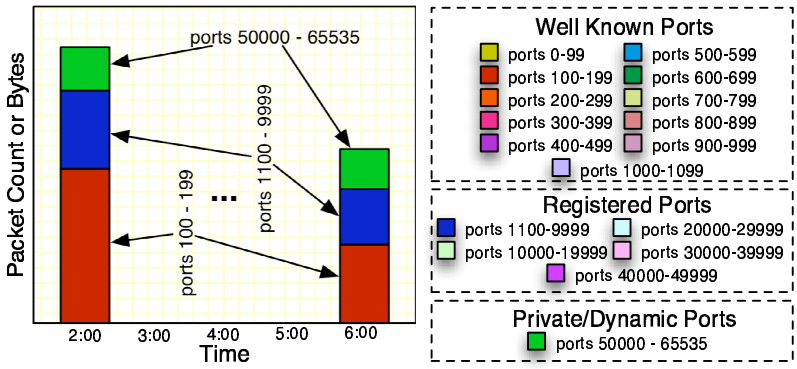

Fig. 2. Concept diagram of plotting technique. Port ranges are categorized and plotted on the graph

A. Design Goals

Our goal is to get an overall view of what is happening on the network. In the case of plotting port activity, we want to separate and arrange it to see important details without occlusion. More specifically we wish to provide context first, followed by the user specified detail in a drill down fashion. Seeing data over time will help to highlight any patterns or trends. Our scheme shows aggregate port quantity, instead of the more common flow count per IP. Some design goals are as follows:

- We want this system to be useful for both forensic and real-time analysis.
- Making the tool lightweight and not CPU intensive is another goal. Only header information is stored, not the payload, dramatically reducing archival storage requirements.
- Reducing the time it takes to learn how to use and understand the tool is another motivation. This helps to reduce the amount of time for understanding the visual results and speeds reaction time for the person observing the visualization.

Using scaling techniques will reduce occlusion, and still show the overall view of the network. In addition to IDS, the human eye can catch a pattern of a possible attack. This tool does not need to run in conjunction with an IDS, since it only needs to parse raw network data. It can complement the function of the IDS by allowing human cognition and observation to assist in identifying attacks and making subsequent decisions.

We used packet capture data from the Honeynet [8] at Georgia Tech for our visualization examples. The Honeynet at GT is directly connected to the Internet and has no production software therefore any incoming traffic is suspicious. The machines have different operating systems hence they can be targeted by a wide variety of attacks.

A.1 Histogram Plots

Histograms are easy to interpret and are good for visualizing a large data set because the data fits easily onto the plot due to stacking, the intra-bar, relative sizing is insightful, and comparisons with other bars on the chart are easily made based on their relative height [21]. Using multidimensional data to plot a relationship between two parameters over time can be visualized by using either a 3D chart or a 2D histogram. The disadvantage of a 3D view is in comparing patterns, widths and heights that are at various distances from the user, i.e., on the Z-axis, which can distort the perception [22]. Avoiding occlusion issues in 3D is difficult, and often requires user navigation. In a 2D stacked chart, it could be more difficult to focus on activity for a single port. To test the effectiveness of these techniques, we employed 2D visualizations against captures of real attacks.

A.2 Axis Parameters

The packet header fields can be categorized as ordinal (source and destination ports and IP addresses) and interval (packet count and total bytes). The horizontal axis is time and the vertical axis is used to display interval quantities. We chose to create stacked histograms mapping the horizontal axis to time and the vertical axis to interval values of packet count and total bytes. See figure 2 for an example.

A.3 Scaling and Occlusion Issues

A.3.a Overall Graph Occlusion . Ideally we want to fit all the information needed on the graph without overlap and disorder. High value quantities would either block or skew the relative scale, making it difficult to interpret results. To counter this, traditional methods from information visualization suggest rearranging the data, allowing the user to tilt, zoom, scale, and pan. Logarithms are one method of scaling data quantities but the cube root function was chosen as a substitute because it scales better with the range of values including zero with the additional benefit that values less than one, but greater than zero, are still mapped to positive values. (See figure 8 and explanation

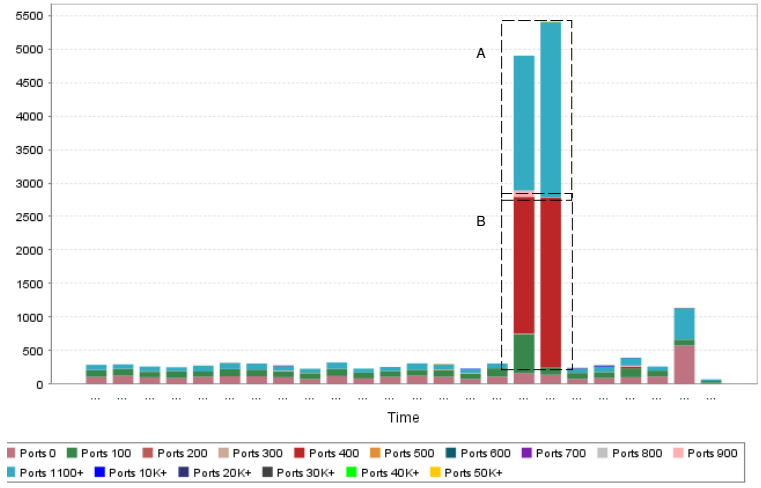

Fig. 3. Success of the Sasser worm and start of the worm traffic - count for every 30 minutes

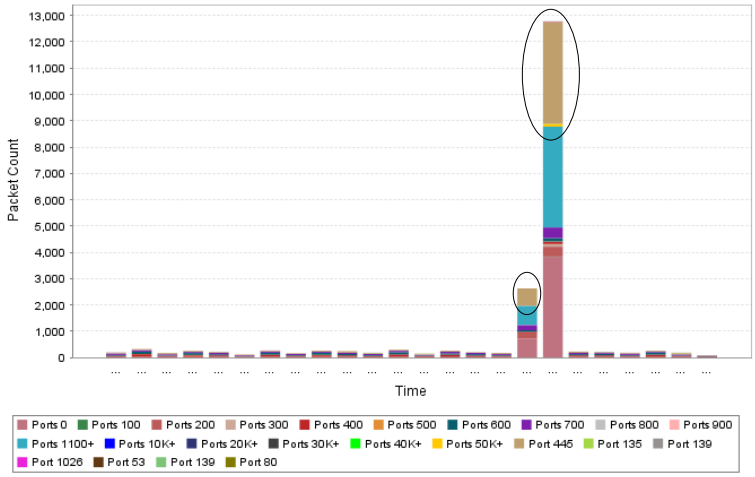

Fig. 4. Sasser graph with sorted out ports: 445, 135, 139, 1026, 53 and 80

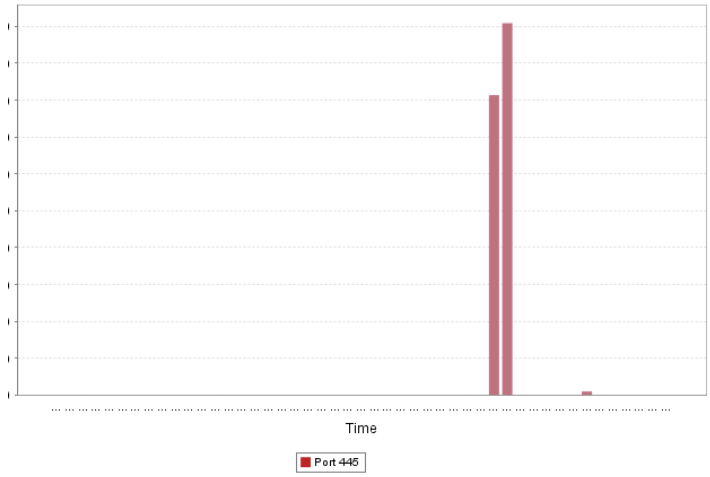

Fig. 5. Sasser graph of the p400 range filtered for focus. In this case traffic is seen to be from port 445

in the results section IV)

One of the weaknesses of the stacked histogram is that data variations lower in each column may make interpretation difficult. The blue and green variables in figure 2 represent this problem, making it hard to discern the green and red variables. One solution is a mouseover popup window that will give the value when the pointer is placed over a portion of the graph, along with other details. We show results in 3D at the end of this paper along with 2D histograms. Allowing user interaction and mouseovers have not yet been fully implemented.

A.3.b Scaling Port Numbers and IP Addresses. With a total of 65,536 possible port numbers and over 4 billion possible IP addresses (for the external network), scaling must be done. A pixel on an axis for each of these is impossible to fit or discern information. But we want the necessary information visible to ensure important detail is not lost. Common ports are based on what services a network has. We divided the rest of the ports into ranges to provide less occlusion. A user can see detail for a port range by selecting that on the graph. In our examples, using the Honeynet capture, we will use the "Top Ten" [23] most probed ports as the known common ports. For both Windows and Unix systems, the well-known and commonly assigned port range is 0-1023 [24]. We want to focus more on these ports because we believe that a majority of vulnerabilities and services occur in this range and most attacks start with this range to gain access. After separating the common ports, the remaining "well-known" ports are placed into bins of 100 (see figure 2) so that they do not cause an over abundance of port segments that are then difficult to read on the graph. We chose 100 because it is small enough to see the relatively high activity in this range. (In our plot, 0-1099) The registered port range is 1024 through 49151 and can be used by any application; these are also temporarily assigned for a connection attempt to a server. Traffic can occur here, but not as much as the well-known ports. These are divided up by groups of 10000. (In ours 1000-49,999) The private or dynamic ports are 49152 through 65535. Typically no service should be assigned these ports. They are still included in the plot as they are frequently used by Trojans and other malicious applications, but they do not need to be as fine grained; hence this is in one separate range [24]. If a user needs to see a more detailed breakdown of ports they may zoom in on a selected range. Plotting the port range is a far more tractable problem when compared to visualizing the entire 32 bit external IP address range. This is an open problem we would like to pursue for future work.

A.3.c Time Scaling. Scaling time is a function of desired plotting granularity which allows us to see different types of network activity. For forensic analysis or archives, the time sample can be varied so that differing trends may emerge. For example, slow scans occurring over long periods of time (year) vs. viruses and worms quickly infiltrating a network. In real-time traffic analysis, the time window should be long enough to capture a representative sample of packets, but not so long that fluctuations in port activity are hard to recognize.

Honeynet traffic generates good illustrations for our visualization since all of the traffic is "abnormal" and because a specific attack has been collected in the captures we have used. Of the techniques mentioned, time scaling needs more experimentation than the others to effectively be used with regular network traffic. We want just the right granularity; if it is too small, we can mistake variances in normal traffic to be anomalous, and if it is too large, again we will not see the variances that are anomalous. The other scaling techniques are applicable to both situations. For example, the Sasser capture did have other traffic in the background (representative of "normal" background traffic), and through IP, port and quantity scaling, we are able to see both that background traffic and the penetration and subsequent activity of the Sasser worm. With more development and testing, we would like to experiment with regular network traffic.

B. Threat Models

The following classes of attacks are the most common types that occur. To help demonstrate the effectiveness of the tool, packet captures of these attacks are visualized.

B.1 Network Scanning and Mapping

Scanning a network is commonly a precursor to an attack. A blueprint of the network can be made, finding active ports and IP addresses on the victim host. This is performed by sending probe packets to groups of desired IP addresses. If a response is received, then we know the respective ports and hosts are active. While less aggressive (or more subtle) naive scans are relatively easy to detect, scans on common ports are difficult to detect in the midst of legitimate network traffic [25]. Typically, the packets are small, and few in number. Such scans are difficult to detect, particularly very slow scans. If non-existent hosts and unused ports are probed it is easier to detect because we expect little or no traffic to those destinations. Visually we will see values plotted for port ranges that have not shown up before as in figure 6.

B.2 Viruses, Worms and Trojans

There are two primary motivations for those who create worms. One is for Distributed Denial of Service (DDoS) and the other for installing Simple Mail Transfer Protocol (SMTP) servers that receive connections on high ports and relay mail (outbound connection) to port 25 of the destination server, normally used for Spam. Since traditional methods of propagating Spam are being denied, worms are being used [26]. This allows a malicious sender to gen-

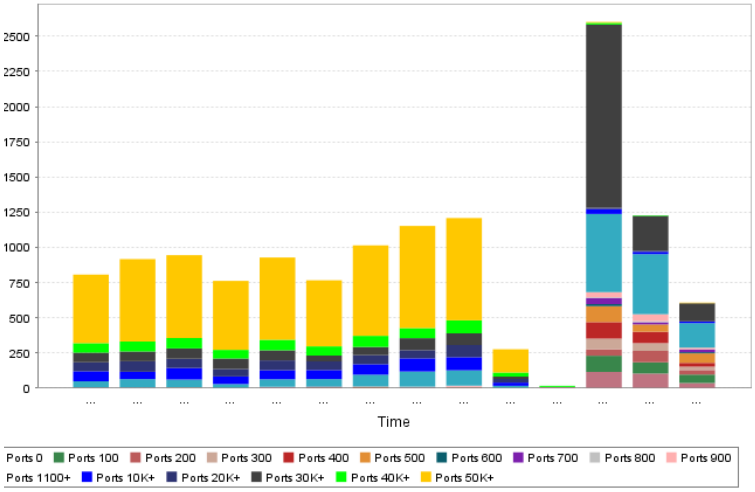

Fig. 6. Network scan from honeynet, packet count every 2 minutes. Pattern or ports probed over time is seen for this scan

erate Spam from all attackers' compromised systems [27]. Infecting a system usually occurs by exploiting a vulnerability on a system. Once a system is infected, then it will scan for other machines and infect those in a similar way. A burst of activity will be seen on the vulnerable port. If penetration is successful we typically see odd behavior on a backdoor port which is used to scan for other hosts to infect, see figure 3.

B.3 Backdoors and Rootkits

A backdoor is installed after the successful takeover of a system to help maintain control. It opens a port on the system to send and receive traffic, thereby maintaining a hidden entry point. Unusual activity on a normally active port or non-active port is readily apparent. Packet count and total size increase in a visualization if a significant amount of data is going in and out of the backdoor. Similar to backdoors, rootkits go one step further and replace existing application binaries instead of running as a new application like a backdoor does, e.g. a modified telnet program can be a rootkit. Another covert method to access a port is with IRC bots, which have been used for DDoS attacks. The binary for the bot is small and can easily be installed on someone's system without their knowledge. Commands can then be passed to the bot. Typically the bot is told to attack a given IP address. The servers and channels the bots are placed on can be difficult to find when the channel is set to hidden. Detecting unknown bot activity requires a check on the default IRC port 6667 or other chose ports [28]. In this case, there would be a rise of packets on the IRC port or other chosen ports used to transmit the bot traffic (see figure 7).

IV. Results

Our prototype helps to show the pattern and quantity of traffic on ports and systems where unknown attacks can be identified without a signature. Because of the general characteristics of how common attacks behave, this is a useful scheme. Observing port behavior is both a good indicator for a precursor of an attack or an actual attack. Our approach did not help to detect malicious behavior during the actual compromise on the local system (ie., when someone gained root access). Also, the prototype does not show network topology, layout and link information. Flow information is also not considered, just individual packet header data. Subsequently we get the port information instantaneously, and not wait for a flow to finish, especially that of long flows.

The following graphs are of Sasser worm [8] and botnet [29] activity from the Honeynet. These results are used to illustrate the efficacy of the system, such as scaling, and overview/detail.

The first example considers the Sasser worm. Figure 3 shows normal probes and chatter that usually occur. The spikes indicate an increase of packet count for two port ranges. In region B, the port 400 range in red show the start of Sasser activity which had at that point successfully broken into the system. The tcpdump log showed that all of this was from port 445. Region A, which represents ports 1,100-10,000, was a response port to port 445. Again, checking the log indicated that this was port 2552. This example shows that a user can be guided to notice activity in these port ranges, in the midst of usual traffic. Without having to look at the log, the user can focus on the port 400 range by filtering the graph to show only the port counts for 400 to 500. This is shown in figure 5, where we can see that this is port 445.

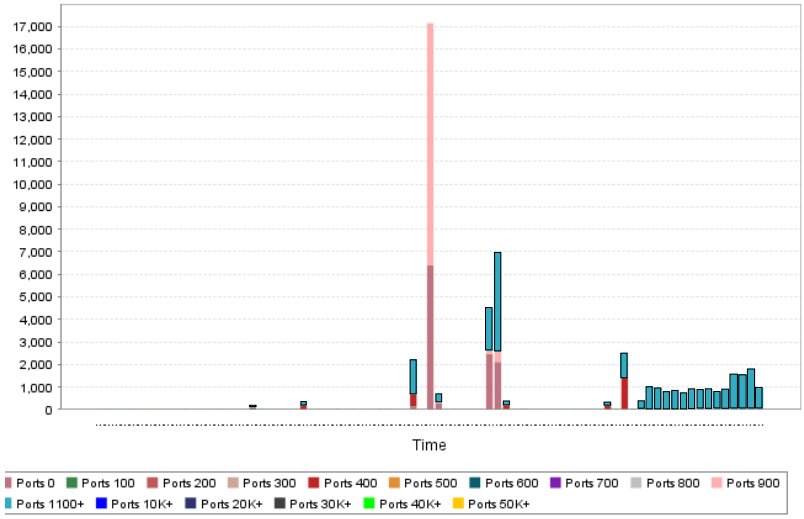

Fig. 7. A system compromise by a botnet followed by the victim joining the botnet and transmitting traffic. Packet count every 15 minutes.

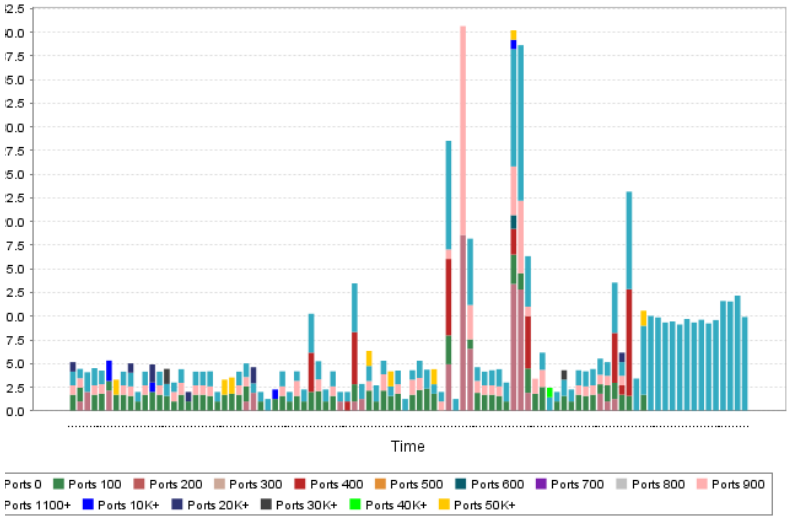

Fig. 8. Scaling packet count using cube root for botnet traffic capture

Ports that are commonly used on a network can be sorted out from the port ranges and plotted on their own. Doing this helps to both focus on the usual active ports for any abnormality and reduce the count from the port ranges. When the count is reduced, we can then focus on the rest of the ports where we do not usually expect traffic without being occluded by the high count of the active ports. We can see the same traffic capture of Sasser in figure 4 where ports from the top ten probed ports list are sorted out from the rest. At that time ports 445, 135, 139, 1026, 53 and 80 were getting probed on the Honeynet, which is why we choose to single these out for this illustration. The circled region shows where port 445 is plotted now. Compared to figure 3, we can focus on port counts in the ranges minus the value of port 445 count obscuring the counts.

In circled region of figure 9, we can see the start of p400 (port 445), to appear jagged, compared to the original graph count of every 30 minutes. A smaller time sample helps to notice behavior sooner, which is important for a regular network that needs to react quickly, but at the cost of a smaller picture. One example is the network scan that occurred in 30 minutes (figure 6) where we needed a smaller time scale to quickly notice this activity.

We can see the ports with smaller counts more clearly and its pattern over time using a cube root. In the botnet graph, because of the high count of bot packets skewing the scale (around 17,000), all of the other values are barely visible. In figure 8, we can see the other port activity now.

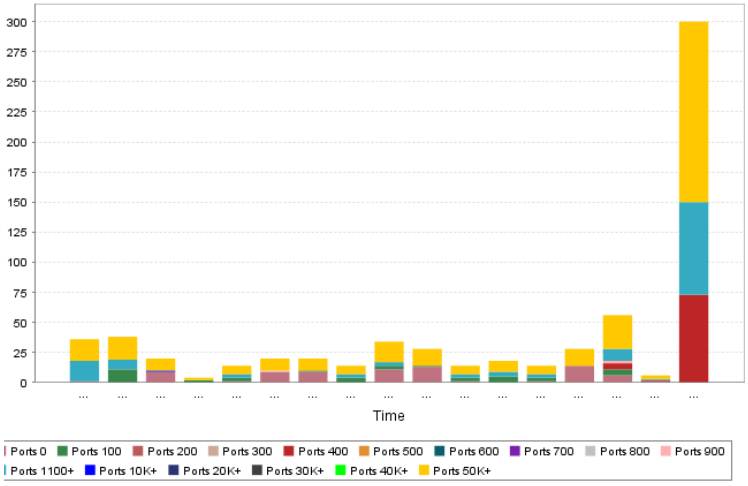

Fig. 9. Same packet count of Sasser using a smaller time scale of 5min

Those were the vulnerabilities the bots were trying to exploit on the Honeynet system, until they were successful and gained access.

V. Future Work

Our goals are to add an interactive GUI and with the discussed visual tools implemented to help facilitate more findings into the data. Having a plot with a scaled outside IP address for a multidimensional plot will be helpful for real-time traffic, as we need to know both incoming and outgoing traffic flows. We would like to consider ICMP, TCP flag packets and TTL values. These can give useful information and possible patterns as attacks can be composed of these other types of packets. Finding other potential packet and traffic behavior would also be of interest.

VI. Conclusion

This paper explained motivations for using information visualization with network data for security purposes. The research performed to come up with our scheme included an explanation of the packet header data and the two modes of real-time and forensic monitoring. We have worked with forensic data and showed these results. Histograms are helpful for determining trends and evaluating quantity. The parameters for the axis were given as well as scaling issues and solutions. Brief explanations of the general types of attacks that networks encounter were also given to explain how this scheme would be helpful. Current and future work on this prototype was presented followed by the benefits of this prototype and graph results. We presented some techniques that process network data statistics and coupled them with information visualization techniques. These techniques help in detecting anomalies quickly and identifying the cause.

References

[1] C. Shannon and D. Moore, "The spread of the witty worm." CAIDA. Updated April 2004.

[2] W. Arbaugh, "A patch in nine saves time?," *Computer*, pp. 82–83, June 2004.

[3] P. Lyman, "How much information 2003," October 2003.

[4] P. Jungck and S. S. Y. Shim, "Issues in high-speed internet security," *Computer*, vol. 37, pp. 36–42, July 2004.

[5] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, 1999.

[6] C. Ware, *Information Visualization: Perception for Design*. California: Academic Press, 2000.

[7] "Tcpdump/libpcap." Last updated April 3, 2004.

[8] J. Levine, R. LaBella, H. Owen, D. Contis, and B. Culver, "The use of honeynets to detect exploited systems across large enterprise networks," in *Proceedings of the IEEE Workshop on Information Assurance*, IEEE Systems, Man and Cybernetics Society, (West Point, NY), pp. 92–99, June 2003.

[9] R. Becker, S. Eick, and A. Wilks, "Visualizing network data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, pp. 16–28, March 1995.

[10] K. Cox and S. Eick, "Case study: 3d displays of internet traffic," in *Proceedings of Information Visualization (INFOVIS)*, IEEE Computer Society, pp. 129–131, Oct. 1995.

[11] R. Erbacher, K. Walker, and D. Frincke, "Intrusion and misuse detection in large-scale systems," *Computer Graphics and Applications*, vol. 22, pp. 38–48, January/February 2002.

[12] K. Nyarko, T. Capers, C. Scott, and K. Ladeji-Osias, "Network intrusion visualization with niva: an intrusion detection visual analyzer with haptic integration," in *10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS '02)*, pp. 277–284, March 2002.

[13] X. Yin, W. Yurcik, Y. Li, K. Lakkaraju, and C. Abad, "Visflowconnect: Providing security situational awareness by visualizing network traffic flows," in *Proceedings on the Workshop on Information Assurance (WIA04)*, 2004. held in conjunction with the 23rd IEEE International Performance Computing and Communications Conference (IPCCC 2004).

[14] E. Swing, "Flodar: Flow visualization of network traffic," *IEEE Computer Graphics and Applications*, vol. 18, pp. 6–8, Sept.-Oct. 1998.

[15] A. D'Amico and S. Salas, "Visualization as an aid for assessing the mission impact of information security breaches," in *Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX)*, vol. 2, pp. 18–20, April 2003.

[16] "Stealthwatch+therminator." Lancope, 2004. http://www.lancope.com/.
[17] D. Ford, "Application of thermodynamics to the reduction of data generated by a non-standard system," tech. rep., Department of Physics Naval Postgraduate School, Monterey, CA., Feb. 2004. Available at http://www.arxiv.org/abs/cond-mat/0402325.
[18] S. D. Donald and R. V. McMillen, *Therminator2: Developing a Real Time Thermodynamic Based Patternless Intrusion Detection System*. PhD thesis, Naval Postgraduate School, Monterey, California, 2001.
[19] W. Yurcik, K. Lakkaraju, J. Barlow, and J. Rosendale, "A prototype tool for visual data mining of network traffic for intrusion detection," in *3rd IEEE International Conference on Data Mining (ICDM) Workshop on Data Mining for Computer Security (DMSEC)*, 2003.
[20] D. Gilbert and T. Morgner, "Jfreechart," 2003 - 2005.
[21] D. Keim, M. Hao, and U. Dayal, "Hierarchical pixel bar charts," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, pp. 255–269, July-Sept. 2002.
[22] R. Spence, *Information Visualization*. England: ACM Press, 2001.
[23] "10 most probed ports." Distributed Intrusion Detection System, June 2004. http://www.dshield.org/topports.php.
[24] "Port numbers." Internet Assigned Numbers Authority. Last updated 28 May 2004.
[25] E. Skoudis, *Counter Hack: A Step-by-Step Guide to Computer Attacks and Effective Defenses*. New Jersey: Prentice Hall, 2001.
[26] M. Mimoso, "Experts ponder spam, worm-writing connection." Search Security, Nov. 2003.
[27] N. Krawetz, "Anti-spam solutions and security, part 2." Security Focus.
[28] "Netsys.com: The intelligent hacker's choice." DDoS article.
[29] "Honeynet project: Scan of the month," 2004.

ISBN 0-7803-9814-9/$10.00