Passive Visual Fingerprinting of Network Attack Tools

Gregory Conti and Kulsoom Abdullah

ABSTRACT

This paper examines the dramatic visual fingerprints left by a wide variety of popular network attack tools in order to better understand the specific methodologies used by attackers as well as the identifiable characteristics of the tools themselves. The techniques used are entirely passive in nature and virtually undetectable by the attackers. While much work has been done on active and passive operating systems detection, little has been done on fingerprinting the specific tools used by attackers. This research explores the application of several visualization techniques and their usefulness toward identification of attack tools, without the typical automated intrusion detection system's signatures and statistical anomalies. These visualizations were tested using a wide range of popular network security tools and the results show that in many cases, the specific tool can be identified and provides intuition that many classes of zero-day attacks can be rapidly detected and analyzed using similar techniques.

Categories and Subject Descriptors
H.5.2 [Information Systems]: Information Interfaces and Presentation - User Interfaces

C.2.3 [Computer-Communication Networks]: Network Operations: Network monitoring

C.2.0 [Computer-Communication Networks]: General - Security and Protection

General Terms
Security

Keywords
network attack visualization, visual fingerprinting, application fingerprinting, passive fingerprinting, operating system fingerprinting, information visualization

VizSEC/DMSEC '04, October 29, 2004, Fairfax, VA, USA.

1. BACKGROUND AND MOTIVATION

Classical algorithmic intrusion detection systems (IDS) rely upon machine-detected signatures and statistical anomalies to discover intrusions. While great progress has been made, there exists an unacceptable rate of false positives and false negatives in such systems. By allowing the network analyst to continually observe network traffic in a highly efficient manner, analysts develop an intuitive feel for the usually legitimate and sometimes anomalous activities on their network in a way that augments more traditional systems. False positives and false negatives become of an entirely different character. For example, an anomaly-based intrusion detection system may be slowly trained over time to overlook malicious activity and a signature-based intrusion detection system generally will not detect new attacks unless they exist in its signature database. While it is still possible to fool a network analyst or system administrator, we argue that properly designed visualizations enhance the capabilities of the human in such a way that greatly complicates the efforts of an attacker.

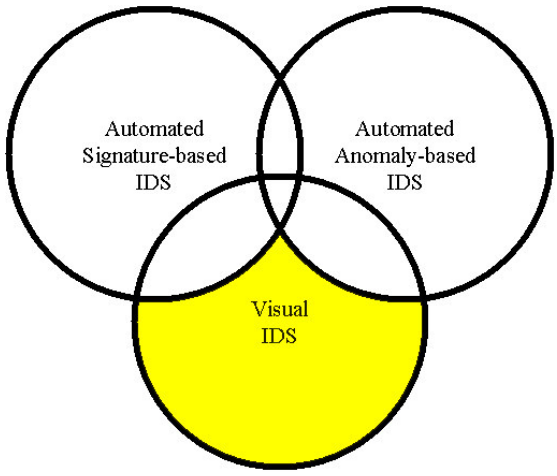

Figure 1: Mutually Supporting Capabilities of Intrusion Detection Systems

A human will inherently detect different signatures and different anomalies. Historically, such fields as machine vision and machine learning have shown that

the tasks in which a machine stands out are typically entirely different from those in which humans excel. This is a strong argument for research into visual intrusion detection. A would-be intruder must fool both the human *and* the machine. Figure 1. illustrates the mutually supporting capabilities of these complementary intrusion detection systems. The shaded area is the increased coverage provided by a visual intrusion detection system. We believe that classical intrusion detection systems working symbiotically with a visualization-enhanced human will outperform algorithmic systems operating alone. By bringing humans more directly into the intrusion detection loop, correct visualizations can tap into the high bandwidth visual recognition capabilities of the human cognitive system and help address the serious problem of false positives and false negatives that exists today. As experienced amateur radio operators can immediately identify digital signals based upon their audio characteristics, so can human analysts visually identify network attacks even if they do not exactly match the precise signatures or statistical anomalies of past attacks. For a visual intrusion detection system to be effective it must focus on tasks that cannot be easily performed by an automated system. Section 3.1 lists the subset of these tasks we focused on for this work.

In this paper we argue the following benefits of visualization:

- Specific attack tools, and to a lesser-degree their host operating system, can be passively identified by their visual signature. This aids law enforcement forensics, provides insight into an attacker's methodology and experience level and helps allow the network defender to initiate appropriate responses.
- Some stealthy attacks are resistant to detection by traditional intrusion detection systems, but are readily visible using appropriate visualizations.
- Visualization techniques can be used that require little state and are remarkably resistant to overload caused by high volume network traffic and resource consumption attacks that can incapacitate traditional intrusion detection systems. The lightweight, fixed memory requirements of some of these visualizations allow visual intrusion detection systems to operate for long periods of time.
- With appropriate visualizations of network traffic there is no such thing as a false positive or false negative as typically defined in the classical intrusion detection system domain. Human operators can still be fooled, but the character of human decision-making is entirely different from that of machines. Consider the problems encountered when attempting to filter spam. A human can immediately identify most spam, where computer algorithms provide only limited detection. This diversity provides overlapping capabilities that, when combined, provide greater effectiveness than an IDS or human operating alone.
- By their very nature, most zero-day (never seen before) attacks do not match existing intrusion signatures. Visualization techniques can provide clues to impending attacks and facilitate quick response analysis. For example, the scanning of a newly released worm would be readily apparent.
- Distributed scanning and slow scanning (hours-weeks) can be effectively detected.
- While legitimate network traffic can cloud the visual intrusion detection environment of a given network, many attacks are still readily apparent through the noise.

The goal of this paper is to explore and defend these claims as well as systematically examine the strengths and weaknesses of several visualization techniques as they are applied to network intrusion detection.

Section 2 examines the current state of the art in this area. Section 3 explores the task-driven development of candidate visualizations and their usefulness toward analysis of attack tools. Section 4 describes the results from a series of controlled laboratory experiments. Section 5 presents our conclusions and directions for future research.

2. RELATED WORK

The primary contributions of this work include the demonstration of the efficacy of fingerprinting common attack tools, the ability to provide rapid insight into the attacker's operating system type and the possible lineage of the code in use, the ability to detect some classes of stealthy attacks and the ability to detect slow scans despite the visual noise of legitimate traffic. Related work falls into several main areas including current network security visualization research as well as application and operating systems fingerprinting. While network visualization and intrusion detection are relatively mature areas, there is a limited body of work covering network security visualization. Representative recent research includes analysis of the stability of Internet routing[1,2], analysis of stepping stone pairs[3], monitoring the security status of large networks[4], mapping of the Internet[5,6], application of statistical methods for intrusion detection[7], intruder behavior characterization[8], worm propagation[9], rapid prototyping[10], TCP/IP sequence number generation[11,12], haptic integration [13] and the construction of a toolkit for visual intrusion detection[14]. Availability of security-centric commercial and open source/free visualization systems is likewise limited. Representative examples include: SecureScope[15], StealthWatch + Therminator [16], Ethereal[17], Etherape[18], Netstumbler[19], 3DTraceroute[20] and XTraceroute[21].

3. NETWORK SECURITY VISUALIZATION PROCESS

We chose a comprehensive approach to visualize network attacks that included consideration of all TCP, UDP, IP and Ethernet header fields as well as many features that can be derived from this data. After examining the data available, we considered a broad range of visualization techniques from classic information visualization literature and current research in network security visualization. Finally, we examined a variety of network attack tools from the Top 75 Network Security Tool List produced by fyodor, the creator of *nmap* [22]. This list was constructed based on a May 2003 survey of *nmap* developers. From this consideration of data, visualization techniques and security tools we constructed a series of experiments to test the hypothesis that these tools could be effectively fingerprinted. We understand that a proficient attacker can evade many of these techniques, primarily due to lack of authentication in today's network protocols, but feel that they remain some of the most effective techniques at present. Such is the case with much of the information security field, for now and into the foreseeable future it will likely continue to be an on going battle of one-upmanship.

3.1 Task Analysis

Specifically, we wished to design visualizations that would effectively visualize passively captured packets in real time in order to accomplish the following goals:

- fingerprint popular attack tools
- provide insight into the attacker's operating system
- detect stealthy attacks (TCP evasion techniques in particular)
- provide insight into future zero-day attack detection systems
- detect slow scans
- detect distributed scans
- detect attacks despite the visual noise of legitimate traffic
- supplement traditional signature and anomaly based intrusion detection systems so as to reduce the overall number of false positives and false negatives.

The degree to which we were able to achieve each of these goals is discussed in the results section.

3.2 Exploration of Available Data

3.2.1 Direct Data

Passive sniffing tools such as *tcpdump*[23] and *snort*[24] make available all of the header information contained in packets as they traverse an Ethernet collision domain. To constrain the problem, we limited our consideration to the most common protocols: Ethernet, TCP, IP and UDP, believing these would

provide representative insights that could generally be applied to other protocols. The packet formats of these protocols are well documented. By carefully considering the relevancy of the direct data available and the applications to be analyzed it is possible to construct candidate visualizations. As a simple example, if one is attempting to fingerprint a simple port scanning program it is useful to visualize the source IP, destination IP, source TCP port, source UDP port, destination TCP port and destination UDP port. In addition, the ability to analyze attack tools using direct data from a sniffing program is enhanced by the use of feature construction. Feature construction is discussed in section 3.2.2.

3.2.1.1 Link Layer (Ethernet)

Link layer headers are typically created by the node one link distant from the receiving node. For this information to be compromised a nearby node must also have been compromised. Link layer information is particularly useful for detecting anomalous behavior initiated on a local network segment: for example, to detect 802.11b wireless network abuse, address resolution (ARP) spoofing and attempts to sniff across collision domains in a switched network (e.g. switch flooding, ARP redirects and MAC address spoofing). For purposes of this paper we chose to consider source MAC address, destination MAC address and the overall length (in bytes) of the Ethernet frame.

3.2.1.2 Network Layer (IP)

Network layer packets are used for host-to-host communication across the Internet and have been subject to much abuse by malicious entities. While we chose to focus our visualizations on the source and destination IP address fields there are many areas for future work. Of particular interest are the time to live (TTL) field and the fragmentation offset which can be used for such activities as detecting Honeynets[25] and insertion and evasion attacks to bypass intrusion detection systems[26].

3.2.1.3 Transport Layer (TCP and UDP)

Transport layer protocols provide process-to-process connectivity across the Internet. Both TCP and UDP use the notion of ports to support this connectivity. Due to the fact that ports are fundamental to Internet connectivity and that many attack tools probe these ports in an attempt to discover vulnerabilities we chose to include the source and destination ports for TCP and UDP for our visualizations. For future work we leave the visual examination of TCP sequence numbers and flags.

3.2.1.4 Application Layer

Application layer headers and data provide a great deal of information about the nature of attacks, but due to the wide variety of application layer protocols we chose to limit our visualization research to raw hex and printable ASCII decodes of this data. There is a great deal of research potential in the visual examination of application layer data. As an example, many zero-day network-based buffer overflow attacks will likely have distinct visual signatures.

3.2.2 Feature Construction

Feature construction allows one to add new attributes to the packet capture dataset constructed based upon the captured data as well as information from the network security domain[27]. We chose the following candidate features as useful for visualization:

- Cumulative source and destination ports (by protocol) used for a given period.
- Cumulative source and destination IP addresses (by transport protocol) used for a given period.
- Sequence of packets and ports (by transport protocol and length) used during a given period.
- The notion of home network and external network. This allows the analyst to define the locality of their services and packets. This also facilitates graphing that is relative to internal and external networks and is similar to what is seen in Snort configuration [28].

While we chose just the subset of possible features listed above, for future work we would like to consider other possibilities. The literature of intrusion detection has a rich body of work to draw upon when considering visual network intrusion detection [29]. Generalizing the feature construction survey by Brugger [30] potential candidates include visualization of the time variant nature of network traffic (e.g. duration of

connections & services, timestamp of packets), conformance to protocols, IP and TCP flag usage, number and type of packets/protocols, number and type of connections, number of resent and duplicate packets, amount of fragmentation, services available/used and errors encountered. To this list we would add the following features as potentially interesting candidates for visualization: stated checksum vs. actual checksum (for each layer), stated length vs. actual length (for each layer), transformation from time domain to frequency domain (using Fourier transforms), time between packets, probable operating system, additional IDS evasion techniques (dropped packets, packet overlap, out of order packets, known malformed packet types [31]) and the likelihood of human vs. machine operation based on packet timing. Finally we believe the use of composite variables and the general application of the following statistical measures will prove useful: duration, frequency, quantity, ratios, percentages, deviation from independence, influence, standard deviation, variance, average, mean, median and mode.

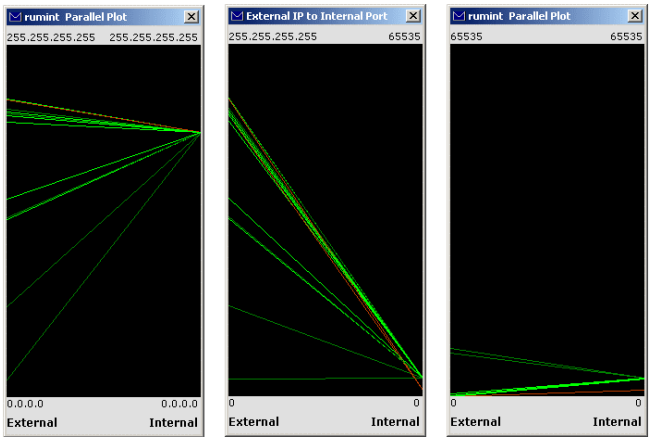

Figure 2: External IP to Internal IP (left), External IP to Internal Port (center) and External Port to Internal Port (right) Parallel Coordinate Plots

3.3 Visualizations

3.3.1 Introduction

There are a wide variety of potential visualizations that can be used to display network traffic data in a way that is meaningful for security analysis. The comprehensive surveys by Tufte [32,33,34] and Spence [35] cover classical information visualization techniques in detail and are very useful for inspiration. From this range, we selected the parallel coordinate plot [36,37,38] for its strength in showing relationships within hypervariate datasets as well as its proven usefulness in the intrusion detection domain [39]. We also selected a technique similar to that used in the Seesoft system to help illuminate the time variant aspect of the network traffic [40].

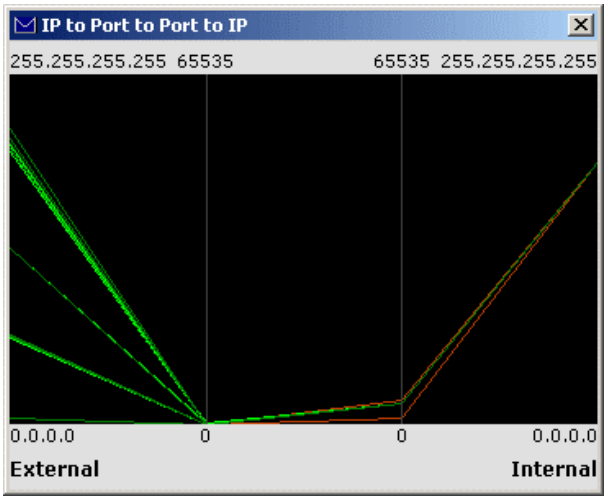

Figure 3: External IP to External Port to Internal Port to Internal IP Parallel Coordinate Plot

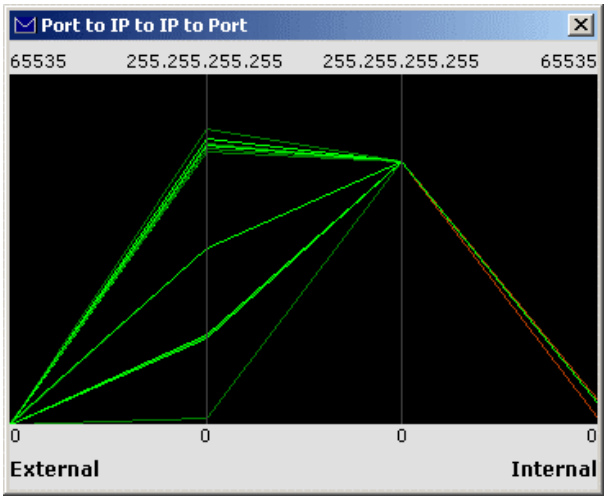

Figure 4: External Port to External IP to Internal IP to Internal IP Parallel Coordinate Plot

3.3.2 Parallel Coordinate Plots

For parallel coordinate visualizations, we limited our variables to: source IP, destination IP, source port, destination port and protocol type (TCP or UDP, inbound or outbound from home network). The parallel coordinate plots below extend previous work in the VisFlowConnect system[4] by applying additional dimensionality, alternative encoding techniques, real-time packet capture and focused attack tool specific

data. We used parallel coordinate plots of the hypervariate data in the following combinations:

- External IP to Internal IP (figure 2, left)
- External IP to Internal Port (figure 2, center)
- External Port to Internal Port (figure 2, right)
- External IP to External Port to Internal Port to Internal IP (figure 3)
- External Port to External IP to Internal IP to Internal Port (figure 4)

Ports and IP addresses are plotted on a continuous scale. Alternatively we could have chosen to treat those fields as categorical or ordinal data by allocating space to only those data points encountered in the real-time network traffic. This proved to be in conflict with our design goal of a lightweight system as a buffer and significant screen redrawing would be required. Ultimately we chose the continuous scale in an attempt to provide an effective overview of network activities and allow the technique to scale to very large networks. The advantage is that the entire 32-bit IP address space can be viewed at a glance. We understand that this comes at the cost of coarser resolution. Future systems will incorporate improved zoom, filter and details-on-demand capabilities to complement this technique. Color is mapped to the protocol in use as well as the direction (to/from home network). The left side of the plot is considered to be the external network and the right side is considered to be the home network (as defined by the network analyst)

3.3.3 Scrolling Packet Plots

We chose two variants of a scrolling visualization to provide insight into the time-variant nature of the network traffic. While our parallel coordinate plots were designed to show the overall relationships between source and destination IP addresses and TCP/UDP ports we wished to create visualizations that would better show a running sequence of packet data as they arrived at our observation point.

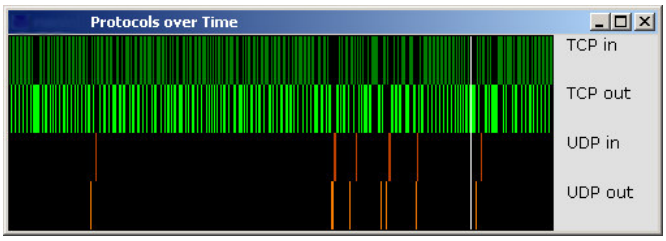

Figure 5: Scrolling Protocol Type over Time

3.3.3.1 Simple Categorical Scrolling Plot

The first plot mapped packet protocol type (TCP/UDP) and direction (inbound/outbound from the home network) to specific colors and vertical locations on the screen. (figure 5) As each packet was captured it was plotted as a small vertical line, one pixel wide, on the graph. Its position on the horizontal axis was incremented (to the right) by one pixel with each packet. When network traffic caused the plot to move to the extreme right it wrapped around, beginning at the extreme left. A vertical marker line spanning the entire plot window was used to indicate current position. This marker was placed one position ahead of the current plot.

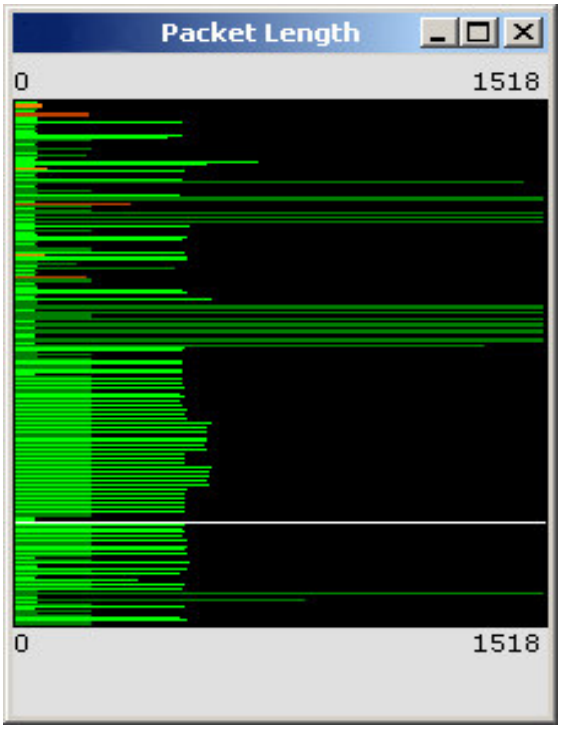

Figure 6: Scrolling Packet Length over Time

3.3.3.2 Packet Length Scrolling Plot

This visualization is similar to that used in the Seesoft system. Packet length is mapped to a variable length horizontal line. (figure 6) The length of this line was calculated from the raw size of the Ethernet frame (in bytes) divided by the maximum allowable size of the frame according to the Ethernet protocol (1518 bytes). Each packet detected caused a new line to be calculated and drawn one pixel lower in the view window. This had the effect of removing the aspect of time delay between packets from the display. Similar

to the simple categorical scrolling plot described in 3.3.3.1 color was mapped to the protocol type and direction of the traffic. As network traffic caused the display to fill, the plot wrapped around to the top of the display window. A horizontal marker line was used to indicate the current position.

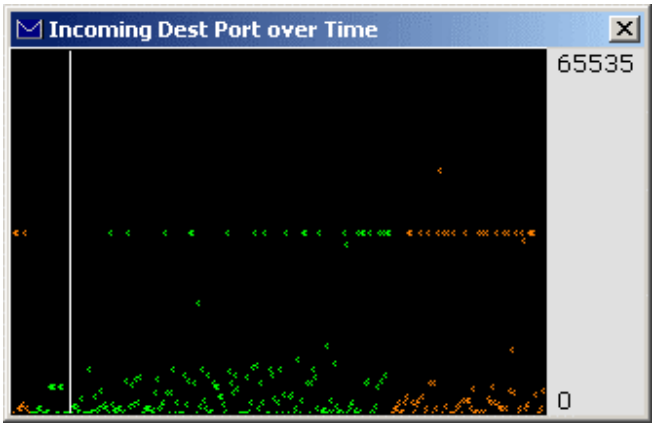

Figure 7: Incoming Destination Port over Time

3.3.3.3 Incoming Port Scrolling Plot

The final visualization we explored was designed to show the sequence of inbound destination ports over time. (figure 7) The vertical axis is the port number and the horizontal axis is the arrival sequence of packets. As each packet arrives a small marker is plotted according to its destination port number on the vertical axis. Each subsequent packet is plotted one pixel to the right of its predecessor. When the current plotting position exceeds the rightmost display position the display wraps. A vertical white line is used to display the current position. Marker colors are mapped to the protocol of the inbound packet (UDP or TCP).

In summary, the following table shows the mapping of features to visualization.

	Port to Port	IP to IP	IP to Port to Port to IP	Port to IP to IP to Port	Categorical Scrolling	Packet Length Scrolling	Incoming Port Scrolling
Internal Ports	√		√	√			√
External Ports	√		√	√			
Internal IP		√	√	√			
External IP		√	√	√			
Protocol Type	√	√	√	√	√	√	√
Packet Direction	√	√	√	√	√	√	√
Packet Sequence					√	√	√
Port Sequence							√
Raw Packet Length						√	
Cumulative Internal Ports	√		√	√			
Cumulative External Ports	√		√	√			
Cumulative Internal IP		√	√	√			
Cumulative External IP		√	√	√			
Packet Arrival Sequence					√	√	√

Figure 8: Visualization vs. Feature Displayed

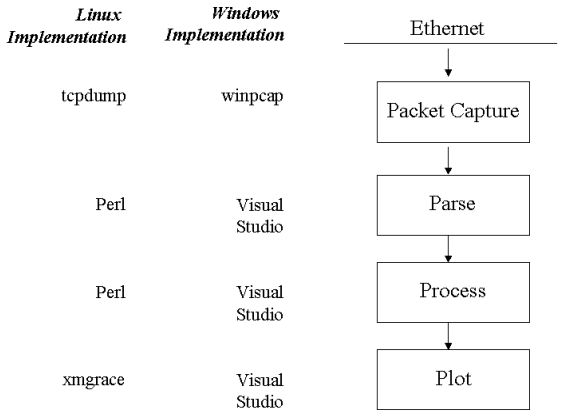

Figure 9: System Architecture

3.4 System Overview

We went through several iterations when developing our system including the use of existing packet capture tools such as *tcpdump* and *snort* piped to Perl to parse, process and graph the data in near-real time (figure 9, left column). While reasonably effective, we found this technique was less suitable for real-time plotting and interaction. Ultimately we used the *winpcap* library and *Microsoft Visual Studio* to directly access the packet capture library and build applications (figure 9, right column). This combination excelled at rapid GUI development and visualization construction without excess overhead. While we could have used *GTK+* or *QT*, we felt that we would make more rapid progress using a visual development environment and port our work to the *Unix* domain at a later stage. Our final system captures packets and creates the visualizations in real time. It supports both promiscuous and non-promiscuous mode packet capture. Given that we wished to work towards an effective visual intrusion detection system we primarily relied upon passive, promiscuous mode packet capture feeling that this greatly reduces the chance of detection.

4. RESULTS

4.1 Experiments

Our experiments were conducted in a networking laboratory and gathered data using the following scenarios: baseline ("normal") traffic, attacks using single tools without extraneous traffic and attacks using single tools with typical traffic. Our intent was to test how well our candidate visualization techniques performed with and without the noise of routine traffic. We plan further experiments that test less aggressive tools and multiple tools in parallel with and without routine traffic. Beyond real-time analysis we wish to include the ability to examine interesting packet capture datasets such as from our university honeynet, the United States Service Academies' Cyber Defense Exercise and the Root-Fu/Capture the Flag events conducted at hacker conventions.

4.2 Attack Tools

The system and visualization suite was tested with a range of popular network attack tools falling into two broad categories: network reconnaissance and vulnerability assessment. The network reconnaissance class of tools typically allow ping sweeps, TCP/UDP port scans and operating systems detection. Many high quality tools of this class are freely available and widely used by attackers. Network assessment tools probe target machines for known vulnerabilities. To test the efficacy of our approach we utilized the tools and host operating systems listed in the following table.

Tool	**Attacker OS**
nmap 3.0	Windows XP, Redhat 8
nmap 3.5	Windows XP
nmapwin 1.3.1	Windows XP
Superscan 3.0	Windows XP
Superscan 4.0	Windows XP
scanline 1.01	Windows XP
nessus 2.0.10	Redhat 8
nikto 1.32	Windows XP
sara 5.0.3	Redhat 8

Figure 10: Attack Tools Tested

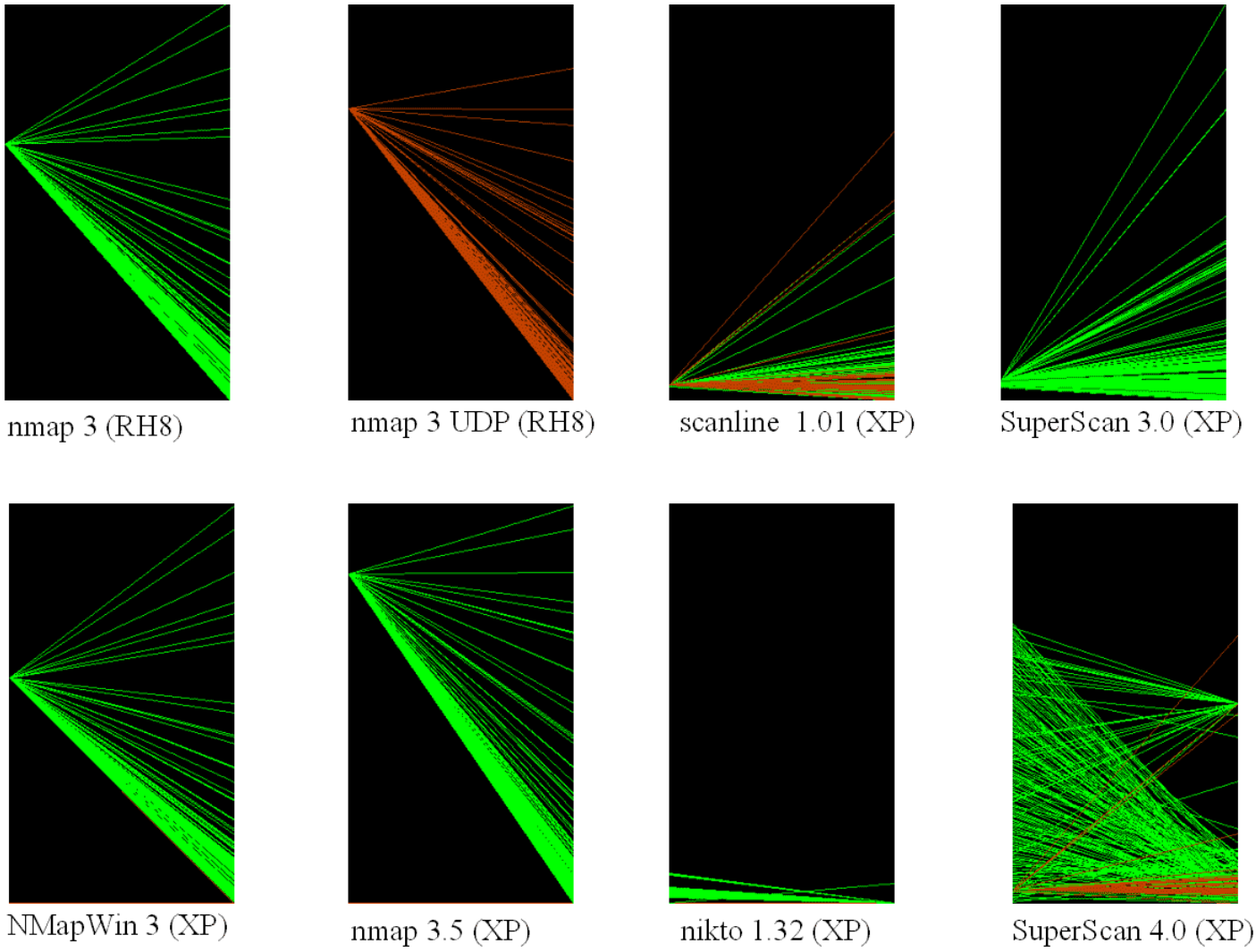

Figure 11: Attack Tools Fingerprints (External Port to Internal Port)

4.3 Analysis

To varying degrees, all the visualizations we tested proved effective in analyzing and fingerprinting attack tools. In particular, the port-to-port parallel plot proved to be of significant value. The images above (figure 11) dramatically show the differences and similarities between several tools run from both Linux and Windows XP operating systems. UDP traffic is in orange and TCP traffic is in green. Each fingerprint can be reliably reproduced with each subsequent use of the tool with only slight variations in the location of the attacker's source ports. The default target ports remain the same. Some may argue that these tools are flexible and alternate ports may be chosen by the attacker. In addition, some tools have publicly available source code and an attacker could create a heavily modified application and thus alter the fingerprint. This is true, but naive use of default settings would indicate that the attacker might be of limited experience. Some tools do not have publicly available source code and offer only a limited set of functions. Another insight is that by knowing the attack tool in use, the network administrator can take appropriate action. For example, if your web server was probed using the *nikto* vulnerability assessment tool (pictured above). The system administrator might wish to do the same in order to be certain that the tool did not report any vulnerabilities. We were surprised to find the striking similarity between *scanline* and *SuperScan 4.0.* If you look closely you can see what appears to be the visual signature of *scanline* embedded in *SuperScan.* While this view provides an excellent overview of network traffic it lacks the ability to zoom and filter as well as provide details on demand. In our future work we will attempt to address these issues.

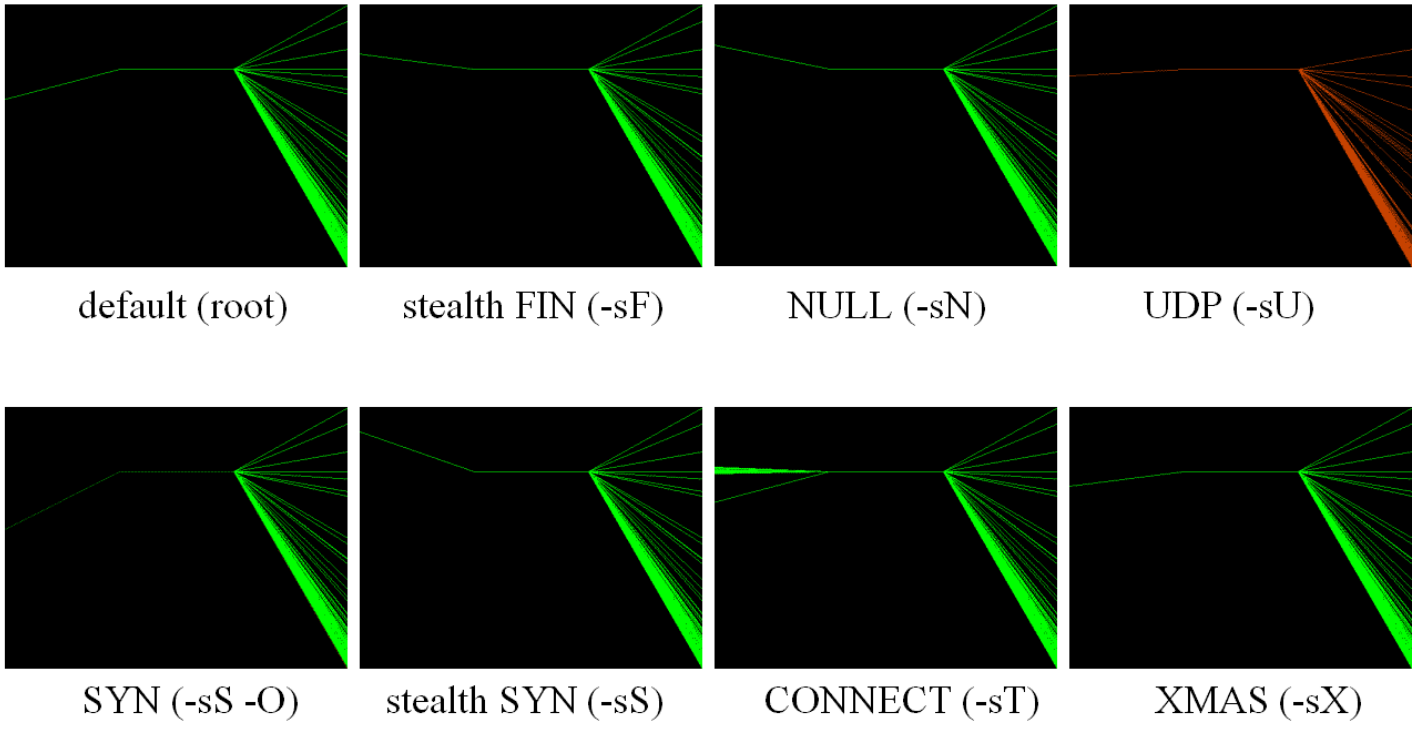

Figure 12: In-Depth Look at Common *nmap* Options
(External Port to External IP to Internal IP to Internal Port)

The port-to-IP-to-IP-to-port views in Figure 12 compare common modes of *nmap* 3.0. The respective mode and command line switch is listed underneath each image. After studying these images we noted several things. This view is useful for normalizing the characteristic *nmap* fan because all attacks against the same IP address show the base of the fan at the same point. Stealthy attacks that take advantage of weaknesses in the TCP protocol, such as the SYN scan, still need to send packets across the network and the signature is still visible. While it is still possible to take advantage of different implementations of the TCP/IP stack to perform evasion or insertion attacks, the visualizations above show that some classes of stealth techniques can be detected. Aspects of the underlying implementation and operating system show through as well. If you consider the range of source ports used by each of the above you will see that there is a difference. *Nmap* typically relies on raw sockets allowing the application to control virtually every aspect of packet construction. The CONNECT scan above shows a wide range of source ports in use that we suspect is due to reliance upon the *connect* system call. The ability to predict operating system source ports was recently proven to be a critical component of TCP reset attacks. A weakness of the above visualization is the inability to detect subtle differences between most of the scans. The FIN, NULL, XMAS, SYN and SYN with operating system fingerprinting all appear the same. In future work we plan to develop visualizations that show the flags in use by TCP packets as we believe that this will show an attacker's operating system fingerprinting attempts.

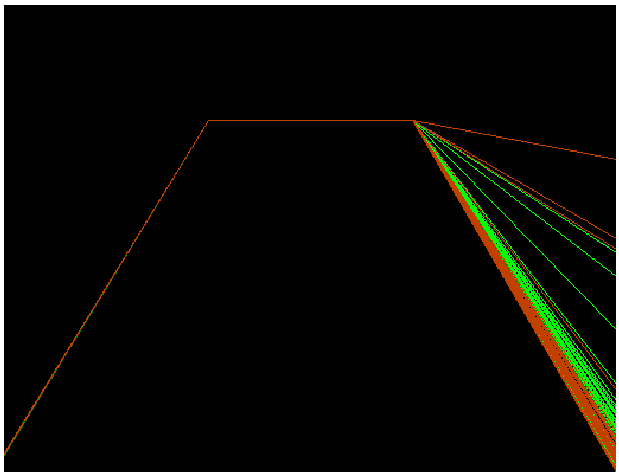

Figure 13: ***scanline 1.01***

(External Port to External IP to Internal IP to Internal Port)

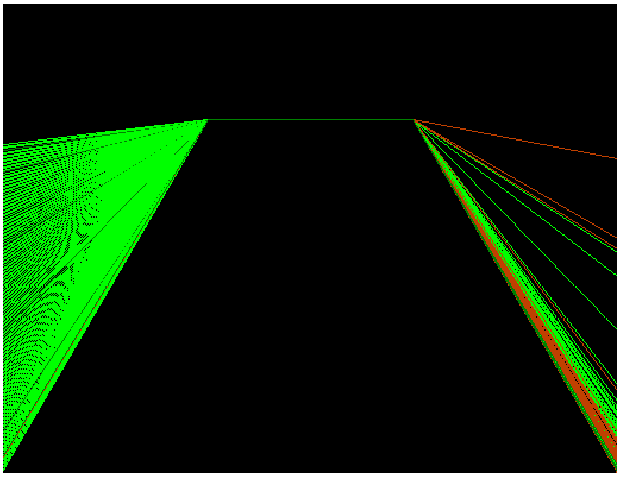

Figure 14: Superscan 4.0

(External Port to External IP to Internal IP to Internal Port)

Reliance upon the same code base may also be evident in some cases. This evidence might prove useful with such tasks as quickly estimating if two malicious software applications were created by the same person or same malware toolkit. Figure 13 shows the *scanline 1.01* tool and figure 14 shows *Superscan 4.0*. Both tools were provided by the same company with *scanline* being made available some time before *Superscan.* The port scanning fans are virtually identical, but the source port fan is dramatically different. We suspect that *Superscan* was developed from the same base source code, but with the addition of multithreading. We are unable to confirm this, as source code for these tools is not available.

While it is possible to see the sequence of ports being scanned it is not a strength of our parallel plot views, but using the scrolling packet length and scrolling port views we were able to gain insight into the time variant nature of the tools.

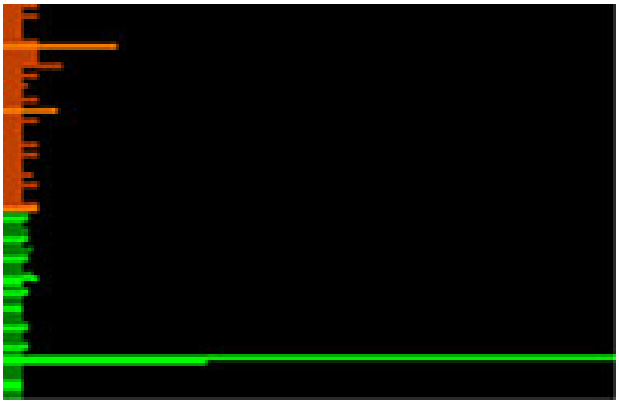

Figure 15: Superscan 4.0 Detail

(Scrolling Packet Length)

We examined each of the tools using the scrolling packet length visualization and found that the technique was useful for determining the relative number of packets generated by each tool and the interleaving of protocol types and responses. We also found distinct visual fingerprints associated with each tool. These fingerprints often proved to be distinct even when interleaved with authorized traffic. Figure 15 shows a portion of a Superscan 4.0 scan. Additional images are omitted due to space constraints.

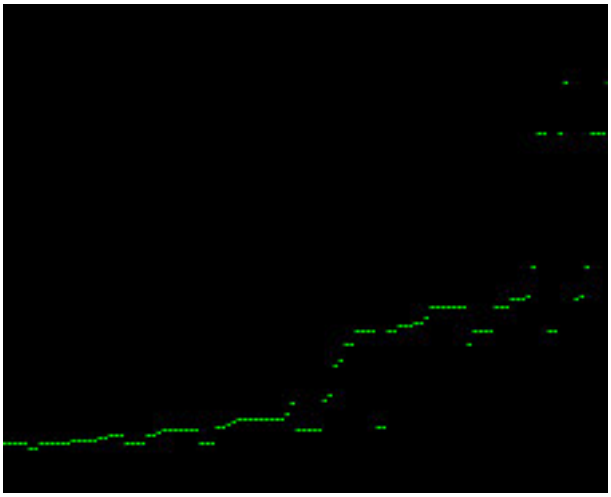

Figure 16: Superscan 3.0 Detail

(Scrolling Internal Port)

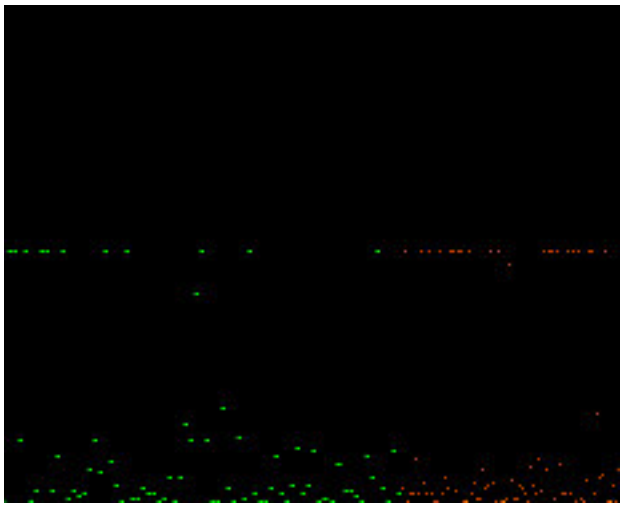

Figure 17: Superscan 4.0 Detail

(Scrolling Internal Port)

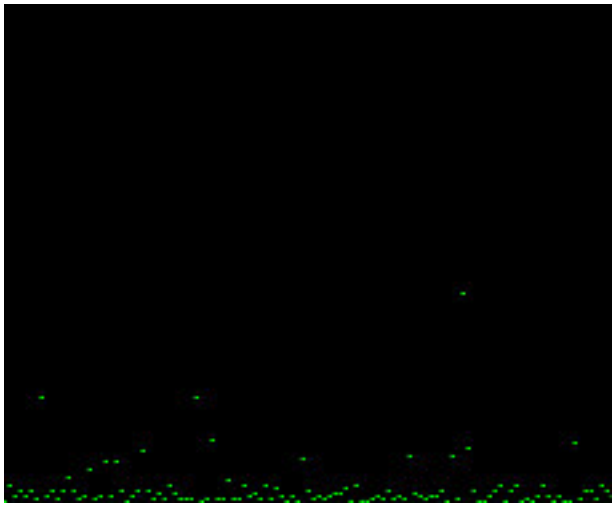

Figure 18: nmapwin 1.3.1 Detail

(Scrolling Internal Port)

The scrolling internal port visualization also provided interesting tool specific insights. Figures 16 shows that Superscan 3.0 utilized sequential port selection. Figures 17 and 18 show that nmap and Superscan 4.0 use a more random approach to target port selection. For future work we plan to examine other techniques that provide better insight into the timing of packet arrival. Even when run from the same machine on an isolated network we noticed a marked difference in the speed with which each tool conducts its scan. Over longer distances across the network this would become noisier with additional network latency, but may still provide valuable insight.

The next series of experiments involved the visualization of distributed attacks and very slow scans against the backdrop of legitimate traffic. Figure 19 (left image) shows the external port to internal port activity from 24 hours of legitimate, attack-free network usage on a small Windows XP network. We tested the system with routine traffic (HTTP, SSH, FTP, SMTP) over the time period and found that a relatively stable picture developed. The external port to internal port view proved to be particularly useful as the traffic was primarily from low client ports to low server ports.

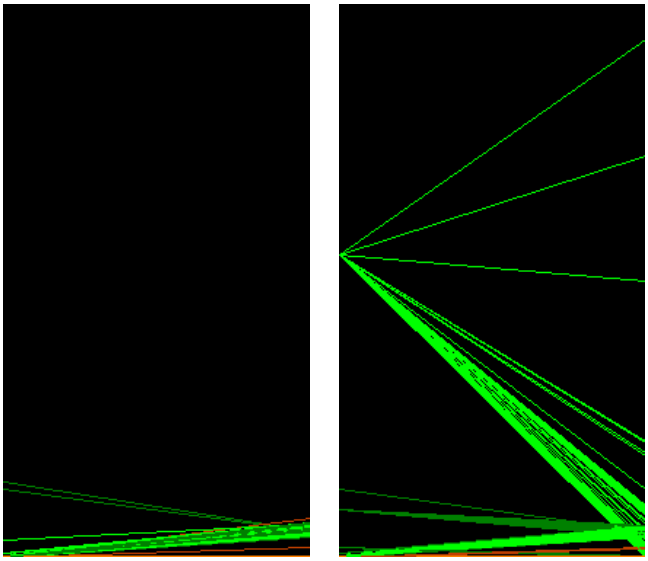

Figure 19: Routine Traffic (left) and Routine Traffic with Slow Scan (right)

(External Port to Internal Port)

Figure 19 (right image) shows a similar period of legitimate network usage, but includes an *nmap* scan. This scan was made using *nmap*'s slowest speed setting "paranoid." The scan clearly shows through the noise of routine traffic with only moderate occlusion. Depending on the nature of routine traffic we believe this visualization technique will be very effective in detecting a variety of slow scans and other slow speed malicious activity. We have conducted additional tests using university and home honeypots and found that it was particularly effective on these systems as virtually all honeypot activity is suspect. Other techniques, such as the external IP to internal IP visualization proved to be less useful for longer term and higher volume activity, although it might prove useful for analysis of distributed denial of service attacks, worm propagation or machine level network

mapping. Over time, IP addresses filled much of the display space. Our conclusion is that the port-to-port visualization will be useful for future research and the IP to IP visualization will be of only moderate value unless it is combined with interactive zoom & filter capability. Our final experiment tested the efficacy of our system to detect distributed scans. The external IP to internal port visualization proved to be most effective. Figure 20 shows scanning activity from several attackers in the presence of legitimate activity.

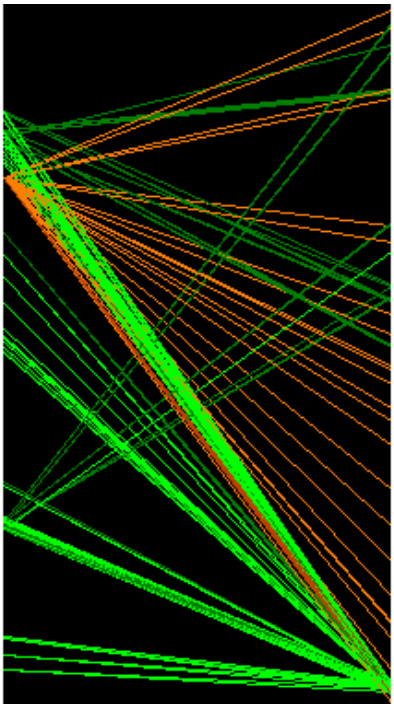

Figure 20: Distributed Scan with Routine Traffic
(External IP to Internal Port)

While we believe this visualization technique is effective against distributed scans, for maximum effect it must be combined with the ability to zoom and filter as well as provide details on demand. These additions will allow an analyst to better isolate the activities of malicious entities.

5. CONCLUSIONS

Our results demonstrated that popular attack tools of the network reconnaissance and vulnerability assessment classes can be readily detected by passive promiscuous mode sniffing and appropriate visualizations. While some occlusion occurs, fingerprints are frequently visible despite the visual noise of routine traffic. This is true even in the case of very slow scans occurring over a range of days to weeks. One important concern is that some tools such as *nmap* are extremely flexible and that more advanced users can construct attacks that will frustrate naive visual analysis. Further research in this area is required to counter these custom applications. In addition to visual fingerprints, the source port allocation used by the attacker is likewise apparent. While not foolproof, the allocation of ports can be an indication of the operating system in use by the attacker. This knowledge can also be used to determine if a network protocol manipulation attack, such as the recent TCP reset attack, is underway. Specific tool fingerprinting and operating system detection can be used to profile attacker activities, skill level and motivations. This intelligence is vital for initiating appropriate responses and for law enforcement investigations. Application specific features such as multi-threading and multiple processes is frequently apparent. This information is useful in determining potential lineage between various attack tools. Lightweight, persistent visualizations are useful against slow and distributed scans and are able to provide insightful overviews of network traffic, but can be fooled by occlusion and visualization specific countermeasures. More subtle attacks (e.g. zero-day attacks) and detailed forensics require more advanced overview and detail functionality combined with greater interactivity such as dynamic querying capability down to the individual packet content level, but doing so will require RAM or disk buffers. These buffers will make such systems more susceptible to resource consumption attacks. Additionally, due to the large datasets involved, effective scaling and labeling techniques must be used. For future work we envision fingerprinting worm behavior and other well-known malicious network activity (e.g. spyware, trojans and warez servers), dual-use activities (e.g. FTP servers, Internet Relay Chat servers) as well as typically legitimate network activities (e.g. email, web browsing). Our long-term goal is to create a library of visual signatures that can be used by the expert or novice analysts to detect malicious activity.

Ultimately, we believe that visual intrusion detection systems can effectively supplement traditional signature and anomaly based intrusion detection

systems, but care must be taken avoid overwhelming the human operator. To this end, the effectiveness of any tool is defined by its usability. Careful task-driven usability studies that optimize ease of use and analyst intuition, will make visual intrusion detection systems more successful and drive down the skill barrier required for effective use.

6. ACKNOWLEDGMENTS

Dr. Wenke Lee, Dr. John Stasko, Dr. Henry Owen, Dr. John Levine, Julian Grizzard, Chris Lee, Byung-Uk Roho and Jinsuk Jun of the Georgia Institute of Technology as well as many members of the Atlanta 2600 chapter provided invaluable insight in support of this research. Dr. Robert Spence and Dr. Edward Tufte's excellent books on information visualization were used throughout this research for their thoughtful surveys of the field and insightful analysis.

7. REFERENCES

1 Teoh, S; Ma, K; Wu, F and Zhao, X. Case Study: Interactive Visualization for Internet Security, *Proceedings of IEEE Information Visualization*, 2002.

2 Teoh, S; Ma, K and Wu, F. A Visual Exploration Process for the Analysis of Internet Routing Data, *Proceedings of IEEE Information Visualization*, 2003.

3 Teoh, S. Graphical Presentation of Stepping-Stone Pairs Found. Initial Results. http://graphics.cs.ucdavis.edu/ ~steoh/ research/tcpdump/tcpdump.html, last accessed April 2004.

4 Security Incident Fusion Tool, National Center for Advanced Secure Systems Research Group. http://www.ncassr.org/projects/sift/papers/, last accessed April 2004.

5 Cheswick, B and Burch, H. The Internet Mapping Project. http://research.lumeta.com/ches/map/, last accessed April 2004.

6 An Atlas of Cyberspaces. http://www.cybergeography.org/atlas/atlas.html, last accessed April 2004.

7 Marchette, D. *Computer Intrusion Detection and Network Monitoring: A Statistical Viewpoint*, Springer, 2001.

8 Erbacher, R and Frincke, D. Visual Behavior Characterization for Intrusion and Misuse Detection. *Proceedings of the SPIE '2001 Conference on Visual Data Exploration and Analysis VIII*, CA, January 2001, pp. 210-218.

9 Code Red Worm Infections. Cooperative Association for Internet Data Analysis (CAIDA) http://www.caida.org/tools/visualization/walrus/examples/codered/.

10 Juslin, J. Intrusion Detection and Visualization Using Perl. O'Reilly Open Source Conference 2001, San Diego, California, U.S.A., 23rd - 29th of July 2001.

11 Zalewski, M. Strange Attractors and TCP/IP Sequence Number Analysis. http://razor.bindview.com/publish/papers/tcpseq.html, last accessed April 2004.

12 Zalewski, M. Strange Attractors and TCP/IP Sequence Number Analysis - One Year Later. http://lcamtuf.coredump.cx/newtcp/, last accessed April 2004.

13 Nyarko, K; Capers, T; Scott, C and Ladeji-Osias, K. Network Intrusion Visualization with NIVA, an Intrusion Detection Visual Analyzer with Haptic Integration. 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. March 24 - 25, 2002. Orlando, Florida, p. 277.

14 Goodall, J. Information Visualization for Intrusion Detection. The Intrusion Detection Tool Kit (IDtk). http://userpages.umbc.edu/~jgood/idtk.php, last accessed April 2004.

15 SecureScope. Secure Decisions. http://www.securedecisions.com/, last accessed April 2004.

16 StealthWatch + Therminator. Lancope Corporation. http://www.stealthwatch.com/, last accessed April 2004.

17 Ethereal: A Network Protocol Analyzer. http://www.ethereal.com/, last accessed April 2004.

18 Etherape: A Graphical Network Monitor. http://etherape.sourceforge.net/, last accessed April 2004.

19 NetStumbler Homepage, http://www.netstumbler.com/, last accessed April 2004.

20 3D Traceroute Homepage, http://www.hlembke.de/prod/3dtraceroute/, last accessed April 2004.

21 The Xtraceroute Homepage. http://www.dtek.chalmers.se/~d3august/xt/, last accessed April 2004.

22 Fydor, "Top 75 Network Security Tools," http://www.insecure.org/tools.html, last accessed March 2004.

23 TCPDUMP Public Repository, http://www.tcpdump.org/, last accessed March 2004.

24 Snort Project Page. http://www.snort.org/, last accessed March 2004.

25 The Honeynet Project. http://project.honeynet.org/, last acccessed April 2004.

26 Ptacek, T and Newsham, T. Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection. Secure Networks, Inc. January, 1998. http://www.insecure.org/stf/secnet_ids/secnet_ids.html, last accessed April 2004.

27 Feature Construction. University of Dortmund. http://kiew.cs.uni-dortmund.de:8001/mlnet/instances/7f000002e50a928a37 last accessed March 2004.

28 Packet Logger Mode, *Online Snort Manual*. http://www.snort.org/docs/snort_manual/node5.html, last accessed April 2004.

29 Lee, Wenke and Stolfo, Sal. "A Framework for Constructing Features and Models for Intrusion Detection Systems," *ACM Transactions on Information and System Security*, Volume 3, Number 4 (November 2000).

30 Brugger, Terry. Data Mining for Network Intrusion Detection. PhD Dissertation draft. Unpublished as of March 2004. University of California - Davis.

31 Zalewski, Michal. Museum of Broken Packets. http://lcamtuf.coredump.cx/mobp/. Last accessed March 2004.

32 Tufte, E. *The Visual Display of Quantitative Information.* Second Edition. Graphics Press, May 2001.

33 Tufte, E. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press, February 1997.

34 Tufte, E. *Envisioning Information.* Graphics Press, May 1990.

35 Spence, R. *Information Visualization.* Pearson Addison Wesley, December 2000.

36 Inselberg, A. Multidimensional Detective. *IEEE Proceedings of Information Visualization '97*, pp. 100-107.

37 Inselberg, A. The Plane with Parallel Coordinates, *The Visual Computer*, 1, pp. 100-107.

38 Wegman, E. Hyperdimensional Data Analysis Using Parallel Coordinates. *Journal of the American Statistical Association,* 85:411, pp. 664-675.

39 Marchette, D. *Computer Intrusion Detection and Network Monitoring: A Statistical Viewpoint.* Springer Verlag, July 2001.

40 Eick, S; Steffen, J and Sumner, E. Seesoft - A Tool for Visualizing Line Oriented Software Statistics. *IEEE Transactions on Software Engineering,* 18, 11, November 1992, pp. 957-968.

The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Military Academy, the Department of the Army, the Department of Defense or the U.S. Government.