

Self-Similar Properties of Network Scanners

Devin Burns

Smithsburg High School

Aaron Nuzman

Aberdeen High School

Dr. John Brand, Mentor

August 15, 2003

Army Research Lab

CISD

ABSTRACT

Network data traffic has been shown to be self-similar.[7] The purpose of this project was to explore self-similar properties found in the traffic produced by network scanners, software used to look for vulnerabilities in a network. Potentially, this information could be used to augment current benchmarking in intrusion detection systems.

INTRODUCTION

Intrusion detection has become an important issue in recent years. As computer networks link computers from all over the world, information security becomes a critical issue. Studies have shown that information theft is up 250%. 99% of all major companies have reported at least one incident of malicious activity, and telecom and computer fraud has totaled over \$10 billion in the US alone. [3]

Network traffic has been demonstrated to be self-similar.[7] Self-similarity is the retaining of a similar behavior or appearance of a data set over space or time. In the case of a network, self-similarity leads to “burstiness” of traffic over a wide range of time scales.

Systems designed to monitor networks for inappropriate, incorrect or anomalous activity are called intrusion detection systems.[6] These systems are not designed to prevent intrusions; instead they are designed to bring malicious activity to someone's attention after a break-in occurs in an effort to abate future attacks. Intrusion detection systems help uncover new ways to deter hackers, persons who make deliberate unauthorized attempts to access information or render a system unreliable or unusable.[5] The detection of a hacker helps pinpoint vulnerabilities in a network and alerts security personal to further security measures to secure the network.

Network scanners are tools that can be utilized by hackers to detect weaknesses in a network. These devices send out series of pings requesting responses from remote hosts. From the responses received the network scanner is able to draw conclusions about the scanned system. Critical data such as the systems open ports and operating system can be derived from the results of a scan. These conclusions assist the hacker in his/her attempt to infiltrate the network.

Network scanners are getting more complex and elusive, making the job of identifying malicious intent more difficult. Identifying characteristics unique to network scanner traffic makes the detection of a scan much more practical. Determining the self-similarity characteristics of scanner traffic may help aid in the detection of network scanners.

Research has been conducted on the effect that an attack has on the self-similarity of overall network traffic.[2] This research paper measures the self-similarity of the intrusion device itself.

SETUP

Computers. Our research began with the construction of four computers. Two of the computers were constructed completely from scratch, whereas the other two computers were in working order, but lacked an operating system. A different operating system was installed on each machine. Windows 98, 2000, XP and Red Hat Linux 8 were each installed on separate machines, although one machine was a dual boot system, running both Windows XP and Red Hat Linux 8.

Network. The next concern was networking the four computers. We obtained a CentreCOM 24 port hub/repeater and four 3Com Network Interface Cards which we used for the construction of our LAN.

Software. The necessary scanner and sniffer software was installed on the dual boot machine and the machine running Windows XP. It was not deemed necessary to install scanner and sniffer software on each computer. Two computers with software would allow for diverse measurements, and assure that the location of the scanner and sniffer had no effect on the resultant traffic. Time was taken to examine different network sniffers. Eventually Ethereal was chosen as the sniffer due to the numerous features it offered and its availability and functionality in the

Windows environment. Ethereal works in conjunction with the pcap library which was also installed for proper functionality. Nmap, Netlab, and SARA, three network scanners all freely available from the Internet, were downloaded and installed. Nmap and Netlab were installed on both computers, but SARA could only be installed on the dual boot machine running Linux, as it is not offered for Windows. Our final software tool installed for statistical analysis was S-Plus. S-Plus was chosen for its availability and ease of use when graphing and calculating best fit lines.

Self-Similarity Programs. Variance.c, rslog, and calculations_14 were compiled for use in determining the self-similarity characteristics of our data. Calculations_14 is a program written by our mentor Dr. John Brand based on an algorithm written by Jan Beran. Variance.c and rslog were programs found on the Internet and are written by Duke P. Hong from the Department of Information and Computer Science at the University of California, Irvine.[4]

PROCEDURE

Scanner and Scan Type. To gather data on each scanner, many test scans were run. Nmap has several different types of scans. The different scans gather different sets of information and some take measures to avoid detection. Eleven scans were tested: Connect, SYN stealth, FIN stealth, UDP scan, Null scan, Xmas tree, IP scan, ACK scan, Window scan, RCP scan, and list scan. Nmap also has several “throttle” settings, which change many factors including how many hosts are scanned at once, the time between packets, and the time Nmap will spend scanning a host.

SARA had fewer options, just listing seven scanning “levels”. These included four basic scans ranging from light to extreme, and three custom levels with special scans. Netlab had only very basic scanning options, letting the user choose the range of IP addresses and which ports are scanned, as well as the host timeout. Netlab was

not examined as thoroughly because it seemed less useful as a scanner.

Capturing and Formatting Data. Ethereal was used to capture the network traffic as each scan was performed. Ethereal gathers data including the types of packets, source and destination, and information in the packets. However, all that was needed to investigate the self-similarity was the level of traffic over time. To get that information from Ethereal, Tethereal was used. Tethereal is the version of Ethereal that works from the DOS prompt, so a list of times and sum of the packets could be converted into a simple text file. This text was converted into two columns, time and number of packets, as was necessary for input into the programs.

Calculation of Hurst Parameter. The formatted capture from each scan had to be analyzed for self-similarity. It was decided that the Hurst parameter would be used to find self-similar behavior. Three programs were tested for ability to find the Hurst parameter accurately, rslog, variance.c and calculations_14. To test the programs, synthetic data with a known Hurst parameter was run through each program. The output was put into S-plus to find the slope of the line of closest fit on a log-log scale. For the program that used the rescaled range, the Hurst parameter is estimated to be equal to the slope. For the programs that used the variance-time algorithm, the Hurst parameter is estimated to be equal to one plus half the slope. Inconsistencies were found when testing variance.c and calculations_14, it was decided that these inconsistencies would hurt the accuracy of our calculations. The rslog program generated fairly consistent results so it was chosen as our model for the calculation of the Hurst parameter.

Once it was decided that rslog was the most accurate, the data generated from this program was formatted for plotting in S-Plus. The data required formatting in

order to use S-Plus's linear regression models. The data was normalized using the norm program found with Duke P. Hong's rslog program. The normalized data was then opened in S-Plus and a linear regression model was used to calculate the slope of the data set. The slope is the estimate of the Hurst parameter and our approximation of the self-similarity of the data.

RESULTS

Data. The level of self-similarity as measured by the Hurst parameter varied, but certain patterns in the data sets were evident. Appendix A gives some sample data sets displaying Hurst parameters for each scan. Shorter scans and scans that did not produce as much traffic generally lacked self-similarity characteristics. As the scans got larger and produced larger traffic streams, Hurst parameters tended to range from 0.6 to 0.9. As evidenced by this data set, network scanners do not produce traffic that has a consistent Hurst parameter. The scanner that is being used to run the scan, the type of scan, and any other special features, such as timeout options, have different effects on the self-similarity of the traffic stream produced. It should also be noted that our network consisted of only four computers, one scanner and three targets. With a larger network and with background traffic scanners may behave differently.

Certain characteristics of the scanners were also observed by viewing patterns in data traffic. For example, it was observed that Nmap scans run in two parts. First, Nmap will first start to check each IP address in the range it is given. Then when a host is found, Nmap goes into the specific scan of that host, and then continues to search the range.

Many scans did not have adequate amounts of data. After running traffic from the scans through the rslog program, several had only five or six points. It was

decided that this was probably not a very accurate indicator, but it is difficult to decide at what point the data could be considered accurate.

The slower settings, with long delays between pings were unable to be fully evaluated because of time constraints, though it was evident from trial scans that the longer length of the scans had effects on the self-similarity of the traffic.

Possible Uses. Self-similarity could be used as an effective second order detection technique. An IDS system is especially effective at detecting bolder scans, where the destination port remains constant for a longer period of time and where the packet count is abnormally high per time interval. On the other hand the IDS weakest attribute is detecting “stealthy” scans. These “stealthy” scans have varying destination ports and longer intervals between pings. An IDS system will have trouble detecting this type of scan because a “stealthy” scan attempts to blend itself in with normal traffic, making it much more difficult for an IDS to detect. The possible implementations of self-similarity as a second order detection technique to the IDS system could be used most effectively to detect longer scans.

A number of reasons support the use of self-similarity in the detection of longer possibly more “stealthy” scans. For instance, viewing network traffic as a whole and not individual traffic streams going to each port eliminates the elusiveness of a scanner switching ports between pings. This technique is used of course when measuring self-similarity over an entire network. Also, the low packet count or number of pings per time interval will increase the length of the scan. Increasing the length of a scan increases the accuracy of a self-similarity measurement. By comparing the normal levels of self-similarity of network traffic to traffic with the presence of a scan it may be possible to discern the presence of malicious activity. This assumes that when scanner traffic is mixed with the average network traffic a

distinct indication of two distinct traffic streams will be evident.[1]

CONCLUSION

While some of our results showed high levels of self-similarity and long range dependence others did not. The large variance in self-similarity values of network scanners can only lead us to conclude that the self-similarity values of a network scans are not consistent. This conclusion is based on a wide variety of different scan types using different scanning tools.

Our conclusion does not suggest however, that self-similarity can not be used effectively as an aid to intrusion detection. Specific patterns in data samples could be used as an effective means of detecting certain network scans. If a scans level of self-similarity differs from that of the networks, it may be possible to detect the presence of the scan. This is especially true in cases of traffic streams with very high self-similarity levels.

Also our conclusion does not take into account the fact that our research was conducted in a lab with no background traffic and only three targets. A larger network of targets will elongate certain scans and may therefore have an effect on the self-similarity of that type of scan. Also, a network with background traffic would likely increase the level of self-similarity, because it would result in the collisions of packets. These collisions would require a standard waiting period before the re-sending of the packet, resulting in increased long range dependence.

A hindrance to the accuracy of our data is the short length of some of the scans. Self-similarity cannot be an easily identified characteristic without a scan of sufficient length. This of course bolsters its use to detect longer scans.

Our results are not conclusive, further study needs to be done to discern the self-similarity characteristics of network scanners and its possible use in IDS.

ACKNOWLEDGEMENTS

We would like to thank our mentor, Dr. John Brand for his devotion in educating us for life and for science. His support and patience were unparalleled and greatly appreciated. Thanks especially to our alternate mentor, Ms. Ann Brodeen, for her guidance and technical knowledge, that were necessary for the completion of our project. We also need to extend our gratitude to Mr. Fred Brundick for his assistance in the evaluation and analysis of the self-similarity programs. We are also grateful to Maria Lopez for teaching us C programming, which we used in formatting the data from Tethereal. We wish to thank everyone who assisted us in the construction of presentation. Finally, just a general thank you to everyone at ARL-CISD for treating us kindly and with the utmost respect.

APPENDIX A

Sample Hurst Parameters

Scanner	Type of Scan	Avg. # of data points generated	Avg. Hurst Parameter
SARA	Normal	12	.6034
SARA	Heavy	15	.8528
SARA	Extreme	17	.9692
SARA	Custom	7	.7758
Nmap	Connect	34	.5461
Nmap	FIN	2	.8547
Nmap	Null	4	.7572
Nmap	ACK	3	.7798
Nmap	UDP	8	.8474
Nmap	IP	5	.8517
Nmap	List	28	.6803
Netlab	Scan	8	.7839

Each sample scan was run a minimum of three times. The resultant Hurst parameters, and points generated were averaged.

REFERENCES

- 1 A. E. M. Brodeen, J. Brand, G. W. Hartwig Jr., F. S. Brundick, M. C. López. “Self-Similarity in Tactical Network Traffic and Tactical Radio Experiments.” ARL MR, to be published.
- 2 Allen, William.
<<http://www.cs.ucf.edu/courses/cop46101L/Notes/IntrusionDetection.ppt>>
- 2 Eugene H. Spafford, Security Seminar Department of Computer Sciences, Purdue University, Jan 1996.
- 4 Hong, Duke P. <<ftp://ftp.ics.uci.edu/pub/duke/self-similar/>>
- 5 J. P Anderson. “Computer Security Threat Monitoring and Surveillance.” Technical report, James P Anderson Co., Fort Washington, Pennsylvania, April 1980.
- 6 Lehmann, Dirk “Intrusion Detection FAQ: What is ID?”
<http://www.sans.org/resources/idfaq/what_is_id.php>, © 2002-2003.
- 7 W. E. Leland, M. S. Taqqu, W. Willinger, and D.V. Wilson. “On the self-similar nature of ethernet traffic (extended version),” *IEEE/ACM Trans. Networking*, vol. 2, no. 1, pp. 1-15, Feb. 1994.

Other References:

Northcutt, Stephen. *Network Intrusion Detection An Analyst's Handbook*. Indianapolis, IN: New Riders Publishing, 1999.

Beran, Jan. *Statistics for Long-Memory Processes*. NY: Chapman & Hall/CRC, 1994.