AN EVALUATION OF SCAN-DETECTION ALGORITHMS IN NETWORK INTRUSION DETECTION SYSTEMS

Richard J Barnett[1], **Barry Irwin**[2]

Rhodes University
Department of Computer Science
South Africa

[1]**barnettrj@acm.org**, [2]**b.irwin@ru.ac.za**

ABSTRACT

Network Intrusion Detection Systems are becoming more prevalent as devices to protect a network. However, the methods they use for some forms of detection are flawed. This paper builds upon existing research by van Riel and Irwin which illustrated these flaws in Snort and Bro's scan-detection engines. Indeed, it has been ascertained that a number of different scanning techniques are not identified by either Snort or Bro.

This paper highlights current research into the improvement of these scan-detection algorithms and presents insight into how this research is being conducted at Rhodes University. This research will improve on the scan-detection engines in Snort and Bro, permitting them to be used in a production environment without fear of succumbing to the false negative problem which currently exists.

KEY WORDS

Network Security, Intrusion Detection, Port scanning, Snort, Bro

AN EVALUATION OF SCAN-DETECTION ALGORITHMS IN NETWORK INTRUSION DETECTION SYSTEMS

1 INTRODUCTION

This paper describes current research being performed by the Security and Networks Research Group, in the Department of Computer Science at Rhodes University. It expands on research already performed in the Department in previous years, and in particular picks up on problems highlighted by van Riel and Irwin in [5].

Network Intrusion Detection Systems (NIDS) are more and more frequently becoming valuable aids to network administrators in the constant battle against attacks on current network centric computing. Indeed, whilst firewalls are now standard in the design of networks, network administrators need to know if and when an attack breaches that first line of defence. NIDS alert the administrator to any abnormal happenings inside a network. Most current NIDS rely on signature based detection on traffic flows through the network. The Open Source NIDS Snort [2] and Bro [1] use this method. [7]

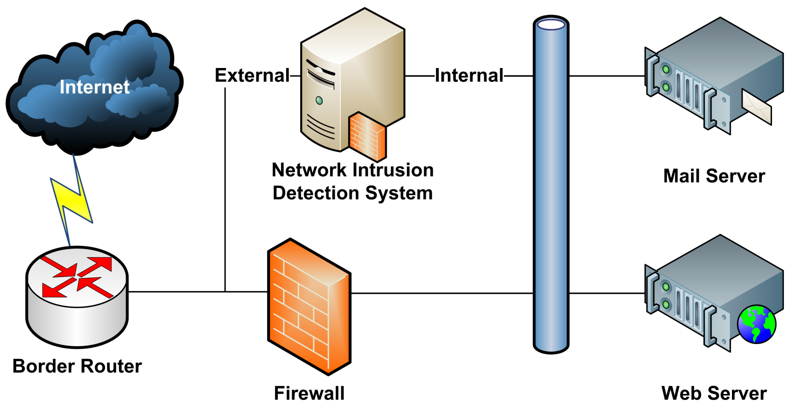
Figure 1 shows a typical placement of a NIDS inside a network. A NIDS would usually sit on the inside of the firewall and would sit on a span port of the local switch. This would give it the ability to view all traffic destined for all hosts on the network. In this case, it is illustrated by a web and mail server. The NIDS would be configured to apply rules targeting web and mail traffic.

Current NIDS can suffer from two major problems. False positives (which occur when a NIDS alerts on traffic which is benign) and false negatives (when a NIDS does not alert but an intrusion has occurred) are significant problems which can render a NIDS useless, either by wasting administrators time or by lulling them into a false sense of security.

Port scanning is a frequently used tool for identifying specific vulnerabilities in networked hosts, and is usually a precursor to further intrusion attempts. It is reported that the current (very large) volume of network attacks and specifically scanning activity may be “the tip of a very large iceberg” [14]. Despite the fact that most current NIDS make use of signature detection, both Snort and Bro have the additional capability to perform scan-detection.

van Riel and Irwin [5] (in the Department of Computer Science) have identified a number of flaws with the algorithms in both Snort and Bro. This was

Figure 1: A NIDS Inside a Network

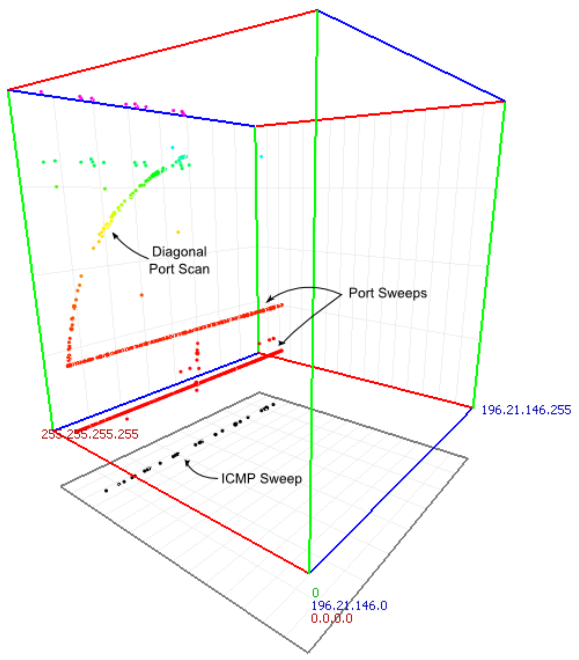


done by developing [13] and using a graphical tool developed for the analysis of network traffic, specifically traffic from network telescope captures. The tool, *InetVis*, presents traffic in a 3D space with a a time delay animation. These flaws present a variety of challenges and question the usefulness of the Snort and Bro algorithms. Figure 2 shows three different possible scans identified using *InetVis*. A horizontal black line with is an incomplete ICMP sweep, two red lines with represent port sweeps, and a curved multicoloured line which represents a diagonal scan. This permits scans to be rapidly identified in the telescope capture. These scans can then be isolated from the capture file with *InetVis* and then be replayed to Snort and Bro. This permitted van Riel and Irwin to identify the flaws in Snort and Bro with some efficiency.

New research is underway to investigate the possibility of extending the Snort and Bro algorithms to work in an efficient and effective manner. This research aims to increase the usefulness in deploying a NIDS and monitoring scans.

The remainder of this paper is structured as follows: Section 2 provides a fairly comprehensive look at some related work in the field of network intrusion detection and scan-detection. Section 3 will discuss how the authors plan on investigating improvements to the algorithms in Snort and Bro. Section 4 will investigate the outcomes that this research hopes to produce, and finally Section 5 will present the conclusions that can be drawn from the research performed so far.

Figure 2: Scanning Techniques Identified in InetVis



2 RELATED WORK

There is a wide variety of literature available in network intrusion detection and port scanning. This section looks at a selection of existing research which may be beneficial for the authors attempts at improving scan-detection in existing NIDS. This section is broken down into work related to port scanning (which will be presented in Section 2.1), work relating to statistical analysis of packets (in Section 2.2) and finally work related to anomaly detection in Section 2.3.

2.1 Port Scanning and Scan Detection

Port scanning can be identified by classifying both connection attempts and the hosts which make such attempts. Allman *et al.* [3] propose a method of classifying connections into three categories. “Good” connections, which are those which are successful. “Bad” connections, those that do not lead to established connections, and “unknown” connections, which are those which cannot be identified for any reason. Hosts can be classified into either “good” hosts, or “bad” hosts. Allman *et al.* observe that there could be numerous methods of performing such categorisation, but that a simple and effective method is to classify hosts which make a majority of "good" connections as "good" and hosts which make a majority of "bad" connections as being "bad". From this, it is possible to classify scanning activity as traffic which is "bad" and originates from a "bad" host.

Through the use of *InetVis*, van Riel and Irwin [5] have identified a number of different scan types which include diagonal scanning, step scanning and what they refer to as a "creepy crawly" scan. Some of the scans types that have been identified are slow and long running, and exhibit timing and destination address and port selection intended to avoid observation.

2.2 Statistical Analysis in NIDS

A number of authors have investigated the possibility of introducing statistical analysis into NIDS to improve their reliability, whilst decreasing their overhead. Crotti *et al.* [4] propose the use of statistical fingerprinting to quickly identify the contents of a given packet. Their research suggests the use of *Probability Density Functions* to perform the fingerprinting. They do, however, note that one of the primary problems with such an approach is that the contents of packets are required to train the system and that most publicly available packet traces remove all useful application level information for security reasons.

A similar method is proposed by Karamcheti *et al.* [6], who propose the use of *inverse distributions* to classify packets. This method relies on separating each packet into a number of sub-strings and performs comparisons on these sub-strings against known traffic samples. The relationship between the two can then be fitted to the *inverse distribution* to determine the nature of the packet. As with Crotti *et al.*, this solution suffers from the need to have full packet traces to perform seeding.

However, this use of full statistical fingerprinting can, itself, be a limiting factor for NIDS. Ramaswamy *et al.* [11] propose an alternative method. Their method uses *approximate* fingerprinting which makes use of a sliding window across the packet contents to fingerprint the contents. This method may produce false positives, but will never produce false negatives. Therefore, matching packets can then be analysed in more detail making use of more traditional methods.

These methods all require the use of normalised traffic, Rubin *et al.* [12] propose a very sophisticated algorithmic approach which they call *protomatching* to make a single pass over unnormalised network traffic. Because of this, this method can perform well in high throughput network environments.

2.3 Anomaly Detection

Despite the usefulness of each of the approaches discussed above, they all focus quite extensively on signature based NIDS, and look at the contents of a given packet in relation to a rule. Kompella *et al.* [8] and Krügal *et al.* [9] both discuss alternative methods for performing anomaly detection.

Firstly, Krügal *et al.* [9] discuss a method of performing service specific anomaly detection making use of a two part system including a packet processing unit and a statistical processing unit. The packet processing unit performs stream reassembly and other normalisation tasks, and could easily be a NIDS such as Snort or Bro. The statistical engine performs the anomaly detection and can be a separate application, or could be integrated into a NIDS as a plug-in. The statistical analysis proposed by Krügal *et al.* includes analysis on the type of request, the length of the request and the payload distribution of the packet.

The alternative proposed by Kompella *et al.* [8] suggest the use of an "intelligent" data structure which is called a *Partial Completion Filter (PCF).* This is suggested as a possible method to perform scan-detection (amongst other uses) and is designed to be scalable as it uses a largely fixed amount of memory, as it stores only a count of seen packets which match against a hash function. By having several hash functions and counters and a trigger condition on each counter, it is possible to build a sophisticated *PCF* which scales and targets a number of possible scan areas.

There are two traditional approaches to performing scan-detection [8]. The first being that a number of events during a given interval are counted, the second that the number of failed connections in an interval are counted. This corresponds well with work done by Levchenko *et al.* [10] who make a similar assertion. Further to that, however, Levchenko *et al.* discuss the assumption that port scanning requires per-flow state to be stored, which does (as has been seen) not scale effectively. Their paper proves mathematically that ingress detection of port scanning cannot be performed without maintaining state. They do, however, prove that egress detection of scanning activity does not rely on the presence of state tables.

In this context, Levchenko *et al.* define ingress detection as that which looks at incoming connections, and egress detection as that which identifies the TCP RST packets which are transmitted when a connection to a closed port is attempted. (Or the ICMP "Port Unreachable" packet which is sent for UDP connection attempts.) This method suffers from the possibility that host-layer firewalls may prevent such packets from being transmitted.

3 RESEARCH APPROACHES

Initially, the authors intend on verifying the effectiveness of Snort and Bro's scan-detection algorithms in alerting on a variety of scanning techniques. Using the graphical tool *InetVis* [13], the current flaws in the Snort and Bro algorithms will be verified. The results of this process will be used in conjunction with detailed packet analysis and statistical processing, to develop techniques for detecting such scans.

The most common scans, port scans and sweeps are well defined and in the simple case are easy to identify. This is also the case with the related techniques ICMP scanning and other host reachability scanning. Pseudo-random phenomena, which could be attributed to backscatter and the results of network configuration errors, can also be the result of significantly more complex scanning techniques which are more difficult to identify.

A number of a constraints involved in the processing of network traffic are anticipated. These constraints involve the physical system resources required in looking for components of scans, specifically the memory required to detect long, slow running scans and the processing power required to scan large volumes of traffic. Pseudo-random scans are particularly difficult to isolate with limited resources, as a lot of memory is required within a non-trivial temporal reference frame. Experiments on the performance of each of the developed methods when provided with large volumes of data - as would be the case on a high throughput link - will be investigated. This experimentation will permit the authors to determine which scan-detection techniques will scale and which will not.

This will permit algorithms to be developed which solve the false negative problem and which perform in a scalable manner. These methods are expected to involve statistical analysis, multidimensional matrix operations and projections. The integration of these methods and techniques into Snort and Bro is the ultimate outcome of this research as discussed in the next Section.

4 RESEARCH OUTCOMES

This research plans to take validated results from prior research at Rhodes University to identify problems with current scan-detection algorithms in use in common NIDS. Having identified and validated the problems in existing algorithms, the authors aim to adapt and improve these algorithms.

These algorithms will be used to develop new plug-ins for both Snort and

Bro. Whilst the authors intend on producing plug-ins for just two NIDS, the result could be easily modified to produce plug-ins for any system.

As the architectures of Snort and Bro are quite different, the process which will be taken to build plug-ins for them will also be somewhat different. The Snort plug-in is best suited to being developed as a C++ shared object which will act as a Snort Preprocessor. This can be integrated into the Snort pipeline in a suitable place to allow for maximum performance [7]. Bro has a much more flexible customisation system. The use of the Bro language will be considered and used if possible. If a more complex solution is required a Bro *analyser* is an alternative which would be developed in C++ [1].

In addition to the Snort and Bro plug-ins, the authors intend on producing a hardware accelerated version of the plug-ins which can be deployed as an alternative in environments where high volume networking prevents the CPU from efficiently scanning all traffic.

5 CONCLUSIONS

This paper has illustrated current research into the improvement of scan-detection at Rhodes University. The research is still in its infancy, but is anticipated to improve the state of scan-detection engines in NIDS when complete. The authors intend on developing new techniques for scan-detection based on the existing Snort and Bro plug-ins, and a number of other methods. A variety of methods are being investigated and are likely to include a significant statistical analysis component. Existing research suggests this is a positive approach.

ACKNOWLEDGEMENT

The authors would like to acknowledge the support of their colleagues in the Department of Computer Science. We would also like to acknowledge the support of Telkom SA, Business Connexion, Comverse SA, Stortech, Tellabs, Amatole, Mars Technologies, openVOICE and THRIP through the Telkom Centre of Excellence in the Department of Computer Science at Rhodes University. Finally, the authors would also like to acknowledge the support of the National Research Foundation.

REFERENCES

[1] Bro intrusion detection system - bro overview. Online: http://www.bro-ids.org/, Accessed: 28/01/2008.

[2] Snort - the de facto standard for intrusion detection/prevention. Online: http://www.snort.org/, Accessed: 28/01/2008.

[3] Allman, M., Paxson, V., and Terrell, J. A brief history of scanning. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (New York, NY, USA, 2007), ACM, pp. 77–82.

[4] Crotti, M., Dusi, M., Gringoli, F., and Salgarelli, L. Traffic classification through simple statistical fingerprinting. *SIGCOMM Comput. Commun. Rev. 37*, 1 (2007), 5–16.

[5] Irwin, B., and van Riel, J.-P. Inetvis: a graphical aid for the detection and visualisation of network scans. In *Conference on Vizualisation Security (VizSec2007)* (2007).

[6] Karamcheti, V., Geiger, D., Kedem, Z., and Muthukrishnan, S. Detecting malicious network traffic using inverse distributions of packet contents. In *MineNet '05: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data* (New York, NY, USA, 2005), ACM, pp. 165–170.

[7] Kohlenberg, T., Alder, R., Dr. Everett F.Carter, J., Foster, J. C., Jonkman, M., Marty, R., and Poor, M. *Snort Intrusion Detection and Prevention Toolkit.* Syngress Publishing Inc., 2007.

[8] Kompella, R. R., Singh, S., and Varghese, G. On scalable attack detection in the network. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement* (New York, NY, USA, 2004), ACM, pp. 187–200.

[9] Krügel, C., Toth, T., and Kirda, E. Service specific anomaly detection for network intrusion detection. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing* (New York, NY, USA, 2002), ACM, pp. 201–208.

[10] Levchenko, K., Paturi, R., and Varghese, G. On the difficulty of scalably detecting network attacks. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security* (New York, NY, USA, 2004), ACM, pp. 12–20.

[11] Ramaswamy, R., Kencl, L., and Iannaccone, G. Approximate fingerprinting to accelerate pattern matching. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement* (New York, NY, USA, 2006), ACM, pp. 301–306.

[12] Rubin, S., Jha, S., and Miller, B. P. Protomatching network traffic for high throughputnetwork intrusion detection. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security* (New York, NY, USA, 2006), ACM, pp. 47–58.

[13] van Riel, J.-P., and Irwin, B. Inetvis, a visual tool for network telescope traffic analysis. In *Afrigaph '06: Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa* (New York, NY, USA, 2006), ACM, pp. 85–89.

[14] Yegneswaran, V., Barford, P., and Ullrich, J. Internet intrusions: global characteristics and prevalence. In *SIGMETRICS '03: Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems* (New York, NY, USA, 2003), ACM, pp. 138–147.