Detecting Scans at the ISP Level

Carrie Gates, Software Engineering Institute

Josh McNutt, Software Engineering Institute

Joseph B. Kadane, Department of Statistics,
Carnegie Mellon University

Marc Kellner, Software Engineering Institute

April 2006

TECHNICAL REPORT
CMU/SEI-2006-TR-005
ESC-TR-2006-005

Carnegie Mellon
Software Engineering Institute

Pittsburgh, PA 15213-3890

Detecting Scans at the ISP Level

CMU/SEI-2006-TR-005
ESC-TR-2006-005

Carrie Gates
Josh McNutt
Joseph B. Kadane
Marc Kellner

April 2006

Networked Systems Survivability Program

Unlimited distribution subject to the copyright.

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

Copyright 2006 Carnegie Mellon University.

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Internal use. Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use. Requests for permission to reproduce this document or prepare derivative works of this document for external and commercial use should be addressed to the SEI Licensing Agent.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

List of Tables

Abstract

Scans are often used by adversaries to determine the potential weaknesses in a target network or system prior to an intrusion attempt. In other cases, exploits are packaged with the scans themselves. This report presents a novel approach to detecting scans (including very stealthy scans) against, or passing through, very large networks. It meets operational requirements that are particular to detecting scans in ISP level networks.

This scan-detection approach performs an ongoing, incremental analysis of flow-level data regarding traffic inbound to a network. It is multi-dimensional and flexible, based on up to 21 characteristics describing traffic collected from any single source. The report describes in detail a method developed to provide a probability that a particular traffic sample contains a scan. In validation testing using a manual analysis of traffic collected from a high-volume network, this method correctly classified 99.3% of TCP traffic samples.

This report also compares this new approach to other scan approaches, particularly a naïve scan approach, based on simple thresholding, and a modified version of the threshold random walk approach, to which it performed comparably. Combining the random walk approach with the new approach produced very good results, reducing the number of false negatives to zero.

1 Introduction

Scans are a concern in cyber-security because they either (1) indicate a potential future threat or (2) represent an immediate threat of attack. With regard to the first case, scans have traditionally been used by adversaries to map a target network and identify potential vulnerabilities before attempting an intrusion. Military literature documents a direct correlation between the success of an attack and the thoroughness of the preceding reconnaissance [McCarthy 94]. There is every reason to believe that this is also true in the digital realm, wherein an intrusion attempt is more likely to be successful if the adversary is familiar with the target network and its services. By detecting that an adversary is conducting reconnaissance—and understanding this may be a precursor to an attack—a network administrator is provided the opportunity to determine the specific weaknesses likely to be exploited, and to be especially vigilant, knowing that an attack may be pending. For example, Panjwani and colleagues have discovered that approximately 50% of attacks are preceded by some form of scanning, most often vulnerability scanning, where open ports are probed at the application level to determine what version of what service is running on a particular port [Panjwani 05]. Regarding the second case where scans represent an immediate threat of attack, intrusion attempts have been programmed to immediately follow scan packets, and in other instances the attack packets themselves completely constitute the scan. Worms in their propagation phase are excellent examples of these behaviors, but this is also seen in more focused attack tools that have been deployed by "script kiddies" [Honeynet 02]. Scan information can potentially be used by a network administrator to determine whether any host has been potentially infected or compromised.

However, many commonly deployed port scan detection methods consist of a simple thresholding approach applied in real-time—determining whether the source IP address attempted to contact more than some minimum number of destinations within a particular time frame. This approach[1] suffers from being too general, and is therefore prone to a large number of classification errors. As a result, many scans may be missed and/or so many erroneous alerts may be generated that they are largely ignored by network administrators. In addition, attackers have responded to common detection methods by developing more stealthy techniques for scanning. It is widely believed that very stealthy scans suggest more sophisticated adversaries who probably pose a greater threat to the target. As a consequence, detection of stealthy as well as non-stealthy scans is an important need in the cyber-security community.

[1] Other detection approaches have been reported in the literature, and are summarized in Section 3.2.

This report presents a novel approach to detecting scans against, or passing through, very large networks. Unlike previous methods, this approach is based on a combination of expert opinion and statistical analysis of network traffic. Experts developed a list of characteristics that can contribute to determining whether a set of flow records contains a scan. A logistic regression model using these characteristics as independent variables, and developed using a Bayesian process, was generated based on network traffic. This model is designed to detect "noisy scans," such as those that are readily detected by other approaches, as well as many stealthy scans that currently avoid detection by other means. Rather than provide a binary (yes/no) decision on whether the event contains a scan, we provide a probability that the selected traffic contains a scan, providing the user with some measure of confidence in the decision. This new approach performs an ongoing, incremental analysis of flow-level data about traffic inbound to a network. It therefore does not rely on having traffic records for both directions (which are not always available on large networks due to practices such as asymmetric routing), nor does it require examining individual packets. In addition, it does not require any knowledge of the internal architecture, or server and workstation deployment, of the network being monitored.

An introduction to port scanning is provided in Section 2. Section 3 provides our motivation for performing this work, including our design considerations and how other approaches compare to these requirements. Our approach to detecting scans is described in detail in Section 4. In particular we outline our design considerations, the indicators of scanning activity that we recognized in aggregated flow records, and a description of the modeling approach used including how it was both developed and tested. In Section 5 we directly compare our algorithm to a modified version of the threshold random walk and a naïve version of flow-level scan detection [Jung 04]. We then describe some initial results in Section 6, including a discussion on how a database of scanning activity might be used, and present our conclusions in Section 7.

2 Background

Most definitions of scanning focus on the purpose of reconnaissance. However, as noted at the outset of this report, scanning has evolved to include intrusion attempts programmed to follow scan packets immediately, and instances where the attack packets themselves completely constitute the scan. In this work, we have extended the traditional concept of scanning to include these newer cases. Therefore, our definition of scans is a reconnaissance activity that is aimed at multiple targets. This definition includes activities such as worms, which may not be performing a scan that is targeting the monitored network explicitly.[2]

Several methods, described below, have evolved to prevent port scans from being detected [de Vivo 99, fyodor 97, hybrid 99, Staniford 02].

- Abuse the TCP/IP protocol (e.g., SYN scans, FIN scans, and undefined flag combinations such as in SYN-FIN, Xmas tree, and NULL scans).
- Randomize the order in which IP addresses and/or ports are scanned.
- Insert a time delay between scans and/or randomize the length of the time delay between scans.
- Randomize TCP packet field values (e.g., sequence numbers, acknowledgment numbers, source ports).
- Hide the real scan within scans from spoofed IP addresses.
- Distribute port scans across multiple sources.

By abusing the TCP/IP protocol, adversaries can avoid having their activities logged at the application level, and can also avoid being blocked by some firewalls (e.g., those which might not allow a SYN packet through, but will not confirm that a FIN packet is associated with an established session). By randomizing the order in which IP addresses are scanned, as well as the TCP packet field values, early intrusion detection systems (IDSs) would not log the activity because the detection of scans would be based on assumptions about these values. Additionally, many IDSs are configured to detect a scan only if some minimum number of unique targets is contacted within a particular time frame. By inserting time delays between targets, adversaries can avoid detection by these IDSs. Some scanning software, such as nmap also provides the ability to hide the true scanning IP within a number of spoofed source IP addresses [nmap 06]. In addition to obfuscating which source is the actual scanner, some

[2] We do not claim that this approach is suitable for the real-time detection of worms, but rather that this method will detect worms as performing scans, in addition to detecting the more traditional reconnaissance form of scanning.

IDSs will log only the first five sources; when the actual source is the sixth source none of its activities are logged at all [Solar designer 98]. Finally, distributing scans across multiple sources can make the scan from each individual source be small enough not to be logged. At the very least, it obfuscates the amount of information about the targets actually gathered by the adversary.

3 Motivation

The development of a new approach to scan detection has been motivated by a desire to obtain comprehensive, integrated information about the multitude of scans directed against, or passing through, very large networks. This includes very stealthy as well as less stealthy scans. The approach is intended for organizations managing very large intranets or portions of the Internet backbone, as well as for smaller Internet Service Providers and intranets. Network traffic will be monitored at the border routers of such a network. Today's intrusion detection systems cannot handle the rate and volume of data necessary for successful deployment at the borders of a very large network—leading to the need for this research [Kruegel 02].

3.1 Design Considerations

The operating conditions for our scan detection system can be summarized as follows.

- A very large network generally includes multiple border routers, quite possibly geographically distributed, and may also employ asymmetric routing policies.
- Traffic is being routed for multiple administrative domains, which are relatively, if not completely, independent. The overall network managers have relatively little control over, or knowledge of, these administrative domains.
- The managers of very large networks generally do not desire to take real-time actions in response to scans (e.g., automatically blocking them), as these actions could cause a denial of service if an adversary were to spoof legitimate IP addresses as scanning sources, for example. Given that there are multiple (independent) administrative domains, it is difficult, and potentially impossible, to maintain a white list of IP addresses that should never be blocked at the router.
- Any traffic data to be transmitted or stored must be substantially smaller than the original network traffic.

Based on our stated motivation, and these operating conditions, we have developed a scan detection approach with the following characteristics:

- It operates on flow-level traffic data, not on packet-level data. Briefly, flows are aggregations of packets sharing the same protocol, source address and port, destination address and port, and arriving reasonably close together in time. Cisco's NetFlow which is unidirectional, and Argus, which records bidirectional flow information, are probably the best known systems for monitoring flows [Cisco 06, QoSient 04]. Flow-level traffic data does not include payload, and is far less voluminous than packet-level data (even packet summaries without payload). Some potential users of our system will not want to record payloads due to privacy considerations. Even packet-level traffic summaries are too voluminous to manage for large networks.

- It does not need to operate in real-time. Thus our approach is to analyze traffic data in an ongoing, incremental manner. This is primarily due to two reasons: (1) flow-level data is not collected in real time, as the communication session needs to terminate or time-out before the flow is flushed from the cache, and (2) operators of very large networks generally do not plan to take real-time actions in response to scans, such as automatically blocking the traffic. ISPs do not do this and intranet managers often do not want to trust intrusion detection that much. The actions they do commonly take include sending complaint letters to the source ISPs, blocking offending source addresses, and blocking certain destination ports when they have little legitimate use. However, these actions are only taken after human review and approval—not in real time. Additionally, traffic that has been identified as containing scans is often filtered out so that analysts can focus on the non-scan traffic when performing security analysis.
- It does not depend on bi-directional traffic data; we need to analyze only traffic inbound to the network. Bi-directional data is more voluminous. More importantly, with multiple routers and asymmetric routing we might not be able to observe the replies. Performing later matching of two sets of unidirectional flow data is computationally intensive.
- It does not depend on awareness of network characteristics, such as IP addresses in actual use, or topography. This information is not likely to be available with independent administrative domains and/or dynamic address assignments.

We have chosen to focus initially on single-source (versus distributed) scans. Our approach is designed to detect all of the types of stealthy scans identified in Section 2, except for distributed scans.

3.2 Related Work

Scan detection itself is not a new concept, of course. For example, Snort has a preprocessor that extracts port scans, which is based on either invalid flag combinations (e.g., NULL scans, Xmas tree scans, SYN-FIN scans) or on exceeding a threshold. By default, Snort is configured to generate an alarm only if it has detected SYN packets sent to at least four different ports in less than three seconds [Roesch 99].

Bro, written by Vern Paxson also detects scans using thresholding [Paxson 98]. In this system, network scans are detected by a single source contacting more than some threshold of destination IP addresses. Similarly, vertical scans are indicated by a single source contacting too many different ports (it appears that this is independent of the destination IP address). Paxson notes that false positives are also generated by this method, such as by a single-source client contacting multiple internal web servers, or by an FTP server being located on a non-standard port. Bro also uses payload information, as well as packet information, and so provides analysis for specific applications to further reduce false positives.

Staniford and colleagues have developed a method for detecting port scans, including stealthy and distributed port scans, based on a simulated annealing approach [Staniford 02]. First, packets are pre-processed using a method called Spade, which flags packets as either normal or anomalous. All packets flagged as anomalous are passed to Spice, where they are placed in a graph, with connections formed based on a simulated annealing approach, so that packets that are most similar to each other are grouped together. This approach detects port scans, as well as other events such as denial-of-service attacks and server and router misconfigurations [Paxson 01].

More recently, Robertson and colleagues have developed a method based on return traffic [Robertson 03]. They reconstruct sessions (generating approximate sessions, rather than dedicating the CPU cycles necessary to generate complete reconstructions) and flag any source IP that has contacted a destination that has not responded as performing a port scan. A score is assigned to each source IP based on the number of destinations contacted for which there was no response. A second method, called PSD (for peering center surveillance detection), has additional heuristics for analyzing traffic where there is the possibility that some response traffic information is not available (and hence, no response does not necessarily indicate a scan).

Jung and colleagues have also developed a method based on return traffic (or on the use of an oracle that knows if a particular host and service are available) [Jung 04a]. They use sequential hypothesis testing to determine whether a source is performing a scan as new connection requests arrive. If the intended destination does not respond with a SYN-ACK, it is more likely that the source is performing a scan. Conversely, if the destination returns a SYN-ACK, it is more likely that the source is benign. Once sufficient evidence supporting one or the other hypothesis is obtained, a decision is made.

Visual methods for detecting port scans have also been developed. For example, Conti and Abdullah demonstrated how parallel coordinate plots can be used to detect different scanning tools, using packet-level information [Conti 04]. Lakkaraju and colleagues presented a visualization system called NVisionIP and demonstrated how horizontal scans can easily be detected [Lakkaraju 04]. Rather than using packet information, their system is based solely on flow information. Finally, both Muelder and colleagues and McPherson and colleagues describe PortVis, which produces visualizations based on summarized flow information [Muelder 05, McPherson 04].

The methods described by Roesch and by Paxson require having packet level information and cannot be modified to use flow information, which is one of our requirements [Roesch 99, Paxson 98]. While the methods described by Robertson and by Jung use packet information, the approaches can likely be applied with reasonable success to flow information instead [Robertson 03, Jung 04]. Indeed, we demonstrate in Section 5.1 how the threshold random walk approach can be successfully modified for flow information.

However, these approaches require either that the flow information in both directions be (largely) available or that knowledge of the network architecture is available. Recall that this information is generally not available under the operating conditions which we have specified in Section 3.1.

While the above methods depend on either payload information, or having information available for both directions of a session, two papers have discussed recognizing port scans using only unidirectional flow information. Fullmer and Romig have developed a tool called flow-dscan, which examines flows for floods and port scans [Fullmer 00]. Floods are identified by excessive packets per flow. Port scans are identified by a source IP address contacting more than a certain threshold of destination IP addresses, or destination ports (where only ports numbered lower than 1024 are examined) on a single IP address. Fullmer and Romig make use of a "suppress list" consisting of IP addresses, such as multi-user games and web-based ad servers, that often appear as port scans. Similarly, port scans are identified in by Navarro and colleagues by extracting the source IP addresses that have contacted more than a certain threshold number of targets within a particular time frame [Navarro 00]. In the example used in the paper, the threshold was 64 destination IP addresses within three days. It should be noted that, while these two flow-based approaches do not perform their analyses in real time, they still meet our design requirements.

In all of the approaches described here, except the visualization approaches and that of Staniford and colleagues, at most two thresholds have been used to determine whether some network activity was a scan [Staniford 02]. In the cases where two thresholds were used, this was done to distinguish between a network and a host scan, or to determine whether an item was definitely a scan or definitely not a scan. Additionally, all of these approaches have made a binary decision—classifying the event as a scan or as not a scan. To our knowledge, the approach we present in this report is the first approach to scan detection that provides a continuum representing the probability that the traffic contains a scan.

Spade / Spice and the visualization methods are the only exception to the above observations [Staniford 02]. The visualization approaches, which are not threshold based and have been developed for a range of data granularities, do not automatically detect scans, but rather demonstrate how a user can recognize scans based on the patterns they exhibit. In contrast, our approach provides an automatic classification of traffic as containing scans. Spade / Spice does not use a threshold either, but rather clusters data together, allowing the user to view the clusters and determine whether they represent a scan or some other activity. Thus, Staniford and colleagues do not provide a binary classification of the cluster as being a scan.

4 Approach

We have developed a scan detection approach that meets the needs and design considerations discussed in Section 3.1. In addition, this approach produces a probability that a collection of network traffic contains a scan; this is in contrast to the binary (yes/no) results provided by most other scan detection methods and tools. Moreover, our approach was developed based on an extensible collection of scan indicators, using a statistical analysis to determine the contribution of each indicator and whether any of those indicators could be removed from the model. This is in contrast to the typically one or two metrics used by most other scan detection methods and tools.

The approach processes all flow records collected for a user-specified time interval (e.g., 10 minutes, an hour, a day). On our client site, we collect and process flows using the SiLK[3] toolset [CERTCC 02, Gates 04]. For historical analysis, this approach can be run on any past period of time for which flows are available. However, in routine production mode, it is run in an ongoing, incremental fashion—processing each successive time period as the flow data becomes available. Although it does not perform true real-time analysis, the user can choose to operate on successive short time intervals of a few minutes, making it closer to real time and less historical. Recall that real-time analysis is not needed for the real-world operating conditions for which we have designed this approach (see Section 3.1.). For example, our pilot installation has chosen a one-hour interval.

The first step in this approach is to divide the traffic into discrete events. Here an event is defined as a set of traffic flows, from a single source IP address, occurring during a period of activity that is bracketed on both sides by a period of no activity. The minimum quiescent period (inter-event gap) can be specified by the user (e.g., one minute, five minutes). In addition, the beginning and end of the time period for which data are being processed are treated as qualifying quiescent periods. This guarantees that there will always be at least one event for each observed source address, and that every flow from a source address will be part of some event.

In the heart of our approach, each event is analyzed to assess the likelihood that it contains scanning activity. Only events exceeding a user-specified minimum size (e.g., 32 flow records) are actually analyzed in this stage. However, all smaller events are segregated for processing at a later stage; they are not lost. The details of event analysis are presented below.

Our ongoing, incremental processing also includes provisions for handling activity that extends across time intervals. If the time gap between the last flow in the preceding period and the first flow in the current period (for this source IP address) is less than the minimum

[3] For more information, see http://silktools.sourceforge.net.

quiescent period for an event boundary, we treat this as an extended event and include the previous flows in the event analysis in the current period. This allows periodic, updated reporting of long-lived scans showing the entirety of each since its start. It also captures the beginning of a scan, even if the activity took several time periods to accumulate more than the minimum number of flows selected for processing.

Finally, this approach has been designed to allow multiple passes over the data set. This allows a first pass to process data from relatively short periods of time (e.g., minutes, hours), analyzing larger events and then removing them from subsequent analysis. The remaining flow records can then be combined over longer time periods, and processed again. This process can be repeated multiple times, using longer time periods, using longer inter-event gaps, and accepting smaller event sizes. Our approach can thereby detect progressively smaller and lower intensity scans.

4.1 Scan Indicators

Once the incoming traffic for a discrete time interval has been subdivided into events, each event of sufficient size is individually analyzed to determine whether it contains a port scan. Conceptually, a matrix is generated using the traffic flows for the event, with the destination IP address on the *x*-axis and the destination port on the *y*-axis (see Figure 1). Thus, the destination (the pair of values for destination IP address and destination port) of each flow in the event identifies a specific cell in the matrix. Key information about each flow is recorded in the appropriate cell of this (conceptual) matrix, including packet count, byte count, source port, TCP flags, start time, and duration. (Recall that all flows composing a single event came from the same source IP address.)

The geometry of the filled cells in the matrix can then be analyzed for patterns representing horizontal, vertical, or strobe port scans [Staniford 00]. A horizontal scan will appear as a horizontal line in the matrix, while a vertical scan will appear as a vertical line in the matrix. A strobe scan (a scan of multiple ports across multiple hosts) will appear as multiple horizontal lines.

Domain experts identified 21 variables as potentially indicating that a scan is present or specifically not present, based on a combination of the footprint geometry and other event characteristics (such as some indication that the event consists of completed connections or that the majority of the flag combinations do not indicate the presence of backscatter[4]). These variables are later used to assess the probability that an event contains a scan. The particular

[4] *Backscatter* is the response to a service request initiated by a source that has spoofed its IP address. The response is therefore observed at the network where the spoofed IP address resides. This phenomenon has been used to determine the number of denial-of-service victims [Moore 01].

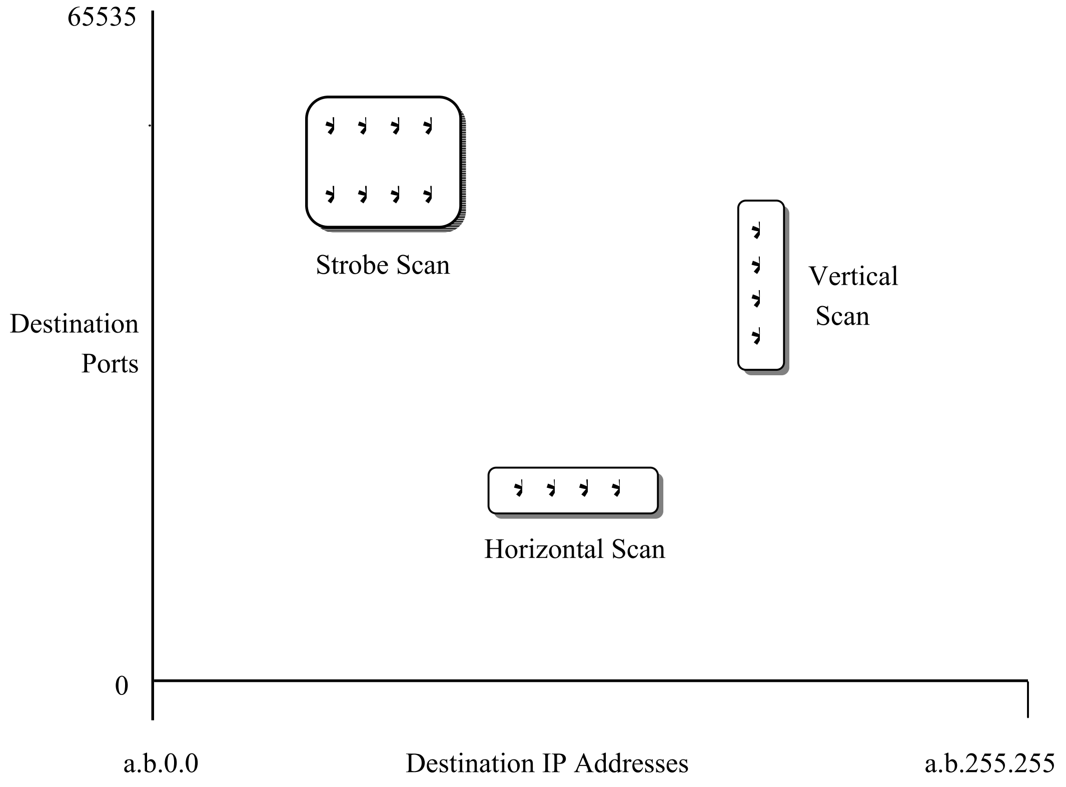

Figure 1: Geographic Shape of Scans in Conceptual Matrix of Addresses and Ports

indicators were devised to reflect characteristics of network traffic that would help to distinguish between scanning and other types of traffic. No single indicator provides a definitive classification of traffic, but in combination these indicators strongly distinguish scans from other traffic. We use a statistical analysis to determine which of these indicators contributes the most to determining if a scan is present or not. The 21 initial scan indicators used in the model are below:

1. maximum /24 subnet run length
2. ratio of flows that do not have the ACK bit set to all flows
3. ratio of flows to known malware ports to all flows
4. ratio of flows with fewer than 3 packets to all flows
5. maximum run length of IP addresses in any one /24 subnet
6. maximum number of IP addresses contacted in any one /24 subnet
7. maximum number of high destination ports contacted on any one host
8. maximum number of low destination ports contacted on any one host

9. maximum number of consecutive high destination ports contacted on any one host
10. maximum number of consecutive low destination ports contacted on any one host
11. number of unique destination IP addresses
12. number of unique source ports
13. average number of source ports per destination IP address
14. ratio of flows with “standard” flag combinations (SYN and ACK set, along with either the FIN or RST bit set) to all flows
15. ratio of the number of flows with the average bytes/packet > 60 to all flows
16. median value of packets per destination IP address
17. ratio of flows with “standard” combination (standard flag combination and at least three packets and at least 60 bytes/packet on average) to all flows
18. ratio of flows with backscatter combination (RST, RST-ACK, or SYN-ACK for the flag combination and the average number of bytes/packet is ≤ 60 and the number of packets per flow is ≤ 2) to all flows
19. ratio of unique destination IP addresses to number of flows
20. ratio of unique source ports to number of flows
21. ratio of flows with backscatter flag combinations (SA—RA—R) to all flows

The minimum number of flows that we require for analysis is 32. This figure was arbitrarily chosen based on the need for a large enough set of data to recognize that a scan has occurred, but a small enough amount of data to catch smaller scans. In the future we would like to address the minimum number of flows required in order to reach a decision on whether or not a set of unidirectional flows represents scanning activity. However, 32 flows have been shown to be sufficient for detecting scanning activity (where the scan consists of at least 32 flows—smaller scans will not be detected).

4.2 Logistic Regression

As mentioned above, each event is individually analyzed to determine the probability that it contains scanning activity. We use a model based on the statistical technique of logistic regression to accomplish this. This particular technique is applicable when the outcome (dependent) variable is dichotomous (i.e., Boolean). Such is the case here, as “This event does or does not contain scanning activity.” In general, a logistic regression model estimates the probability that the outcome variable will be true in any given case, based on the value(s) of one or more predictor (independent or explanatory) variables for that case.

Mathematically,

$$\hat{P}(\text{outcome is true}) = e^y / (1 + e^{y)}$$

where

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n$$

The β values are the parameters of the model and must be estimated based upon a dataset of numerous cases, containing the values of the explanatory variables and indication of whether the outcome variable was true or false for each case. Many statistical software packages will perform the calculations needed to estimate the β values from the dataset provided. Once the model has been calibrated in this way, the equations above can be applied to predict the probability for any other case, based on the values of the explanatory variables for that case.

An important feature of the logistic regression model is that it has a graceful relationship to recoding the event of interest. Thus

$$\begin{aligned} P(\text{outcome is false}) &= 1 - P(\text{outcome is true}) \\ &= 1 - e^y / (1 + e^y) \\ &= 1 / (1 + e^y) \\ &= e^{-y} / (1 + e^{-y}). \end{aligned}$$

Thus studying the event wherein a particular set of records does *not* contain a scan would result in values of β with the same absolute value, but with the sign reversed.

4.3 Model Development

To estimate the β values for the logistic regression model, we must provide a dataset—with the predictor (independent) variables as well as the outcome (dependent) variable—to a statistical software package. The outcome variable is the experts' determination of whether or not the event contains scanning activity. We use our 21 scan indicators as the explanatory variables.

Classical logistic regression approaches use a labeled training set, developing a model based on this training data. This requires that the training data be representative of all possible cases. In practice, such a complete training set is difficult to obtain. We therefore use instead a Bayesian approach to logistic regression, which uses domain experts to determine prior probabilities for each of the variable co-efficients. Once the prior probabilities have been established, the model is developed based on the training data, but with adjustments made to the co-efficients affected by the prior probabilities. This can be loosely thought of as weighting the co-efficients. The resulting model is based on a combination of expert opinion and training data. In addition to mitigating the effects of having a training set that might not be representative of all cases, this also requires less training data.

In order to perform a Bayesian logistic regression, two data sets were required: an elicitation set and a training set. Events were generated based on flow data collected on May 4, 2005. For the elicitation set we used 129,191 TCP events gathered from 17:00-18:00 GMT. For the training set we used 130,062 TCP events gathered during the subsequent hour. The values for each of the 21 independent variables were calculated for each of the events in the two data sets.

One of the assumptions that comes from using a logistic regression is that the relationship between the dependent and independent variable is linear in the logit scale. This assumption implies that the values for the independent variables that will therefore provide the most information are those located at the extremes [Elfving 52]. The ultimate goal is to reduce the variance in the estimated dependent variable $\hat{y}$. The variance of $\hat{y}$ is proportional to the variance of each of the co-efficients. Therefore reducing the variance of $\hat{y}$ requires reducing the variance of each of the co-efficients, β_i, for each of the independent variables. Given that there is an inverse relationship between variance and the sum of squares, we can minimize the variance by maximizing the sum of squares. The sum of squares is maximized by choosing values that are furthest from the mean, and so we choose the most extreme values for each of the 21 variables when generating the model [Weisberg 85, p. 13-15].

We chose the 100 most extreme observations in *X*-space, where the *X* matrix contained 21 columns, one for each variable. In order to ensure that the *X* matrix was invertible, we needed to have at least 21 rows, or observations. These observations were chosen randomly from the entire data set, and used to seed the matrix. Each subsequent observation was chosen using a maximum variance criterion. That is, we selected the next observation to be that observation for which we had the highest remaining uncertainty (highest variance) about our independent variable. The variance of an observation, v, is proportional to $v^T (X^T X)^{-1} v$, where v^T is the transpose of the vector *v*, X^T is the transpose of matrix *X,* and $(X^T X)^{-1}$ is the inverse of the matrix $X^T X$. Note: *X* includes all observations that have been included so far, and each observation, *v*, we consider is chosen from observations not yet included in *X*. For each observation that has not yet been added to the *X* matrix, we calculate its variance. The observation having greatest variance is added to *X*. This process is repeated, using the new *X* matrix and finding the observation with the greatest variance relative to the new *X*, until *X* is considered sufficiently large.

We choose to use 100 observations for the elicitation process. Thus the *X* matrix required 100 rows, where the first 30 were randomly chosen to seed the matrix and the remaining 70 were chosen based on their having the largest variance. The values for each of the 21 variables for these 100 observations were provided to an analyst who used this information to estimate the probability that the event contained a scan (*y*). The analyst was not allowed to use any other information in estimating the probability. The results were used to calculate the priors for the logistic regression model. A prior is calculated by transforming the probability y_i to a value on the logit scale, $log(y_i)$. A linear regression is then used to determine the prior values for each of the co-efficients.

The training set was chosen using the same procedure as described above; however, the *X* matrix in this case consisted of 200 observations. For each of the 200 observations, the analyst was presented with the original flow data, rather than the values for each of the 21 variables. Additionally, the expert could use any other information in order to determine whether the event contained scanning activity. Also contrary to the elicitation process, the expert provided only a binary indication when a scan was present in the event data, rather than a probability. Of the 200 observations, the expert flagged 53 as containing scanning activity.

The Bayesian logistic regression model updated the prior distribution obtained during the elicitation process using the training set resulting from expert analysis. The resulting posterior distribution for the coefficients is proportional to the product of the sampling distribution, or likelihood, and the prior distribution for the model coefficients [Gelman 95, p. 65-88]. We drew samples from the posterior using Markov Chain Monte Carlo simulation [Robert 04]. The coefficients for our final logistic regression model are taken to be the mean values calculated from the MCMC samples. The posterior coefficients have been informed by both expert opinion (via the priors) and the sample training data.

The resulting logistic regression model was

$$\hat{P}(\text{event contains a scan}) = e^y / (1 + e^y)$$

where

$$y = -2.4891 + 0.0032x_1 + 1.9576x_2 - 4.0846x_3 - 0.1572x_4 + 0.0073x_5 + 0.0112x_6 + 0.0002x_7 + 0.4595x_8 + 0.0013x_9 - 0.3714x_{10} + 0.0000x_{11} - 0.0001x_{12} - 0.0002x_{13} - 1.0647x_{14} - 0.3448x_{15} - 0.0005x_{16} - 0.8837x_{17} - 3.5294x_{18} + 1.8988x_{19} + 0.1308x_{20} + 0.1018x_{21}$$

where each variable has been enumerated in Section 4.1.

However having 21 variables can lead to excessive processing time, particularly in the cases where not all of the variables contribute importantly to the calculation of the probability that an event contains a scan. We therefore used the Akaike Information Criterion (AIC) in order to remove any variable whose removal did not importantly affect the model's fit to the data [Akaike 73]. The result was that only six variables are required by the model:

$$\hat{P}(\text{event contains a scan}) = e^y / (1 + e^y)$$

where

$$\hat{y} = -2.83835 + 3.30902x_2 - 0.15705x_4 - 0.00232x_{13} - 1.04741x_{15} + 3.16302x_{19} - 3.26027x_{21}$$

where x_2 is the ratio of flows with no ACK bit set to all flows, x_4 is the ratio of flows with fewer than three packets to all flows, x_{13} is the average number of source ports per destination IP address, x_{15} is the ratio of the number of flows that have an average of 60 bytes/packet or greater to all flows, x_{19} is the ratio of the number of unique destination IP addresses to the total number of flows, and x_{21} is the ratio of the number of flows where the flag combination indicates backscatter (e.g., RST, RST-ACK or SYN-ACK) to all flows.

When the predicted probability that the event contains scanning activity is >= 0.5, one would normally classify that event as a probable scan, where 0.5 is chosen as a threshold based on convention. Using this value of 0.5 as the threshold for a scan, the model is able to correctly classify 197 of the 200 cases in the training data.

One might wonder about interpretation of these β values because regression analysis is so frequently used to gain insight into explanatory and causal relationships between the outcome (dependent) variable and the independent variables. In such applications the researcher must take care to test that the model's parameter (β) values are non-zero at a statistically significant level; those parameters that fail this test are often removed from the final model, as model simplicity is a valued characteristic. We used the Akaike Information Criterion in order to remove those variables that were not statistically significant. We can interpret the remaining co-efficients to determine how the variables affect the prediction that an event contains scanning activity.

A negative co-efficient indicates that as the value for the variable increases, the probability that the event contains a scan decreases. Our model has four negative co-efficients, for variables x_4 (the ratio of flows with fewer than three packets to all flows), x_{13} (the average number of source ports per destination IP address), x_{15} (the ratio of the number of flows that have an average of 60 bytes/packet or greater to all flows) and x_{21} (the ratio of the number of flows where the flag combination indicates backscatter to all flows). The negative co-efficient for x_{13} indicates that the greater the number of source ports per destination IP address, the lower the probability that the event contains a scan. This is because the greater this value, the more connections there were to the same destination host. Since the majority of scans consist of horizontal or strobe scans, their presence is marked by very few connections on average to the same destination host. Note the optimization for detecting horizontal and strobe scans; these are the most commonly found scans in the data, and thus the model adjusts for these scans. The negative co-efficient for x_{15} indicates that the more flows that exhibit payload (indicated by flows with >= 60 bytes/packet), the lower the probability that the event contains scanning activity. This is intuitively apparent because the majority of scanning activity does not consist of connections where data has been exchanged. The negative co-efficient for x_{21} serves to reduce the probability that an event contains scanning activity if the evidence is high that there is backscatter activity, based on the flag combinations of the observed flows.

The negative co-efficient for x_4 is more interesting. As the ratio of flows that are small to all flows increases, the probability that an event contains a scan decreases. This does not make immediate intuitive sense. However, it might be explained by the observation that many scans consist of three SYN packets to each destination (based on standard TCP retries), whereas backscatter, for example, contains only a single packet per flow. Thus the probability that an event contains a scan is reduced the more small flows present in the event. It should be noted that scans with small flows can still be detected based on the values for the other co-efficients.

Only two variables had positive co-efficients: x_2 (the ratio of flows with no ACK bit set to all flows) and x_{19} (the ratio of the number of unique destination IP addresses to the total number of flows). For the first variable this result is to be expected given that the majority of scans do not include a large number of completed connections where data is exchanged. Additionally, scans where the ACK bit has been set are not common. Thus the larger the ratio of flows with no ACK bit, the greater the probability that the event contains a scan. Similarly with the second variable, if there is a large number of unique destination IP addresses compared to the number of flows, it indicates that there were very few flows to each IP address. This behavior would also be observed during horizontal scans, when there is likely only one flow per destination IP address, versus during legitimate activity where there are likely to be several connections to the same host.

It should be noted that the relative values for each of the co-efficients cannot be used to determine the relative importance of each of the variables. This is the case with our model because we did not adjust all the variables to be on the same scale.

One potential weakness of the model stems from reducing it to the most statistically significant indicators. This may have made the model less robust to detection of scan types not well represented in the development and validation samples, such as vertical scans. We also believe that retaining the large number of indicators makes this detection approach far more difficult to defeat than typical approaches using only one or two indicators—even if an adversary is familiar with how the detection approach works. Thus a user more concerned with complete detection rather than processing time might choose to use the full model consisting of 21 variables rather than the reduced model with only six variables.

4.4 Validation

Although our model performed well on the development sample, there is always a risk that such a model has “over-fit” the sample. Therefore it is best to validate the model’s performance on a second sample. Because the validation sample will be used to predict the performance of the model on the full population of events, this sample should be drawn purely at random from that population. Accordingly, 300 events were selected at random from a full dataset of 127,873 events collected on May 4, 2005, from 19:00-20:00 GMT from the same network that was used to gather data for the elicitation and training sets.

The raw NetFlow data for each of these 300 sample events were examined and classified by the same domain expert that analyzed the elicitation and training sets. The six scan indicator values were then calculated for each of these events. The model described in Section 4.3 of this report was applied to compute the probability that each event contained scanning activity based on these six variables. As before, the customary probability of 0.5 was used as the threshold for a scan. Finally, the classification of each event produced by the model was compared with the experts’ classification, as the validation test.

There were 22 events containing scans, and 278 events that did not contain scans, according to the analysis by the domain expert. Our model correctly classified 298 (99.3%) of the 300 events in the validation sample. Of the two incorrect classifications, there was one false negative and one false positive. We use the conditional probability definitions provided by Axelsson where the detection rate is the conditional probability that an alert was generated given there was a scan [Axelsson 00]. Similarly a false positive is the conditional probability that an alert was generated when a scan was not present. Our detection rate was 95.5% (21/22) and our false positive rate was 0.4% (1/278). The sensitivity (true positive rate) is 0.955 while the specificity (true negative rate) is 0.996.[5]

Robertson and colleagues also used domain experts to validate the results of their scan detection system [Robertson 03]. They reported a 0.6% false positive rate—quite comparable to our results. Because they had domain experts review only cases that their system had already identified as scans, their validation approach could not provide any insight into their false negative rate. In both their work and our work, the authors served as the primary domain experts. It should be noted that this does not provide assurance that other experts, unconnected to the work being validated, would classify the same events as scans.

We next turn to the false negative and false positive. The false negative had an estimated probability of being a scan of 41.04%; however, the threshold for classifying an event as containing a scan was set at 0.5 and so this event was classified as not containing a scan. It should be noted that this event had the highest probability of any of the non-scans in the validation data set. This event consisted of 70 flows to seven unique destination IP addresses, all of which were one-to-three packet SYN-only flows.

The false positive consisted of 62 flows to three destination IP addresses. Only two of the flows exhibited characteristics of completed connections to two of the addresses. The remaining 60 flows were all to the third IP address and each consisted of four packets with a SYN-RST flag combination. This event had a probability of being a scan of 60.74%—the lowest probability of any of the scans in the testing set.

4.5 UDP and ICMP Models

In addition to creating a model for detecting TCP scans, we also developed models for User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP) scans using exactly the same procedure. However, not all of the 21 indicators could be used for each of these data sets, as they used information that was specific to TCP data (flag combinations, for example). Thus for UDP we used 15 variables, omitting the 6 variables that included flag information. For ICMP we used only five variables, omitting the variables that included port information as well as flag information from the list of TCP variables. Thus we had four

[5] It is worth noting that ROC (receiver operating characteristics) curves can be used to understand the tradeoffs between false positives and false negatives and determine an appropriate cutoff [Swets 00, Agresti 02, p. 229-230]. In our case, with such a small number of misclassifications, such an analysis is not particularly valuable.

variables that were the same as for TCP scan detection, plus an additional variable that was specific to ICMP: the ratio of ICMP echo requests to all ICMP flows.

The result for UDP traffic, after elicitation and training, was the following logistic regression model:

$$\hat{y} = -3.6090 + 0.0017x_1 + 0.2785x_4 + 0.0087x_5 + 0.0147x_6 + 0.0011x_7 + 0.4190x_8 - 0.0064x_9 - 0.0913x_{10} + 0.0000x_{11} + 0.0000x_{12} - 0.0007x_{13} - 0.0815x_{15} - 0.0011\,x_{16} + 1.8879x_{19} - 0.0977x_{20}$$

where x_1 is the maximum /24 subnet run length, x_4 is the ratio of flows with fewer than three packets to all flows, x_5 is the maximum run length of IP addresses per /24 subnet, x_6 is the maximum number of IP addresses contacted in any one /24 subnet, x_7 is the maximum number of high destination ports (1024 or higher) contacted on any one host, x_8 is the maximum number of unique low-numbered (less than 1024) destination ports contacted on any one host, x_9 is the maximum number of consecutive high-numbered destination ports contacted on any one host, x_{10} is the maximum number of consecutive low-numbered destination ports contacted on any one host, x_{11} is the number of unique destination IP addresses, x_{12} is the number of unique source ports, x_{13} is the average number of unique source ports per destination IP address, x_{15} is the ratio of flows with $\geq$ 60 bytes/packet to all flows, x_{16} is the median number of packets per destination IP address, x_{19} is the ratio of unique destination IP addresses to flows, and x_{20} is the ratio of unique source ports (both low and high) to the number of flows. Using the AIC to reduce the number of variables in the model resulted in the following:

$$\hat{y} = -1.88791 + 0.54368x_4 + 0.02515x_5 + 0.52909x_8 - 1.24418x_{10} - 0.00184x_{13} - 0.22455x_{15} - 0.69794x_{20}$$

The testing set for UDP consisted of 300 randomly chosen events, which contained only three scans. This model correctly identified 299 of the 300 events, where it missed detecting one of the scans. This particular scan was a vertical scan of a single host, where a single 48-byte packet was sent to ports 7273 to 6282, inclusive.

The result for ICMP traffic was the following model:

$$\hat{y} = -4.307079 - 0.08245704x_1 - 0.02800612x_5 + 0.04877852x_6 - 0.000006398878x_{11} + 4.016751x_{22}$$

where x_1 is the maximum number of consecutive /24 subnets that were contacted, x_5 is the maximum run length of IP addresses per /24 subnet, x_6 is the maximum number of IP addresses contacted in any one /24 subnet, x_{11} is the total number of IP addresses contacted, and x_{22} is the ratio of ICMP echo requests to all ICMP flows. We do not use the AIC to further reduce this model because it is already so small, using only five variables. The testing set consisted of 300 randomly chosen events, which contained only a single scan. This model correctly identified all 300 events.

5 Direct Comparison with Other Detectors

As discussed above in Section 3.2, there are several other approaches to scan detection that have been developed. One scan detection approach has emerged as a gold standard in scan detection—the threshold random walk method proposed by Jung and colleagues [Jung 04a]. This approach has been used in, for example, high-speed worm detection [Jung 04b,Weaver 04]. We modified this approach to work on flow-level data, using an oracle to determine whether a host exists, rather than the return network traffic. Our modifications are outlined in more detail below.

5.1 Threshold Random Walk

The threshold random walk (TRW) approach was designed for packet-level inspection, and to be placed either at a border router where both incoming and outgoing data are available, or for use with an oracle that knows what hosts and services are available on the internal network. Given our operating limitations, we modified TRW to operate on flow data and provided it with an oracle.

For a given hour of data, we generated an oracle that indicated the hosts known to exist during that hour. We did this by assuming the source address for any flows recorded on the outgoing interfaces for our network indicated that this particular source address contained an active host. We know that this approach is not perfect because there are, for example, known backdoors on the network that have not been instrumented. However, this oracle provides us with a first approximation for known hosts.

We further modified the TRW algorithm to take into account that our oracle does not have perfect knowledge. If, for a given flow, the destination IP address exists in the oracle, then we consider this as evidence that the source might be benign. However, if the destination IP address does not exist in the oracle, we consider this only as evidence that a host might be scanning if the flow does not have the ACK bit set, indicating that no communication was established between the source and destination. Otherwise, we regard the destination host as existing, and so there is evidence that the source is benign. Thus we err on the side of flagging a source as benign rather than as a scanner.

5.2 Sampling Strategy

The sampling strategy used to compare the capabilities of the TRW approach with our approach and with a naïve algorithm[6] was based on the results obtained from first running

[6] We have also developed a naïve approach to scan detection using unidirectional flow data. In this case we flag as a scan any SYN-only flow records generated by a source with 100 or more flows.

each of the algorithms. We chose all TCP data collected on January 15, 2006, 15:00-16:00 GMT, from a large network. Out of 631,870 total source addresses (regardless of the number of flows associated with each source address), the number of scanners and non-scanners identified by each of the three algorithms is presented in Table 1.

Table 1: Number of Scanners Detected by Each Algorithm

	Naïve	**Our approach**	**TRW**
Scans	9,118	3,184	7,932
Non-scans	622,752	628,686	623,938

Our sampling strategy focused on determining the capabilities of the algorithms given complex examples, as opposed simply to performing random sampling and comparing true and false positive rates. In order to determine appropriate test cases, eight different sets of data were generated, using the results from each of the three algorithms. Each algorithm splits data into two groups: scanners and non-scanners. The data sets that were generated consist of all possible combinations of these groups. The size of each of the eight groups is presented in Table 2. For this comparison, we only used those sources that generated at least 32 flows, in order to ensure that we did a fair comparison between our approach and the other algorithms.[7] When the input set is reduced in this manner, the number of sources is reduced from 631,870 to 135,406.

[7] We recognize that the threshold random walk approach detects scans much more quickly than either our approach or that of the naïve algorithm, as it can detect a scan in as few as five packets as opposed to the 32 flows required for our approach.

Table 2: Number of Sources in each Group

Naïve	Our approach	TRW	# of Sources
Non-scan	Non-scan	Non-scan	124,354
Non-scan	Non-scan	Scan	491
Non-scan	Scan	Non-scan	417
Non-scan	Scan	Scan	1026
Scan	Non-scan	Non-scan	7413
Scan	Non-scan	Scan	96
Scan	Scan	Non-scan	182
Scan	Scan	Scan	1427

We randomly selected 104 samples—23 from each group in Table 2—to analyze. The same domain expert evaluated the raw flow data for each of the samples, identifying 69 of them as representing scanning activity. The results for each of the algorithms were compared to the opinion of three experts, with the results summarized in Table 3. The three experts agreed on 87 of the samples, labeling 24 of them as non-scans and 63 as scans. For the 17 cases where the experts disagreed, the majority opinion was used to determine whether the event was a scan, resulting in a further 5 non-scans and 12 scans.

The threshold random walk (TRW) algorithm has the lowest false positive and false negative rates of any of the methods. The TRW algorithm has a total of 23 misclassifications. We considered a combined approach using both TRW and our scan approach. If either algorithm classifies a source as a scanner, then the combined approach labels the source as a scanner. This method would result in only three misclassifications, a considerable improvement over TRW alone. However, the nature of the misclassifications has changed. The combined approach causes three false positives and no false negatives. The TRW algorithm has essentially no false positives in practice, but it will miss many scans. The combined approach misses essentially no scanners, but it will inaccurately classify a small number of sources as scanners. This indicates that the nature of the scans missed by TRW and our approach are sufficiently different that, when combined, there is very little overlap between the two sets of missed scans.

Table 3: *Number of False Positives and Negatives for each Algorithm.*

	Naïve	Our Approach	TRW	Combined
False Positives	13	3	0	3
False Negatives	36	26	23	0

It should be noted that the numbers provided in Table 3 cannot be used to determine an overall rate for each of the algorithms. For example, the false positive rate of 10.3% (3/29) obtained by our algorithm on this sample cannot be generalized—it does not indicate an overall false positive rate of 10.3%. This is because our sampling strategy was chosen specifically to ensure that the most difficult cases were analyzed, rather than the average case.

In order to determine what the false positive and false negative rates would be for the entire sample, we first assume that we analyzed a representative hour and that the stratified sampling approach represented each stratum effectively. To perform this estimation, we need to account for the frequency of the different strata in our data. For the hour under consideration, we had 135,406 hosts generating at least 32 flows each, where the frequency for each stratum provided in Table 2.

The majority of hosts (124,354) were not classified as scanners by any of the detection methods. To estimate the false positive rates, we must assume that the proportion of scanners to non-scanners found in our sampled strata (based on expert consensus) hold for those strata in the population of all hosts. Please note that we are estimating this proportion for each stratum using a fairly small sample size (N = 13 per stratum). Under this assumption, we would expect the overall false positive rate for our detection approach to be 0.073%. In other words, around 7 out of 10,000 non-scanning hosts would be misclassified as scanners using our approach. We estimate that 96 hosts would have been incorrectly classified as scanners during this hour. The false negative rate for our approach in the population of all hosts in the hour is estimated to be 16.567%. This means that our approach would fail to detect 587 scanning hosts in the larger population during this hour.

In comparison, we would expect an overall false positive rate to be nearly 0%, while the expected false negative rate would be 14.197%. Thus TRW would fail to detect 503 scanning hosts in the larger population during this hour. Combining TRW with our approach results in an expected overall false positive rate of 0.073%, while the overall false negative rate would be nearly 0%. Therefore considerable accuracy can be achieved by combining these two approaches.

6 Deployment Results

Our approach for detecting TCP scans has been implemented and in operational use on a large client network for several months, with summary information on each scan detected saved to a database. In this section we provide some sample results from this database.

Figure 2 provides the number of unique scans and scanning sources per hour for November and December 2005. (Note that there is some missing data during this time period, indicated by either gaps in the data or data points below 500.) In general, approximately 1500 unique sources are scanning this particular network during any given hour, a number that includes worms as well as scanning activity directed by a human actor. There are slightly more scans per hour, which indicates that some sources stop scanning for a period of at least five minutes, and then resume.

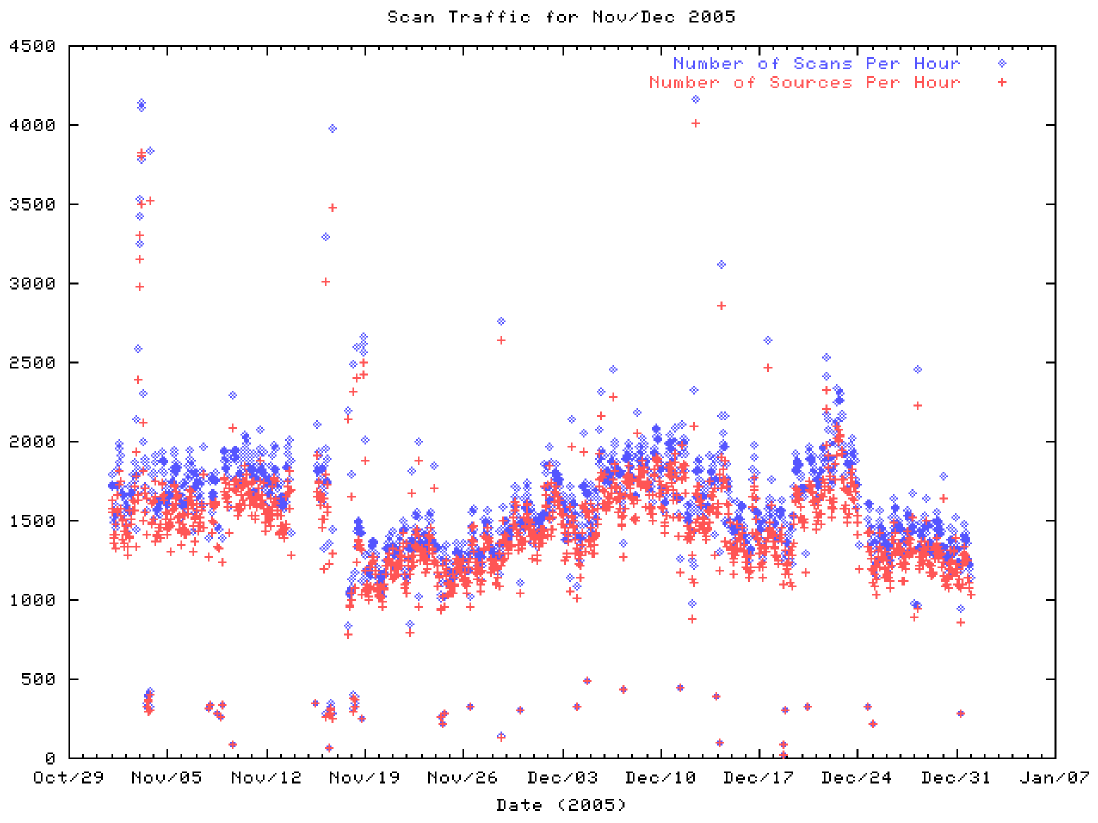

Figure 2: Scan Traffic for Two Months in Late 2005

The top ten most commonly scanned ports for this two-month period can be found in Table 4. Port 80 was by far the most commonly scanned port, both by number of scans and by number of unique sources scanning. A large number of high-numbered ports are being scanned, with eight of the top 10 ports being ephemeral ports. Figure 3 shows the scanning behavior for the top four scanning ports for November and December. Note that some interesting activity can be observed by just looking at these four ports. First, both ports 80 and 25 show an increase in the number of scans around Christmas. Second, scanning against port 1025 stops abruptly around December 14, 2005. Scanning to this port seems to be largely due to an exploit against the Microsoft Distributed Transaction Coordinator. The abrupt disappearance of this scanning appears to coincide with the release of a patch for the vulnerability. Finally, port 51736 suddenly becomes widely popular starting around November 20, peaking around December 11 and the dropping off sharply by December 15. It is not clear why there was a spike in scanning for this port.

Table 4: Top Ten Most Commonly Scanned Ports for November-December 2005

Port	# of Scans	# of Sources
15118	32,650	20,222
2745	37,661	9368
5000	64,038	18,815
3410	73,990	23,812
5900	84,121	8666
4899	102,174	17,579
51736	113,091	60,659
1025	152,140	44,462
25	212,214	39,927
80	504,016	146,470

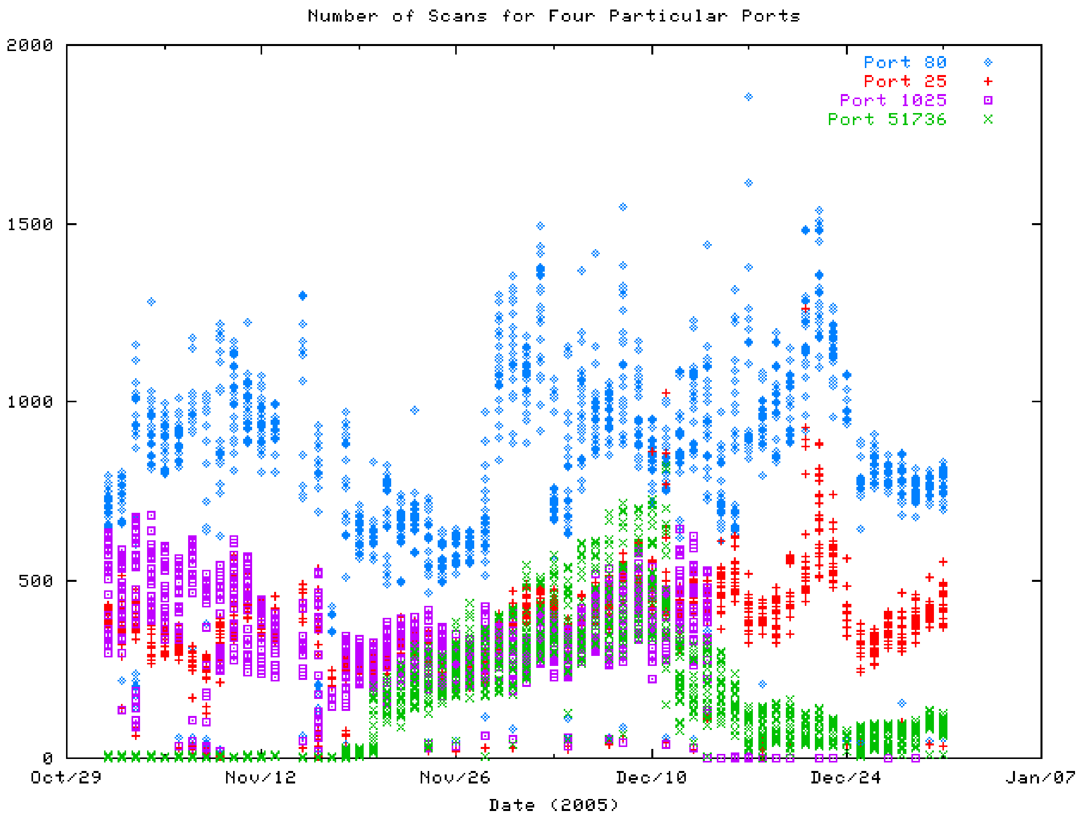

Figure 3: Trending Activity for Four Commonly Scanned Ports for November and December 2005

There are several possible uses for maintaining a database of scan data, including the three below:

1. Add scanning IP addresses to access control lists to block their communication requests at the border. Note that this requires a complementary white list of legitimate external hosts in order to prevent denial-of-service (DoS) attacks on the external hosts. It should be noted, however, that this is difficult to manage on large networks.

2. Investigate IP addresses where possible exploit attempts have been identified. When we process events for scanning activity, we retain a list of all those destinations where there appeared to be further contact based on the number of bytes and the flag combination of the flow. These hosts can then either be investigated to determine whether they have been successfully compromised, or watched for subsequent unusual activity that might indicate a compromise.

3. Perform periodic trend analyses to determine whether new exploits or vulnerabilities have been released/identified. For example, in Figure 3 port 51736 demonstrated a sudden increase in scanning activity. A periodic trend analysis could detect activity such as this, and could result in port 51736 being blocked at the border, for example.

Additionally, in the cases where a large amount of flow data is maintained and used for security analysis, those flows that represent scanning activity can be processed into a second set of data separate from the non-scan data. A security analyst can then focus on the traffic of interest, whether that is scan traffic or non-scan traffic. This reduces the search time for the analyst, by allowing him or her to specify whether the target of interest is in the scanning data set or the non-scanning data set.

7 Conclusions

This report presents a novel approach to detecting scans against, or passing through, very large networks. It is intended for use by organizations managing very large intranets or portions of the Internet backbone, as well as by smaller Internet service providers and intranets. Few other scan detection approaches are suitable for these operating conditions (described in Section 3.1). This scan detection approach performs an ongoing, incremental analysis of flow-level data about traffic inbound to the network. Our implementation successfully runs on very large data sets from live networks.

In addition to these important characteristics, this approach offers four key research contributions:

- It produces a probability that a collection of network traffic contains a scan; this is in contrast to the binary (yes/no) results provided by most other scan detection tools.
- It uses a flexible and extensible collection of scan indicators—each with an associated metric—rather then the typical one or two metrics used by many other scan detection methods and tools. Using a large number of varied indicators makes it more difficult for an adversary to "game" the system and thus avoid detection. A user can repeat our methodology, adding other indicators to the model that would improve the detection capability.
- It provides a method of scan detection based solely on unidirectional flow data. Thus, we require less information to be collected than other scan detection techniques, which require either packet-level or bi-directional flow information be provided in order to recognize scanning activity. Moreover, our approach can operate without knowledge of the monitored network architecture.
- It offers a model for careful, thorough, statistically grounded development and validation of a new intrusion detection approach. We have presented a methodology for generating a model that takes advantage both of expert opinion and data analysis, rather than being driven strictly by one or the other.

In this paper we presented a model that predicts whether or not a set of flows from the same source represents scanning activity, which we validated against a set of 300 randomly chosen events that were manually labeled as containing scans or not. For TCP scans we found that our model has 99.3% accuracy, with a sensitivity (true positive rate) of 95.5% and a specificity (true negative rate) of 99.6%.

We compared our approach to a modified version of the threshold random walk algorithm, using a sampling strategy that focused on events that were sufficiently unusual as to make classification difficult. A test set consisting of 104 such events were chosen, where our approach had a misclassification rate of 29 while the threshold random walk had a misclassification rate of only 23. This result is not surprising, given that the threshold random walk method has access to more information—specifically an oracle that knows which hosts exist—than does our approach, which uses only unidirectional flow information with no network knowledge.

We combined the two approaches and found that the misclassification rate dropped to three. This indicates that there is a disjoint set of scans that each approach misses, where each set has different scan characteristics. Thus, for those instances where both an oracle is available and the preference is for false positives over false negatives, a combined approach might be the best solution. Extrapolating our results to a typical hour, we expect that the combined approach would have a false positive rate of 0.073% and a false negative rate of nearly 0%.

References/Bibliography

URLs are valid as of the publication date of this document.

[Agresti 02] Agresti, A. *Categorical Data Analysis*. Hoboken, NJ: John Wiley and Sons, 2002. ISBN 1-471-36093-7.

[Akaike 73] Akaiki, H. "Information Theory as an Extension of the Maximum Likelihood Principle," (267-281). *Proceedings of the 2nd International Symposium on Information Theory*. Tsakhadzor, Armenian U.S.S.R., Sept. 2-8, 1971. Budapest, Hungary: Akademiai Kiado, 1973.

[Axelsson 00] Axelsson, Stefan. "The Base-Rate Fallacy and the Difficulty of Intrusion Detection." *ACM Transactions on Information and System Security*, *3*, 3 (Aug. 2000): 186-205.

[CERTCC 02] CERT Coordination Center, Software Engineering Institute, Carnegie Mellon University. "SiLK." http://silktools.sourceforge.net (2002).

[Cisco 06] Cisco Systems. "Documentation—NetFlow FlowCollector." http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/nfc/ (2005).

[Conti 04] Conti, Gregory & Abdullah, Kulsoom. "Passive Visual Fingerprinting of Network Attack Tools," (45-54). *Proceedings of the 2004 Workshop on Visualization and Data Mining for Computer Security*. Washington, DC., October 29, 2004. New York, NY: ACM, 2004.

[de Vivo 99] de Vivo, Marco; Carrasco, Eddy; Isern, Germinal; & de Vivo, Gabriela O. "A Review of Port Scanning Techniques." *ACM SIGCOMM Computer Communication Review*, *29*, 2 (April 1999): 40-48.

[Elfving 52] Elfving, G. "Optimum Allocation in Linear Regression Theory." *The Annals of Mathematical Statistics*, *23*, 2 (June 1952): 255-262.

[Fullmer 00] Fullmer, Mark & Romig, Steve. "The OSI Flow-Tools Package and Cisco Netflow Logs," 291-303. *Proceedings of the 14th Systems Administration Conference (LISA 2000)*. New Orleans, LA, December 3-8, 2000. Berkeley, CA: USENIX Association, 2000.

[fyodor 97] fyodor. "The Art of Port Scanning." *Phrack Magazine*, *7*, 51 (Sept. 1, 1997). Article 11 of 17. http://www.phrack.org/show.php?p=54&a=9.

[Gates 04] Gates, Carrie; Collins, Michael; Duggan, Michael; Kompanek, Andrew; & Thomas, Mark. "More NetFlow Tools: For Performance and Security," (121-132). *Proceedings of the 18th Large Installation Systems Administration Conference*, Atlanta, Georgia. November 14-19, 2004. Berkeley, CA: USENIX Association, 2004.

[Gelman 95] Gelman, Andrew; Carlin, John B.; Stern, Hal S.; & Rubin, Donald B. *Bayesian Data Analysis*. Norwell, MA: Chapman Hall (now Luwar Academic Publishers), 1995. eISBN 0-412-03991-5.

[Honeynet 02] The Honeynet Project. *Know Your Enemy*. Boston, MA: Addison Wesley, 2002.

[hybrid 99] hybrid. "Distributed Information Gathering." *Phrack Magazine*, *9*, 55, (Sept. 9, 1999). Article 9 of 19. http://www.phrack.org/phrack/55/P55-09.

[Jung 04a] Jung, Jaeyeon; Paxson, Vern; Berger, Arthur W.; & Balakrishnan, Hari. "Fast Portscan Detection Using Sequential Hypothesis Testing." In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 9-12, 2004.

[Jung 04b] Jung, Jaeyeon; Schechter, Stuart E.; & Berger, Arthur W. "Fast Detection of Scanning Worm Infections." *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection*. French Riviera, France, September 15-17, 2004. *Lecture Notes in Computer Science*, Berlin/Heidelberg: Springer 2004.

[Kruegel 02] Kruegel, Christopher; Vigna, Giovanni; Valeur, Fredrik; & Kemmerer, Richard. "Stateful Intrusion Detection for High-Speed Networks," (266-274). *Proceedings of the 2002 IEEE Symposium on Security and Privacy,* Washington, DC, Oct. 29, 2002. New York, NY: IEEE, Press, 2002.

[Lakkaraju 04] Lakkaraju, Kiran; Yurcik, William; & Lee, Adam J. "Nvisionip: Netflow Visualizations of System State for Security Situational Awareness," 65-72. *Proceedings of the 2004 CCS Workshop on Visualization and Data Mining for Computer Security*, Washington, DC, Oct. 29, 2004. New York, NY: ACM, 2004.

[McCarthy 94] McCarthy, Thomas C, LTC. "U.S. Army Heavy Brigade Reconnaissance During Offensive Operations," (monograph) 1994. http://quanterion.com/RIAC/Library/Library.asp?ArgVal=122856.

[McPherson 04]	McPherson, Jonathan; Ma, Kwan-Liu; Krystosk, Paul; Bartoletti, Tony; & Christensen, Marvin. "Portvis: A Tool for Port-Based Detection of Security Events," 73-81. *Proceedings of the 2004 CCS Workshop on Visualization and Data Mining for Computer Security*, Washington, DC, October 29, 2004. New York, NY: ACM, 2004.
[Moore 01]	Moore, David; Voelker, Geoffrey; & Savage, Stefan. "Inferring Internet Denial-of-Service Activity." In *Proceedings of the 2001 USENIX Security Symposium*, Washington, DC., August 13-17, 2001. Berkeley, CA: USENIX Association, 2001.
[Muelder 05]	Muelder, Chris; Ma, Kwan-Liu; & Bartoletti, Tony "Interactive Visualization for Network and Port Scan Detection." *Proceedings of Recent Advances in Intrusion Detection*. Seattle, WA, Sept.7-9, 2005. Berlin, Germany: Springer-Verlag, 2005.
[Navarro 00]	Navarro, John-Paul; Nickless, Bill; & Winkler, Linda. "Combining Cisco NetFlow Exports With Relational Database Technology for Usage Statistics, Intrusion Detection and Network Forensics," 285-290. *Proceedings of the 14th Systems Administration Conference (LISA 2000)*, New Orleans, LA, December 3-8, 2000. Berkeley, CA: USENIX Association, 2000.
[nmap 06]	nmap. http://www.insecure.org/nmap/ (1996).
[Panjwani 05]	Panjwani, Susmit; Tan, Stephanie; Jarrin, Keith M; & Cukier, Michel. "An Experimental Evaluation to Determine if Port Scans Are Precursors to an Attack," (602-611). *Proceedings of the 2005 International Conference on Dependable Systems and Networks*. Yokohama, Japan, June 28-July 1, 2005. New York, NY: Springer, 2005.
[Paxson 98]	Paxson, Vern. "Bro: A System For Detecting Network Intruders in Real-Time," *Proceedings of the 7th USENIX Security Symposium*. San Antonio, Texas, Jan. 26-29, 1998. Berkeley, CA: USENIX Association, 1998.
[Paxson 01]	Paxson, Vern & Staniford, Stuart. "Topics in Intrusion Detection." Tutorial presented at *The 8th ACM Computer and Communications Security Conference (CSS)*. Philadelphia, PA, Nov. 5-8, 2001. Downloadable through http://www.icir.org/vern/talks.html.
[QoSient 04]	QoSient LLC. Argus. http://www.qosient.com/argus/ (2004).
[Robert 04]	Robert, C.P. & Casella, G.. *Monte Carlo Statistical Methods, 2nd ed.* New York, NY: Springer-Verlag, 2004.

[Robertson 03] Robertson, Seth; Siegel, Eric V; Miller, Matt; & Stolfo, Salvatore J. "Surveillance Detection in High Bandwidth Environments," 130-139. *Proceedings of the 2003 DARPA DISCEX III Conference*, Washington, DC, April 22-24, 2003. New York, NY: IEEE Press.

[Roesch 99] Roesch, Martin. "Snort–Lightweight Intrusion Detection for Networks," 229-238. *Proceedings of the 13th Systems Administration Conference (LISA 1999)*. Seattle, Washington, November 7-12, 1999. Berkeley, CA: USENIX Association, 1999.

[Solar designer 98] Solar designer. "Designing And Attacking Port Scan Detection Tools." *Phrack Magazine*, *8*, 53, (July 8, 1998). Article 13 of 15. http://www.openwall.com/scanlogd/P53-13.gz.

[Staniford 00] Staniford, Stuart; Hoagland, James A.; & McAlerney, Joseph M. "Practical Automated Detection of Stealthy Portscans."*Proceedings of the 7th ACM Conference on Computer and Communications Security (CSS)*, Athens, Greece, Nov. 1-4, 2000. New York, NY: ACM Publishing, 2000.

[Staniford 02] Staniford, Stuart; Hoagland, James; & McAlerney, Joseph. "Practical Automated Detection of Stealthy Portscans." *Journal of Computer Security*, *10*, 1 (2002): 105-136.

[Swets 00] Swets, John A.; Dawes, Robyn M.; & Monahan, John. "Better Decisions Through Science." *Scientific American*, *283*, 4: (Oct. 2000) 70-75.

[Weaver 04] Weaver, Nicholas; Staniford; Stuart & Paxson, Vern. "Very Fast Containment of Scanning Worms," (29-44) *Proceedings of the 13th USENIX Security Symposium*, San Diego, CA, August 9-13, 2004. Berkeley, CA: USENIX Association, 2004.

[Weisberg 85] Weisberg, Sanford. *Applied Linear Regression*. Wiley Series in Probability and Mathematical Statistics, 2nd ed. New York, NY: John Wiley and Sons, 1985. ISBN 0271-6356.

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.			
1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE April 2006		3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Detecting Scans at the ISP Level			5. FUNDING NUMBERS FA8721-05-C-0003
6. AUTHOR(S) Carrie Gates; Josh McNutt; Joseph B. Kadane (Dept. of Statistics, CMU); & Marc Kellner			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213			8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-2006-TR-005
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116			10. SPONSORING/MONITORING AGENCY REPORT NUMBER ESC-TR-2006-005
11. SUPPLEMENTARY NOTES			
12A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS			12B DISTRIBUTION CODE
13. ABSTRACT (MAXIMUM 200 WORDS) Scans are often used by adversaries to determine the potential weaknesses in a target network or system prior to an intrusion attempt. In other cases, exploits are packaged with the scans themselves. This report presents a novel approach to detecting scans (including very stealthy scans) against, or passing through, very large networks. It meets operational requirements that are particular to detecting scans in ISP level networks. This scan-detection approach performs an ongoing, incremental analysis of flow-level data regarding traffic inbound to a network. It is multi-dimensional and flexible, based on up to 21 characteristics describing traffic collected from any single source. The report describes in detail a method developed to provide a probability that a particular traffic sample contains a scan. In validation testing using a manual analysis of traffic collected from a high-volume network, this method correctly classified 99.3% of TCP traffic samples. This report also compares this new approach to other scan approaches, particularly a naïve scan approach, based on simple thresholding, and a modified version of the threshold random walk approach, to which it performed comparably. Combining the random walk approach with the new approach produced very good results, reducing the number of false negatives to zero.			
14. SUBJECT TERMS scan, ISP level, scan indicators, detecting scans, cyber attack, TSP scan, UDP scan, ICMP scan			15. NUMBER OF PAGES 47
16. PRICE CODE			
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL