An Application of Information Theory to Intrusion Detection

E. Earl Eiland
Computer Science Department
New Mexico Inst. of Mining and Technology
Socorro, New Mexico USA
eee@nmt.edu

Lorie M. Liebrock, Ph. D.
Computer Science Department
New Mexico Inst. of Mining and Technology
Socorro, New Mexico USA
liebrock@nmt.edu

Abstract

Zero-day attacks, new (anomalous) attacks exploiting previously unknown system vulnerabilities, are a serious threat. Defending against them is no easy task, however. Having identified "degree of system knowledge" as one difference between legitimate and illegitimate users, theorists have drawn on information theory as a basis for intrusion detection. In particular, Kolmogorov complexity (K) has been used successfully.

In this work, we consider information distance (Observed_K − Expected_K) as a method of detecting system scans. Observed_K is computed directly, Expected_K is taken from compression tests shared herein. Results are encouraging. Observed scan traffic has an information distance at least an order of magnitude greater than the threshold value we determined for normal Internet traffic. With 320 KB packet blocks, separation between distributions appears to exceed 4σ.

1. Introduction

We live in a rapidly changing world. One of the forces driving change is the phenomenal increase in information available via the Internet. An additional challenge is controlling information access. Denning [11] discusses the problem at length, partitioning it into three categories commonly called CIA. Summarized, they are:

1. *Confidentiality:* Information is available only to authorized users in an approved manner and at approved times.

2. *Integrity:* Information comes only from trusted sources, without repudiation and can be guaranteed not to have been altered.

3. *Availability:* Authorized users have access to needed information and services in a timely manner.

Confidentiality and integrity are compromised by unauthorized access to proprietary information and services. Availability degrades because of denial of service attacks or destruction of data. Cost estimates to businesses alone range from $13B to $226B in 2003 [7]. The cost to consumers is not considered in these estimates.

Protecting an information system is an ongoing process. System administrators (sysadmins), in attempts to solve known weaknesses, upgrade their systems and import more problems. Even the patches — software modifications designed to block known holes — often create new vulnerabilities or attack opportunities for intruders [5]. An on-line search using the key words "vulnerability, patch and hole" will turn up thousands of known problems. Sysadmins can protect against known vulnerabilities; detecting and dealing with known attacks is routine. Their worst nightmare is to be attacked by some entity exploiting a previously unknown vulnerability. Commonly called Zero Day attacks, these new exploits may bear little resemblance to known attacks and thus they are difficult to detect.

After-the-fact forensics can recreate the crime. This, however, provides the sysadmin no opportunity to protect their information system against a Zero Day exploit in progress. The ability to detect a Zero Day event requires being able to differentiate between normal and new intrusive system activity in real-time. This kind of detection is radically different from detecting known attacks, where the attacks are well-defined and attack signatures well-established. Differentiating normal activity from intrusive is no easy task, however. Normal activity on one system or for one user, will frequently vary over time. Normal performance characteristics in one setting or at one time, may well be anomalous in another setting or at another time. There is also the

problem that most anomalous system performance is not caused by a Zero Day or any other kind of exploit.

Consider two database queries. The first, a frequently used query is executed by an authorized user; this is normal activity. The same query executed by an unauthorized user may be malicious. Or perhaps this query, applied to a poorly designed new database or poorly patched existing database, causes an error condition or system crash. What was (and may still be in another setting) normal traffic now becomes an exploit in the right (or perhaps we should say wrong) hands. Normal traffic has become malicious or indeterminate (normal or malicious, depending upon the context). In response to a system change, a second query is instituted. Because it's new, it could be considered, at least when first used, anomalous. This query is, however, normal. The intrusion detection challenge is to alarm on the first, but ignore the second — at best a very difficult task. The success with which intrusion detection systems (IDS's) achieve these separations is measured as "False positives" — % innocuous traffic flagged as malicious and "False negatives" — % malicious traffic not flagged.

This paper's contribution is to introduce a detection algorithm for anomalous intrusions that promises to reduce false positives and negatives substantially.

2. Organization

The rest of this paper is divided up as follows: Section 3 — a review of previous anomalous IDS developments; Section 4 — a discussion of conventional Intrusion Detection System (IDS) protocols; Section 5 — presentation of a new IDS concept; Section 6 — applying the new IDS concept; Sections 7 & 8 — issues using data compression to develop the proposed metric; Section 9 — our experimental objective; Section 10— our experimental procedure and results; Section 11 — identifying the optimum packet block size; Section 12 — real-world issues; Section 13 — how our results can be used; Section 14 — future work; Section 15 — conclusion. An Appendix relates our metric to low frequency attack detection.

3. Previous Work

In spite of apparently insurmountable difficulties, unknown (anomalous) exploit detection has received a significant amount of attention, using an assortment of approaches. There are numerous IDS's available, some commercial, others free. To list them and discuss their various features is well beyond the scope of this paper. The on-line list provided by the Center for Education and Research in Information Assurance (CERIAS), located at Purdue University [10], is probably one of the most complete available. It lists over sixty IDS's, many of which offer some kind of anomalous attack detection capability.

Detection approaches are seemingly as numerous as the tools available. Some are static, others are dynamic, adjusting to changing conditions. Machine-learning techniques are popular. Many recent anomalous IDS developments are statistically based, although the features selected cover the gamut. Data mining [28], time-based [32] and Mahalanobis distance [30, 39] are just a few.

This attention is well justified. Because anomalous exploits are new, they take advantage of previously unknown information system vulnerabilities. Systems containing any previously unknown vulnerability are suddenly wide open for attack. The current lack of effective real-time detection tools for unknown attacks can leave sysadmins feeling powerless.

4. Conventional anomalous exploit detection

The first challenge in detecting anomalous exploits is defining what to measure. What metrics will be useful in identifying malicious traffic? Work has focused on two areas: system state and system activity (traffic) [20]. The latter term is broadly defined to include everything from messages and data transfers internal to a specific machine to incoming Internet packets received by a network firewall. Traffic-based metrics offer the earliest warning. Traffic is captured and analyzed before reaching its destination, when it has yet to impact the system. This is not true with system-state based metrics. Before a system state change can be detected, an attack must have been at least partially successful. This difference makes traffic-based detection obviously preferable.

For the most part, traffic-based detection can be summarized as packet substring analysis. Frequently, these analyses have a time sensitive component (e.g., log-in time, number of log-in attempts, number of packets received, etc.) or a location sensitive component (such as user log-in location, communication hops, packet source address, etc.)[20]. The problem is that, individually, these substrings generally are innocuous. Suspicious activity can only be detected by considering a wide array of information jointly. In so doing, however, the uncertainty becomes so substantial that even sophisticated analytic techniques yield unacceptably high detection error rates [28]. The challenge is so

great that feature selection has become an important research topic in its own right [8, 38].

One would hope that malicious traffic is not the norm, but can be considered to be a separate, infrequent traffic class that merges with the predominant, normal class. In this situation, statisticians call this malicious traffic a contaminant [1] of normal traffic. Contaminants have their own probability distribution functions (pdf's) and are visible only to the extent they impact the target distribution. One way they are visible is as outliers. (Outlier analysis is decidedly different from population-based analysis. It even has its own label, discordancy [1].) Restating the problem pointed out in the previous paragraph, the dominant problem in anomalous intrusion detection is related to the normal and malicious traffic pdf's of the factors measured. The pdf's aren't distinct enough and malicious traffic is such a small fraction of the total, that it perturbs the expected pdf very little. In this situation, malicious traffic is all but indistinguishable. One way to address this issue is to identify metrics where the pdf's are so distinct that even a small amount of contamination is observable. This is the approach we take to evaluating our potential metric.

While traffic-based detection might seem limited to external attacks on an information system, the approach really has much wider application. We believe that the proper metric, with appropriate modification, can be used to detect suspicious traffic between any two information system components or subcomponents. Communication can be via an internal bus, an intranet or the Internet. Internal use could aid in detecting insider attacks, as well as provide a second line of defense against outsiders. We briefly explore issues and opportunities in Section 13.

5. An Information-theoretic paradigm

Those who specialize in measurement theory emphasize the need for a theoretical framework [35]. Because the digital universe is new, it is poorly characterized. A theoretical basis relevant to malicious activity and intrusion detection is lacking. We have found no mainstream intrusion detection systems (IDS's) that are based on a model which differentiates normal from malicious activity. Researchers such as Chebrolu and Sung [8, 38] are dealing with descriptive measures, which typically lack an information-theoretical basis. Recently, some researchers have begun to wonder if the lack of a theoretical framework could be the cause of anomalous intrusion detection systems complexity and poor performance in detecting unknown attacks. Work at the University of North Carolina [29] successfully applied information theory to the problem. Bush and Evans [5] developed a physics of information primarily based on information theory. Among many other results, they successfully differentiated 95% of malicious FTP traffic from normal with only .02% false positives [12].

Evans only had 5% false negatives, using a single metric and a simple (single) rule-based detection algorithm. A similar test using twenty-three metrics and sophisticated detection algorithms reported almost 46% false negatives with 2% false positives [28]. Evans' approach returns almost an order of magnitude improvement in detection. Applying Occam's Razor

> Among all hypotheses consistent with the facts, choose the simplest[23, 26],

it is hard *not* to conclude that information theory deserves scrutiny, at the very least.

One may well wonder why information theory seems to successfully address the seemingly intractable anomalous intrusion detection problem. One answer may be, as discussed at length by Bush and Evans [5], that it takes advantage of one difference between legitimate users and a common type of unauthorized users — system knowledge. Often an intruder initially has little or no knowledge about the target information system [5]. However, based on study of attacker behavior [21], we can infer that an intruders' likelihood of successfully achieving their goal increases significantly as their system knowledge increases. Much as a blind person sweeping the path in front of them with a cane to get the lay of the land, this class of intruder must feel their way around the system. It's not unusual for them to probe a system during an attack. These probes are atypical — legitimate users have little need to probe the system. They can easily navigate the system and promptly access the desired information or activate the desired service.

As noted earlier, information-theory-based metrics — especially Kolmogorov-complexity-based metrics (K) — have been successfully used to detect specific types of malicious traffic. It's also been used to assess information system vulnerability. We believe that just as K can identify system vulnerabilities [5], K can be an effective metric for detecting anomalous intrusions. Our initial phase builds on Bush and Evans observations regarding an intruders lack of system knowledge. Probes sent out by intruders have a lot of similarity to radar and sonar scans. Radar and sonar operators learn of a target's existence when they receive a return signal. The signal structure gives them information about the target's nature. In the same way, intruders learn of a targets existence when they receive a

return packet. From the packet's contents (both payload and header), the intruders learn about the system's makeup. In the case where the intruder knows almost nothing about the target system, initial probes test for openings into the system. Packets are sent out to specific address ports in hopes of some response. Just as radar and sonar scans are detectable by those being scanned, we believe scan packets are detectable.

5.1. Information theory and intrusion detection

Bush and Evans dedicate many pages to developing a physics of information and applying it to intrusion, intruders and malicious traffic [5]. That work is too lengthy to share here, but ultimately they choose Kolmogorov complexity as a key concept. Kolmogorov complexity (K) describes a mechanism for identifying information density of a string [24, 25]. As defined in [9],

> the Kolmogorov complexity $K_{\mathcal{U}}(y)$ of a string with respect to a universal computer $\mathcal{U}$ is defined as
>
> $$K_{\mathcal{U}} = \min_{p:\mathcal{U}(p)=y} l(p),$$
>
> the minimum length over all programs that produce y and halt. Thus $K_{\mathcal{U}}(y)$ is the shortest description length of y over all descriptions interpreted by computer $\mathcal{U}$.

Entropy, the more common information density metric, was passed over for good reason. It relates to the expected value of information density going through a channel. One would hope that malicious traffic is not the norm, but as discussed previously, would usually be considered a contaminant [1] of normal traffic. In this case, detecting malicious traffic requires differentiating strings. Rather than working with averages, intrusion detection requires specific values for each string: the very thing measured by K.

It's important to note that in our context, the length (l) of Kolmogorov's program (p) really consists of two parts:

$$l(p) = l(software) + l(y'),$$

where $software$ is the collection of routines on the universal computer $\mathcal{U}$ and y' is the $software$'s input string that ultimately outputs y. Fortunately, data compression is an active field of study applying this very concept. For any specific string y, a compression algorithm can define a custom alphabet and a (compressed) string using that alphabet. The alphabet and compressed string together, then, represent the maximum information density for the target string available from that compressor. (K is actually immeasurable. In general there is no guarantee that any given compressed string is the best. There is always the possibility that another, more concise expression exists.)

By definition, K includes the program length. Ideally, each output string y would be created by a custom input string y' and custom program p. For our purpose, however, data compression uses a general purpose program of fixed length ($l(software) = c$). In the general case, as $l(y)$ increases, so will $l(p)$. However, since the software length is fixed, its contribution to $K(y)$ diminishes:

$$\lim_{y\to\infty} K_{\mathcal{U}}(y) = l(y').$$

Since the program's contribution approaches zero, we can ignore the program length and focus solely on the compressed input string.

So far, information-theory-based metrics have proved to be quite versatile. Various Kolmogorov complexity aspects have been used:

- to detect Distributed Denial of Service (DDoS) attacks [27],
- for information system vulnerability assessment [14],
- to detect viruses [18],
- to detect FTP attacks [13].

Mutual information, another information theory concept, has been used to detect UDP flooding attacks [19]. All of this work has successfully demonstrated that K is useful for measuring a wide assortment of very specific information assurance tasks. Because of its very versatility, we suspect it is also useful for more general issues. We will test Kolmogorov complexity's utility as a pre-selection tool for a general purpose IDS. In particular, we shall focus on detecting a K characteristic common to anomalous intrusions. This is an endeavor ideally suited to K. Due to their novelty, anomalous intrusions are quite difficult to detect using conventional measures.

6. Applying Kolmogorov complexity to anomalous intrusion detection

Compared to normal traffic, malicious traffic is a contaminant and thus generates outliers to the normal traffic pdf. Quantifying just how much K for any particular packet varies from the norm will indicate that packet's degree of discordancy. This may also be a useful flag for an intrusion. We can determine this by comparing target traffic K to a baseline. (In a

real-time test, the target traffic would be the current traffic.) The more the target K varies from the baseline/normal value, the greater the likelihood the target is or includes, malicious traffic. Information distance (E) provides us the appropriate concept [2]. The simplest value would be *absolute distance*, the difference between the actual and expected values of K;

$$E = K(actual) - K(expected).$$

There is a problem with this value however. Network traffic packet sizes vary greatly. Comparing results for different packets will be difficult. Also, should different units be used, the observed values would have to be converted. A unit-less value resolves both of these considerations. We shall use *relative distance* (E_R) instead;

$$E_R = \frac{K(actual) - K(expected)}{K(expected)}.$$

Since $K \approx CompressedFileSize$, we can restate relative distance as

$$E_R = \frac{\Delta CompressedFileSize(actual - expected)}{CompressedFileSize(expected)}.$$

E_R can then be tested against positive and negative threshold values. Any packets exceeding a threshold will then be flagged for further examination. The actual $CompressedFileSize$ is measured directly. Expected $CompressedFileSize$ is calculated from the efficiency function ($TargetFileSize = f(CompressedFileSize)$) for the specific data compressor in use.

7. Compression process overview

Each compression algorithm has its own distinct process, however all compressors follow a similar protocol. Briefly stated, compressors search data streams looking for repetitive sections that can be replaced with shorter flags. The original file, then, is broken into two parts:

- a structured partition containing a list, dictionary or other data structure allowing the algorithm to reconstruct the repetitive portions.
- an unstructured partition that contains the non-repetitive, random portion of the data stream.

It's important to note that depending on the compression algorithm's sophistication, there may be undetected repetitive sub-strings remaining in the unstructured partition. This undetected repetitive structure and the compactness of the structured partition are the basis for the major performance differences between compression algorithms.

8. Compression efficiency measures

Data compressors are presently evaluated by comparing compression ratios and execution speed (for compression and decompression) on corpora — fixed bodies of data. The most commonly used data set is the Canterbury corpus [36]. This corpus is divided into a number of different data types, each of which is compressed separately. The compression ratio, as well as compression and decompression times, are measured for each file. These data are then compared with results for other compressors to get an idea of relative performance. Such test results are publicly available [3, 17, 34, 31, 36].

A number of factors affect a compressor's performance:

- *Compression algorithm:* The major factor affecting compressor performance is the process by which the compression takes place. This is beyond the scope of this work, but Nelson is a good online algorithm overview [34].
- *Target file size:* In a perfect world, compression efficiency increases with file size — asymptotically approaching a maximum defined by the file type and compression algorithm. This is because patterns identified in the structured partition can be repeated more in a large file. Also, more repetitive structures are likely to be identified.
- *File overhead:* As with all files, compressed files have overhead. Some overhead is Operating System and/or File System dependent. The balance is necessary for the decompressor. Compressed files have a compression algorithm identifier, decompression switch settings and data on the structured and unstructured string portions. For practical purposes, file overhead is fixed in size and thus it affects small files most significantly.
- *Compression software design trade offs:* The ultimate compression achieved by any compressor is directly associated with the CPU time consumed in that effort. As a rule, the longer a compressor works, the better the compression will be. Unfortunately, waiting is contrary to human nature. The slower a compressor works, the less acceptable it is to the market. Market acceptance and usability (in a real-time application, for instance) are directly impacted by how fast the compressor executes. Programmers have to choose a balance for their compression product. One way of doing this is to limit the size of the structured partition. On large files, the structured partition

will fill up prematurely, causing some otherwise detectable repetitive sub-strings to remain in the unstructured partition. This reduces compression efficiency on large files. In this study, we are compressing blocks of concatenated Internet packets. Since structures common in initial packets in any block may be rare in following packets, the adverse affect on compression efficiency may be especially noticeable.

As a consequence, compression efficiency varies with file size. Published, corpora-based data may be sufficient for many needs, but fall short for this work. We need a compression rate versus packet block size performance baseline for normal Internet traffic.

9. Experimental objective

Based on work done by Evans [13], we suspect that the measure we propose will detect many attacks. Mixing attack types in the same data stream, however, introduces a potential confounding variable —one that can hide otherwise significant results. We therefore limit this study to one attack process, system scans.

There is a benefit to starting with system scans: they often precede intrusive activity. For a sysadmin, knowledge of a scan in progress is valuable, because this may herald a more ominous event. Such information would give the sysadmin's time to increase intrusion surveillance efforts.

10. Experimental procedure

In this section, the process used in this investigation is discussed.

10.1. Characterize compressor performance on ordinary Internet traffic

1. *Capture network traffic* Using pcap [22] — a packet capture utility — network traffic was captured at the University gateway router. Due to the extreme variability of network traffic, six 700Mb samples, a through f, were taken at randomly selected times on two different days. The traffic was then filtered into three partitions: *incoming*, *outgoing* and *inside only*. This was done to better match traffic that an IDS outside an intranet firewall would actually receive. Of special concern was limiting potential confounding effects. For example, *inside only* traffic would never be transmitted on the Internet. Including this data in test samples could skew the results. *Outgoing* and *inside only* traffic could be useful for considering other intrusion issues. *Outgoing* traffic might be used to study zombie detection in a system. *Inside only* traffic could be used for studying insider attack detection. These and other potential uses are briefly discussed in Section 13.

2. *Build packet blocks* Compressing numerous small files could take an unacceptable amount of time. Also, due to unavoidable overhead, compression efficiency decreases as file size decreases. In order to speed the process, we decided to use packet blocks — packets concatenated in arrival order during a specific time interval (time slice). An alternate blocking scheme would have been to concatenate packets until the block exceeds a predetermined size. We chose time slices for three reasons.

   - Using the system clock generates somewhat less overhead, thus speeding the blocking process slightly.
   - Block size varies, depending upon traffic rate and packet block size. This information may be a useful intrusion detection factor.
   - A clock-based grouping limits processing delay.

   Along with the potential benefits of using time-slice generated packet blocks, there are three potential problems:

   - Sensitivity would be reduced, since malicious packets would likely be included with ordinary traffic. However, we anticipate this will be partially offset by the increased compression range that the packet blocks would fall in. Increased range means increased sensitivity. Figure 1 illustrates this point. Some discretion will be necessary to balance sensitivity and speed; calibrating compression rate versus file size will be a key to this decision.
   - Results could be masked because of the variation in compressor efficiency for different file sizes. We anticipate our clock-based grouping to generate a wide range of packet block sizes. This makes calibrating our compressor all the more important.
   - The median packet block size may vary considerably over time. Short-term — over time intervals of an hour or so — traffic frequency is random. Long term, however (e.g. a day or week), noticeable patterns emerge. For a

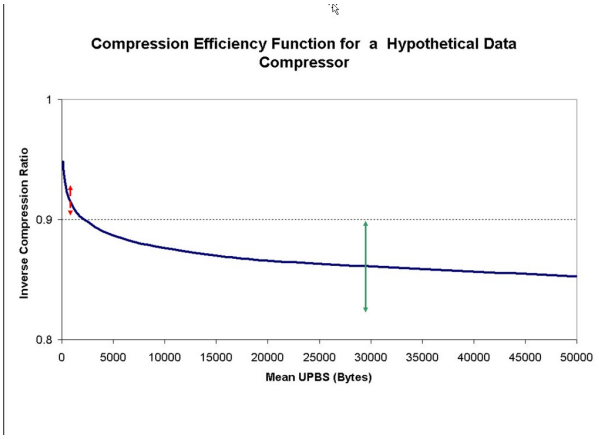


Figure 1. The vertical lines at 1 KB and 30KB show mean compression efficiency for two different packet block sizes. In this example, the range for the more compressible 20KB packet blocks is about three times that of 1 KB blocks. We believe that for any given string mix, compression is relative. Therefore a lower Inverse Compression Ratio (ICR) will result in a wider compression distribution for individual blocks. The increased distribution will increase sensitivity.

business, traffic may all but stop after closing, while traffic at a residence may well peak in the evening. Using a fixed time slice may result in unacceptable packet block size variations. Since our data sets consist of two separate channels (incoming and outgoing) collected at random times, we base the time slice for each set on the average bytes/second over the collection period.

There were two other potential confounding variables that could have impacted our results:

- On the occasions when a packet spans time slices, starting in one, but ending in another, we chose not to fragment packets. Packets were blocked based on their arrival time. The concern here was building the dictionary in a consistent manner. Starting a block with a packet fragment could well impact final compressed file size. Such an effect has been observed with the LZ78 compression algorithm [15].
- One issue specific to pcap files was that pcap overhead is attached to each packet. This was stripped off prior to bundling, as it would not be present in normal network traffic.

3. *Compress packet blocks* Each packet block was compressed by the targeted compressor and the Inverse Compression Ratio (ICR) was calculated.

$$ICR = \frac{CPBS}{UPBS}$$

$CPBS$ stands for the compressed packet block size, $UPBS$ stands for the uncompressed or target packet block size. ICR was chosen because in the general case, it's bounded by zero and one, as opposed to the Compression Ratio, which ranges from one to infinity. (With extremely small files, compressed files can actually end up larger than the uncompressed versions [15]). The occurrence of this condition varies from compressor to compressor. Our time slice value was chosen to avoid generating compressed packets with ICR's greater than 1.

4. *Analyze data* As discussed earlier, the classic, corpora-based analytical approach would not produce the necessary compression efficiency information. Consequently, statistical procedures were used extensively.

    - *Data distribution;* The first step was to examine the distribution of data set sizes. Distribution matching showed that the data are a good fit to the Weibull distribution [37], with a correlation around 98%. Because the raw data are well described by the Weibull distribution, distribution of the calculated value of interest (ICR) can be derived. While outside the scope of this paper, the moments for the ICR distribution can be shown to exist. Drawing on Casella & Berger's expression of the Central Limit Theorem (CLT) [6], the CLT can be applied to both the packet block size and the ICR. Applying the CLT allows us to work with sample means, which have a Normal (Gaussian) probability distribution function. This both simplifies and improves the quality of the analysis. As a result, the error for all means used in this work (at a 95% confidence) are consistently within 1%.
    - *Data compressor efficiency:* The next step is to define the efficiency function for the chosen data compressor. In order to visually identify the form the compressor efficiency function will take (such as exponential or polynomial), the initial analysis considered ICR over packet block sizes ranging from 10 KB

to 19 Mb for one data set. Compression was with Win_RK, a highly rated data compressor. When displayed on a log-log graph, a linear relation appears. This can be seen in Figure 2. There is a separate graph for each data set, showing results for incoming and outgoing traffic. Comparing sets, we find two similarities: all datasets have almost the same slope and the ICR's are similar. Such a relation suggested a power function ($y = bx^a$) would describe the packet block size-ICR relation.

- *Compression efficiency:* Win_RK has been applied to the full suite of data sets, with extremely consistent results. b from the above power function can be taken as the theoretical starting point for data compression of a one byte file. This is a theoretical value. In no case can a tiny file be compressed much. a in the power function appears as the slope in Figure 2. Of the two constants, a most greatly describes the performance curve. As compression performance improves, the slope increases. For this reason, a is the dominant factor for our study. Although b varies somewhat across the data sets, the lines are virtually parallel, indicating an invariable slope; all of the data sets shown have virtually the same a. Although actual compression may vary from packet block to packet block, we can confidently conclude that compressor performance is nearly constant for blocked packets. Pooling the results from all data sets, we calculate the mean power function for normal Internet traffic compressed by Win_RK:

$$ICR = 0.9908\ UPBS^{-0.0202}.$$

  Figure 3 shows the efficiency curve and upper and lower bounds for the 95% confidence interval.

  Figure 3 also suggests that our earlier stated belief that a lower ICR would mean a wider ICR range for packet blocks was wrong. In this case, however, being wrong was good. The narrower distribution actually strengthens our results. Investigation into why the ICR range stays relatively constant is ongoing.

- *Packet block size:* With a baseline established, we can now identify an optimum packet block size. All things being equal, the inflection point of the curve in Figure 3 (median block size, 20k bytes) seems a reasonable guess. We judge that this point is a reasonable balance between processing speed and granularity. Packet block size increases beyond this point yield minor improvements in compression, but continue to degrade sensitivity and adversely impact runtime performance. Ultimately, however, the optimum packet block size will be the one at which we find the greatest relative distance between normal and attack traffic. The optimal packet block size is discussed further in the next section. The ideal case, however, is when the same packet block size is equally effective for all types of attacks.

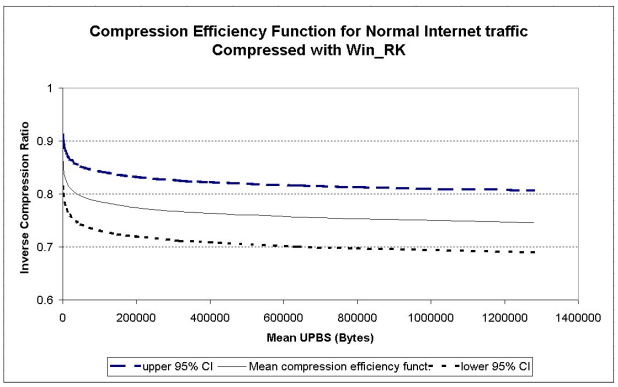


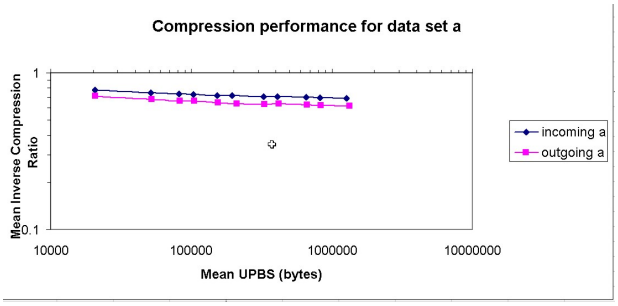
Figure 3. The three curves on this graph show the power function for normal Internet traffic when compressed with Win_RK. The middle curve is the mean, the outer two curves bracket the 95% confidence interval.

10.2. Characterize compressor performance on malicious Internet traffic

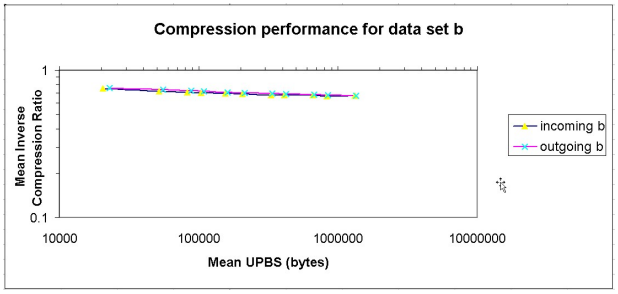
The characterization procedure for malicious Internet traffic is the same as for normal Internet traffic with one exception, data acquisition. Capturing pure malicious traffic "in the wild" attacking an ordinary information system is difficult. Generating malicious traffic against an enterprise information system is politically incorrect. We avoided these issues by collecting data from a red team / blue team (attack / defense) test being run in a controlled environment.

A variety of scan techniques were used:

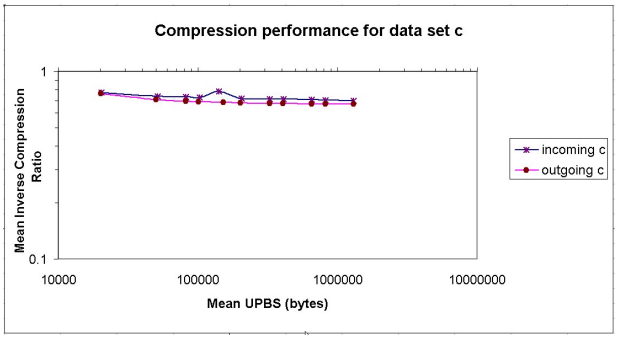
- TCP SYN scan (nmap -sS) [16]
- TCP Xmas scan (nmap -sX) [16]



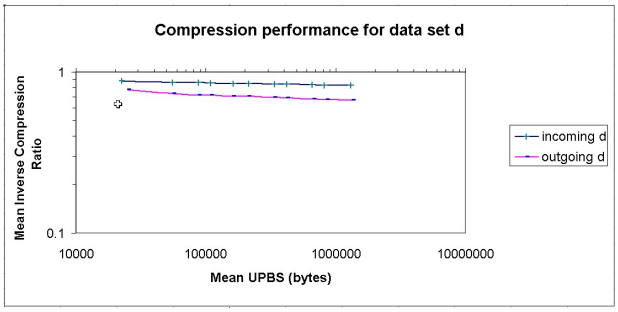
(a) Compression efficiency for data set a.



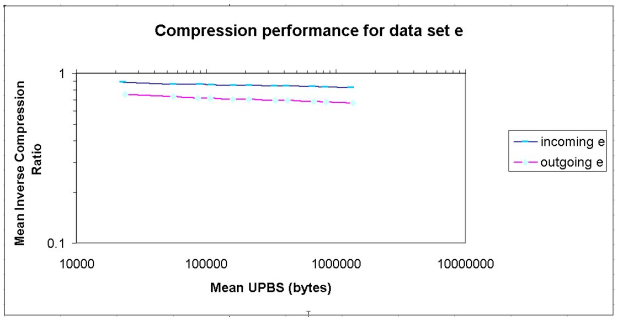
(b) Compression efficiency for data set b.



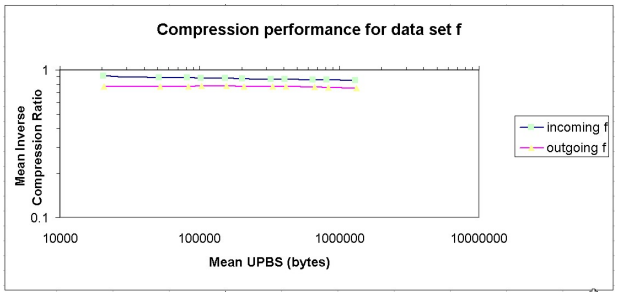
(c) Compression efficiency for data set c.



(d) Compression efficiency for data set d.



(e) Compression efficiency for data set e.



(f) Compression efficiency for data set f.

Figure 2. Each of these log-log graphs represents the ICR vs. file size relation for one data set. Comparison of the lines for each set shows that each relation has a similar slope. Also, the ICRs are constrained to a relatively narrow range.

- The default HPING2-based scan set [4]

The tests generated a little over 2Mb of scan traffic. In contrast to the 5+Gb of normal traffic, this is a tiny data set. This did result in greater uncertainty for scan traffic than for normal (This is visible in the width of the 95% confidence intervals in Figure 4). As shown next, however, we did have sufficient data to give us significant results.

Following the analytic process established for normal Internet traffic, we determined the power function for scan traffic when compressed by Win_RK:

$$ICR = 0.2297\ UPBS^{-0.00576}.$$

Figure 4 shows how the efficiency curves for normal and scan Internet traffic compare. A substantial gap can be seen between the upper three lines, representing the 95% confidence interval band for normal traffic and the lower three lines representing the 95% confidence interval band for scan traffic. Determining the probability of the normal efficiency curve and the scan efficiency curve being the same was beyond the capacity of our statistical software. We can say, however, that the probability is less than $1 \cdot 10^{-4}$.

Figure 5 shows how our proposed metric, relative distance, separate normal and scan Internet traffic. Not surprisingly, analysis using the relative distance metric we propose from Section 6 (shown in Figure 5) tells a similar story to that told by ICR. The median relative distance for normal traffic should fall within +/- 0.076 of the observed value 95% of the time, even with a 1.2Mb packet block size. The band narrows somewhat as packet block size decreases — the 20KB packet block median containing normal traffic would fall within +/- 0.056 95% of the time. The important feature of this graph is that the expected relative distance for scan traffic stays almost constant across the data range. This brings into question the value of calculating relative distance.

11. Optimal Packet Block Size

The operational question of concern to sysadmins is "How many false alarms (false positives) will I have and how many scans will go undetected (false negatives). Figure 6 shows actual data for the "20k" time slice. An inspection of the raw data is encouraging. Even though the small size of our scan traffic doesn't give as fine a granularity as our normal traffic sample, we can make some conclusions. If we set the *scan_traffic_ICR* : *normal_traffic_ICR* threshold at .53 – above the 99% CI for scan packet block ICR, then less than 1% of the scan packet blocks will register a false negative and less than

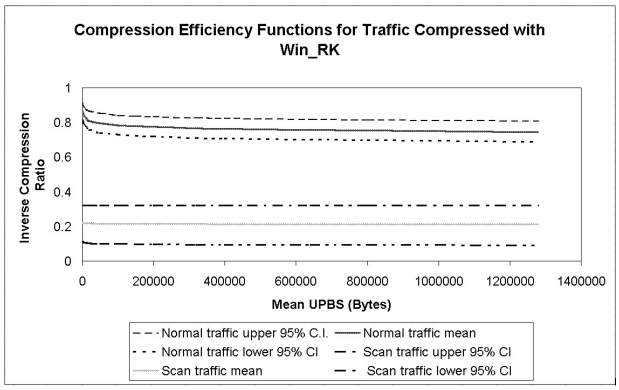


Figure 4. The upper of the two sets of curves show the mean (center top set) and upper (upper top set) and lower (lower top set) 95% CI boundaries for normal Internet traffic. The lower set shows the mean (center bottom set) and upper (upper bottom set) and lower (lower bottom set) 95% CI boundaries for scan traffic. There is a substantial gap between the two sets of lines. This suggests that complexity may be useful in detecting scanning activity.

8% of normal packet blocks will register a false positive. At a 95% CI for scan packet blocks, the false positive rate drops to 0.3%. This is in line with those reported by Bush and Evans [5] in their work on Kolmogorov-complexity-based intrusion detection of FTP attacks.

Comparing Figure 4 and Figure 6, there is seemingly, a discrepancy between calculated and observed results. Based on our statistical analysis, it would appear that the likelihood of a false positive is 38σ and false negatives have a likelihood of 8σ. Both are effectively zero. This is because the values shown in Figure 4 show the expected median value. The Central Limit Theorem is one of the better tools for calculating expected values. The CLT, however, looses the actual data distribution. The all normal packet block ICR distribution shown in Figure 6 is strongly left skewed and the all scan packet block ICR distribution strongly right skewed. The tails overlap, thus there is some risk of false positives and negatives.

The 20KB packet block size promises a significant improvement over conventional anomalous intrusion detection metrics. However, we may be able to do better. Figure 7 shows the difference between the upper scan traffic ICR and the lower normal traffic ICR 99% CI limits for packet blocks ranging from 20KB to 400 KB. The 99% CI's for 20KB packets overlap. At 50KB, the overlap is gone and the separation rises asymptot-

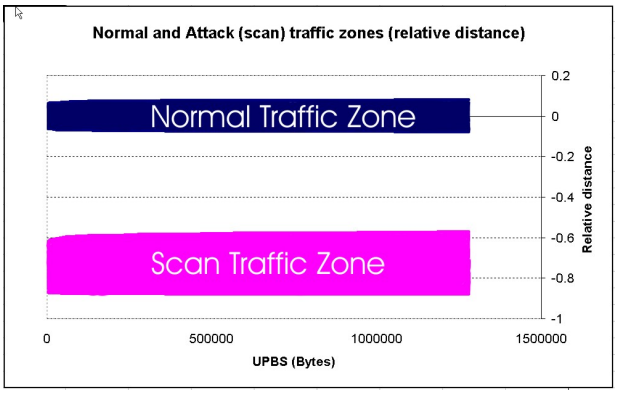


Figure 5. The upper band indicates the 95% CI range for the relative distance of normal Internet traffic packet blocks. 95% of normal Internet traffic packet blocks will have a relative distance between those two lines. The bottom band shows the relative distance range for the relative distance of scanning packet blocks (at the 95% CI). As can be seen, the two bands are widely separated.

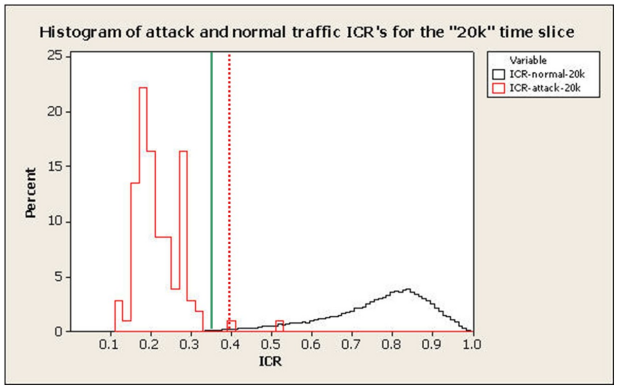


Figure 6. The dashed vertical line indicates the upper 99% CI for the scan traffic ICR (the leftmost distribution). The solid vertical line indicates the lower 99% CI for the normal traffic ICR (the rightmost distribution). Since the dashed line is right of the solid line, the two distributions overlap at the 99% CI.

ically up to a limit around 0.2 ICR. From our data, it would appear that there would be little benefit to using a packet block size above approximately 250 KB. The graphs in Figure 8 show how the normal and scan packet block ICR distributions diverge as block size increases. The small scan traffic sample size results in a large degree of uncertainty in our large packet block distributions. Consequently, it's not possible to estimate false positive and negative rates for the 320KB results with reasonable accuracy. We observed that as normal Internet packet block size increases, the distribution narrows. It's reasonable to expect the scan traffic distribution to do the same. If that's the case, then for the data we tested, false positives and negatives become exceedingly small when the median packet block size is around 320KB. Near-zero errors — this is a significant improvement over the results reported by Lazarevic, et al. [28] for conventional IDSs detecting anomalous intrusions. Their best algorithm, using 23 metrics, yielded zero false negatives, when configured to allow 8% false positives. We're able to exceed these results with one metric. In order to verify this proposition, we continue to gather more malicious data sets.

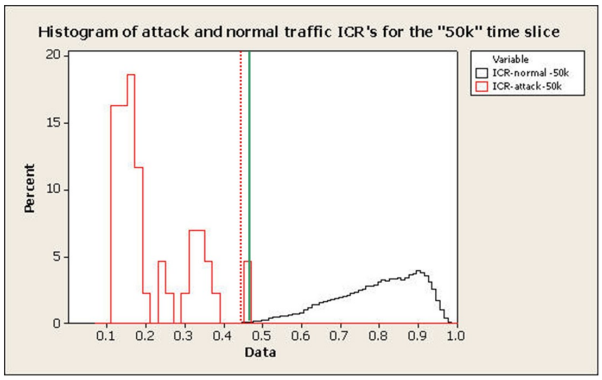
While increasing packet block size improved results in a laboratory setting, this may well not prove true in practice. We used clean scan traffic. Rarely will this be found outside a laboratory setting. In an operational environment, malicious and normal traffic are mixed.

12. Algorithm use in Operational Security

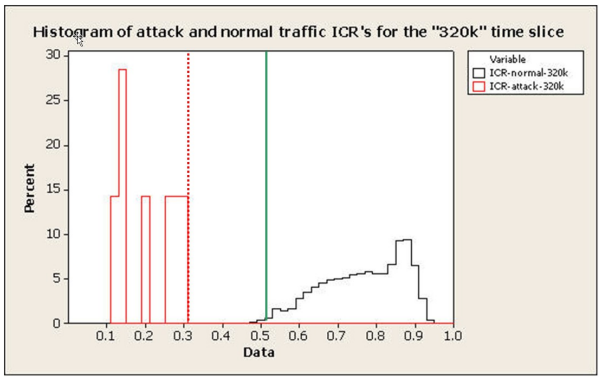
While "clean room" studies provide important insights, rarely can lessons learned be applied directly in real-world settings. This is certainly true in anomalous intrusion detection. Malicious exploits are just a small segment of a sea of legitimate activity: statistically, may just be an insignificant outlier. However, operationally, they can be catastrophic. The challenge is finding detection processes that are minimally impacted by innocuous activity. Practitioners in the field want and generally need, a near-zero error rate.

There are two factors that influence this algorithm's mix of false positives and negatives and overall error rate.

- Packet block size — As can be seen in Figure 8, as packet block size increases, the distributions narrow. When the packet blocks are pure — all normal or all scan packets, this means more separation between traffic types. The result is less misclassification. Unfortunately, such separation is rarely the case in an operational environment.



(a) The separation between 99% CI for the 50KB scan and normal traffic packet blocks is around .02 ICRs. We would expect the failure rate to be well below those reported for conventional anomalous intrusion detectors.



(b) The separation between 99% CI for the 320KB scan and normal traffic packet blocks is around .20 ICRs. This appears to make the chance of false positives at least 5σ. The chance of false negatives may be even less.

Figure 8. Probability distributions for 50KB and 320KB scan and normal traffic packet blocks are compared in this figure. In contrast to the distributions shown in Figure 6, the distributions don't overlap. As packet block size increases, the distributions tighten. This results in a much improved detection accuracy in our lab conditions. As discussed in Section 12, these results may be difficult to attain in real-world settings.

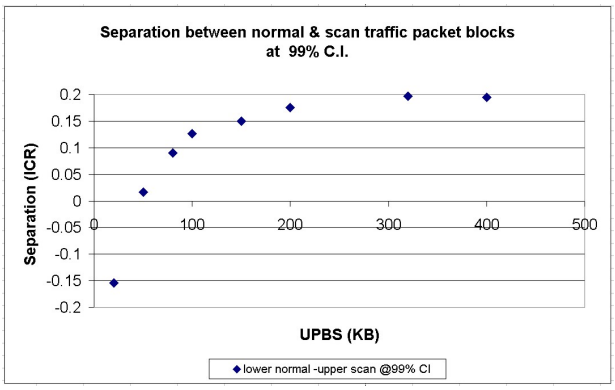


Figure 7. The separation between the ICR distribution for normal Internet traffic and scan traffic increases as packet block size increases, asymptotically approaching a maximum around 0.2 ICRs. For small packet block sizes, the confidence intervals actually overlap (negative separation). The separation goes positive around the 50KB packet block size. There appears to be little benefit to using a packet block larger than 250KB. Depending upon the acceptable error rate, a smaller packet block size may be usable.

Malicious and normal traffic are usually mixed. (Sophisticated attackers will, in fact, attempt to keep their illicit traffic "below the radar" by keeping their packet rate so low that they are undetected. See the Appendix) Packet blocks consisting of mixed traffic will have ICRs between the bounds for pure normal and pure scan traffic. As packet block size increases, scan packets are more likely to be mixed with other packet types and go undetected (resulting in a false negative). Small packet block sizes will be more sensitive to scan traffic, but at a cost. Both traffic distributions will spread, increasing the likelihood of misclassification.

- Threshold — For any given packet block size, a threshold will be set. Any packet block with an ICR below the threshold will be labeled as containing scan traffic. As the threshold moves away from the normal traffic median and approaches the scan traffic median, the risk of false positives decreases. The risk of false negatives increases, however.

A sensitivity analysis would help quantify the trade offs involved. The best approach may be to use packet block ICR as a prescreening metric. Set the ICR threshold sufficiently high to reduce the false negative rate to an acceptable level, then apply another IDS to suspect packet blocks.

Another issue is cost-effectiveness. The time slice used to gather 320KB packet blocks of Internet traf-

fic at the University perimeter varied with the sample, but ranged around one second. The Dell GX240 used for compressing the packet blocks could process one packet block in around ten seconds. Even with this relatively modest traffic rate, in order to use Win_RK, an enterprise would need a distributed processing array involving a considerable equipment investment. Many other compression algorithms are considerably faster. A review of processing speeds reported using the Canterbury Corpus [36] shows that the "ZIP" family of compressors could come close to keeping up with a 320KB/sec data rate on our test equipment. The concern is, can a lower efficiency compressor such as ZIP maintain an acceptable separation between scan and normal Internet traffic? This issue is deferred for future examination.

13. Additional Applications

While this study focused on Internet traffic at a network gateway, potential applications appear to be wide-ranging:

- Using a slightly modified process, Internet Service Providers (ISPs) at all levels — local to backbone — may be able to detect network denials of service (NDoS) at an early stage, allowing ISPs to take preventative action sooner. This could slow down virus spread, for example, giving sysadmins more time to activate defenses. Matrawy, et al. [33] points out that NDoS's can have numerous causes. Our metric should be equally effective in detecting a wide range of NDoS-causing traffic.

- Intranet sysadmins might more readily detect insider attacks by:

  - using this method at internal router and / or device network interface cards (NIC's). In addition to addressing insider attacks, this would also provide a second layer of detection for external attacks.

  - tracking bus traffic within a device looking for low complexity traffic bursts. This could, in some instances, detect malicious traffic coming from external sources such as removable media, keyboards and NICS, as well as something like a time bomb placed on a hard drive. Such a tracker could be implemented with software. It could, however, be bypassed by an attacker using a boot disk. An attacker would have a much more difficulty bypassing a hardware or firmware implementation.

- Compromised system devices acting as zombies could potentially be discovered by testing outgoing network traffic.

Just because something is possible, however, does not mean it should be done. Adding IDS functionality at any level will increase system overhead. In order to maintain acceptable system utility to the user, the security level should be balanced against the system performance. Due to the wide range of user needs and expectations, this will by necessity, occur on a case-by-case basis.

14. Future work

As with all research, one answer generates multiple questions:

- "In the wild", scanning packets will often be diluted by normal traffic. How much does this impact the ability of relative distance to detect scans?

- How well does this process work on other attack types, including a mixture of attacks?

- How can this method be combined with other intrusion detection methods to reduce detection errors and improve performance? We expect that when used in conjunction with other well-established metrics, ICR will reduce errors considerably. Since scans generally test for many open ports, scan packets will most likely be included in more than one packet block. The false positive rate can be reduced greatly by not sending an alarm on one suspicious packet block.

- The computational overhead of calculating relative distance may not be necessary. If ICR is substituted, how much does that increase the error rate? The flatness of the relative distance slopes, seen in Figure 5, suggests that over the relatively small range of packet block sizes produced by our algorithm, calculating relative distance may not be necessary. Testing ICR against a boundary may be sufficient.

- Compared to other well established compression algorithms, Win_RK is slow. How well does ICR detect scans with higher performance, but less efficient compression algorithms?

- We concatenate packets in arrival order when making blocks. Could another packet block assembly procedure yield better results?

- Evans observed, in one of his reports [15], that the order in which substrings were placed affected compression efficiency. How much does that impact detection accuracy?

- Bush and Evans [5] also discovered that in the case of FTP attacks, eliminating some fields before testing improved detection. Can we improve our results here in a like manner?

- An enterprises' Internet packet mix can vary greatly over time (workday vs. weekend, for instance). This experiment used University Internet traffic. The efficacy of this detection method needs to be evaluated using normal (non-malicious)Internet traffic generated by an assortment of different enterprise types, analyzed as a function of time.

We will be conducting further experiments to address these open questions.

15. Conclusion

The usefulness of information-theory-based metrics for intrusion detection is reinforced by this work. In particular, packet blocks containing scan traffic are shown to have a radically lower ICR than packet blocks containing normal Internet traffic. In a laboratory setting, we discover that as packet block size increases, errors in classifying the two distributions (scan and normal traffic) dropped. Ultimately, we show that the two groups can be distinguished with a vanishingly small risk of false positives or negatives.

16. Acknowledgements

The authors thank red team members and Scholarship for Service recipients Scott Miller and Willie Kwan for their cooperation in generating and capturing scan traffic critical to the success of this study.

17. Appendix: Low Frequency Attack Detection

Sophisticated intruders may well try to avoid detection by diluting their traffic in legitimate system activity. This would seem to be an effective detection avoidance technique. A lower attack frequency definitely impacts IDS design and performance. For example:

- For pattern matching, patterns must be compared over greater time spans or greater packet volumes. This means a greater commitment of system assets to intrusion detection.

- For statistically based algorithms, smaller deviations from normal traffic or other system and user statistical profiles must be tolerated. Lower tolerances mean an increased false positive rate. In order to process the additional alarms, more system assets and/or sysadmin time will be required to track IDS errors.

This could become an endless cycle; attackers lower their attack rate to avoid detection, so more system assets are committed to intrusion detection, causing attackers to lower their rate more and therefore causing defenders to increase system assets used for intrusion detection and on ad infinitum. Taken to the limit, IDS's must be able to reliably identify single malicious events. In special cases, this may be possible. Two counter examples move us to conclude that single event intrusion detection is impossible in the general case. A single ping and a single failed login attempt are such events. Both occurrences are, or can be, the result of everyday, legitimate activity. Detecting a single event in an anomalous intrusion is even less likely; their very novelty already makes detection difficult.

The metric we present herein appears to improve suspicious traffic partitioning. To the extent this occurs, intrusion detection should improve. Our metric does not, however, impact the basic low frequency attack issue. At best, we could hope that relative distance or a related metric would tip the balance in the defenders favor. If sufficient system assets can be affordably committed to intrusion detection, then the detectable attack frequency will become so low that all but the most dedicated attackers will abandon that avoidance technique as too time-consuming.

References

[1] V. Barnett and T. Lewis. *Outliers in Statistical Data*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Chinchester, West Sussex, England, third edition, 2000.

[2] C. H. Bennett, P. G'acs, M. Li, P. M. B. Vitanyi, and W. H. Zuerk. Information distance. *IEEE Transactions on Information Theory*, 44(4):1407–1423, 1998.

[3] W. Bergmans. Maximum compression. http://www.maximumcompression.com, October 27, 2005. Eindhoven, Netherlands.

[4] P. Bogaerts. Hping tutorial. http://www.radarhack.com/dir/papers/hping2_v1.5.pdfl, November 26, 2005.

[5] S. F. Bush and S. C. Evans. Information assurance design and assessment. Final report, General Electric Research and Development Center, August 2001.

[6] G. Casela and R. L. Berger. *Statistical Inference*. Duxbury Press, Pacific Grove, Cal., second edition, 2001.

[7] B. Cashell, W. D. Jackson, M. Jickling, and B. Webel. The economic impact of cyber-attacks. CRS Report for Congress RL32331, Congressional Research Service The Library of Congress, April 2004.

[8] S. Chebrolu, A. Abraham, and J. P. Thomas. Feature deduction and ensemble design of intrusion detection systems. *Computers & Security*, 24:295–307, June 2005.

[9] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley and Sons, Inc., New York, 1991.

[10] D. Curry. Intrusion detection systems. http://www.cerias.purdue.edu/about/history/coast_resources/idcontent/ids.html, February 2, 2006. the Center for Education and Research in Information Assurance (CERIAS), Purdue University, West Lafayette, Indiana.

[11] D. E. Denning. *Information Warfare and Security*. Addison-Wesley, 2001.

[12] S. C. Evans. *Kolmogorov Complexity Estimation and Application for Information System Security*. PhD thesis, Rensselaer Polytechnic Institute, July 2003.

[13] S. C. Evans and B. Barnett. Network security though conservation of complexity. In *Proceedings of MILCOM*, The Disneyland Resort, Anaheim, California, October 2003.

[14] S. C. Evans, S. F. Bush, and J. Hershey. Information assurance through kolmogorov complexity. In *Proceedings of DARPA Information Survivability Conference and Exposition II*, pages 322–331, Anaheim, California, June 2001.

[15] S. C. Evans and J. Hershey. A two-stage complexity estimator. Technical Information Series 2002GRC197, General Electric Research and Development Center, August 200.

[16] fyodor. Port scanning techniques. http://www.insecure.org/nmap/man/man-port-scanning-techniques.html, November 26, 2005.

[17] J. Gilchrist. Archive compression test. http://www.compression.ca/act, August 30, 2005. the Carleton University, 1125 Colonel By Drive, Ottowa, Ontario, K1S5B6, Canada.

[18] S. Goel and S. F. Bush. Kolmogorov complexity estimates for detection of viruses in biologically inspired security systems: A comparison with traditional approaches. *Complexity*, 9(2):54–73, 2003.

[19] E. Jonckheere, K. Shah, and S. Bohacek. Dynamic modeling of internet traffic for intrusion detection. In *Proceedings of IEEE American Controls Conference*, Anchorage, Alaska, July 2001.

[20] A. K. Jones and R. Sielken. Computer system intrusion detection: A survey. Technical report, University of Virgina, 2000.

[21] E. Jonsson and T. Olovsson. A quantitative model of the security intrusion process based on attacker bahavior. *IEEE Transactions on Software Engineering*, 23:235–245, 1997.

[22] JWS. Tcpdump public repository. http://www.tcpdump.org/, August 30, 2005.

[23] W. Kirchherr, M. Li, and P. Vitanyi. The miraculous universal distribution. *The Mathematical Intelligencer*, 19(4):7–15, 1997.

[24] A. N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems of Information Transmission*, 1:4–7, 1965.

[25] A. N. Kolmogorov. Logical basis for information theory and probability theory. *IEEE Transactions on Information Theory*, IT-14:662–664, 1968.

[26] V. Kreinovich, S. F. Luc Longpre, and L. Ginzburg. Why is selecting the simplest hypothesis (consistent with data) a good idea? a simple explanation. *Bulletin of the European Association for Theoretical Computer Science (EATCS)*, 77:191–794, 2002.

[27] A. Kulkarni, S. F. Bush, and S. C. Evans. Detecting distributed denial of service attacks using kolmogorov complexity. Technical Information Series 2001crd176, General Electric Research and Development Center, December 2001.

[28] A. Lazarevic, L. Ertoz, A. Ozgur, J. Srivastava, and V. Kumar. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of Third SIAM Conference on Data Mining*, San Francisco, California, May 2003.

[29] W. Lee and D. Xiang. Information-theoretic measures for anomaly detection. In *Proceedings of IEEE Symposium on Security & Privacy*, 2001.

[30] P. Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Scienceof India*, 2:49–55, 1936.

[31] M. Mahoney. The paq data compression programs. http://www.cs.fit.edu/ mmahoney/compression/, October 27, 2005. the Florida Institute of Technology, 150 W. University Blvd. Melbourne, FL 32901, USA.

[32] M. V. Mahoney. *A Machine Learning Approach to Detecting Attacks by Identifying Anomalies in Network traffic*. PhD thesis, Florida Institute of Technology, May 2003.

[33] A. Matrawy, P. van Oorschot, and A. Somayaji. Mitigating network denial of service through diversity-based traffic management. In *Proceedings of Applied Cryptography and Network Security: Third International Conference (ACNS 2005)*, number LNCS 3531, pages 104–121, New York, New York, June 7-10 2005. Springer.

[34] M. Nelson. Benchmarks. http://www.data-compression.info/Benchmarks.shtml, October 27, 2005. the Visicron Corp., 3 Bethesda Metro Center, Suite #700, Bethesda, Maryland, 20814, USA.

[35] R. W. Potter. *The Art of Measurement Theory and Practice*. Hewlett-Packard Professional Books. Prentice Hall PTR, Upper SaddleRiver, New Jersey, 2000.

[36] M. Powell. Canterbury corpus. http://www.corpus.canterbury.ac.nz, October 27, 2005. the Department of Computer Science and Software Engineering, University of Canterbury, Christchurch, New Zealand.

[37] R. L. Schaeffer. *Introduction to Probability and Its Applications*. Duxbury Press, Belmont, California, second edition, 1995.

[38] A. H. Sung and S. Mukkamala. Identifying important features for intrusion detection using support vector machines and neural networks. In *Proceedings of International Symposium on Applications and the Internet (SAINT 2003)*, volume 17, page 209, 2003.

[39] K. Wang and S. J. Stolfo. Anomalous payload-based network intrusion detection. In *Proceedings of RAID'04*, Sophia Antipolis, French Riviera, France, September 15-17 2004.