Evaluation of Statistical Tests for Detecting Storage-Based Covert Channels

Thomas Sattolo, Jason Jaskolka

Evaluation of Statistical Tests for Detecting Storage-Based Covert Channels*

Thomas A.V. Sattolo[0000−0002−8799−0507] and Jason Jaskolka[0000−0001−6316−3040]

Systems and Computer Engineering
Carleton University, Ottawa, ON K1S 5B6, Canada
{thomas.sattolo, jason.jaskolka}@carleton.ca

Abstract. Individuals and organizations are more aware than ever of the importance and value of preserving the confidentiality and privacy of sensitive information. However, detecting the leakage of sensitive information in networked systems is still a challenging problem, especially when adversaries use covert channels to exfiltrate sensitive information to unauthorized parties. Presently, approaches for detecting timing-based covert channels have been studied more extensively than those for detecting storage-based covert channels. In this paper, we evaluate the effectiveness of a selection of statistical tests for detecting storage-based covert channels. We present the results of several experiments which show that complexity-based tests are effective at detecting storage-based covert channels when information is embedded into network packet header fields that are not expected to follow a particular pattern, such as the IP Identification and Time-to-Live. These results can help to guide the construction of practical detection platforms capable of effectively detecting the leakage of sensitive information via storage-based covert channels.

Keywords: Covert Channel · Detection · Statistical Tests · Storage

1 Introduction and Motivation

Data breaches are an ever-growing problem and detecting them is both challenging and valuable as the average total cost of a data breach is now $3.68 million [21]. One of the ways an adversary may seek to avoid detection is by using a covert channel to exfiltrate data from the network.

A covert channel is a means of communication that purports to be difficult for a third-party observer to detect [15]. In the field of information technology, a variety of covert channels have been devised to enable communication between computers without alerting the network owner. Covert channels can be divided into two major categories: timing-based channels and storage-based channels. Timing-based covert channels operate by altering the timing of otherwise legitimate network traffic so that the arrival times of packets encode secret

* This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2019-06306.

information. For example, sending two packets in quick succession could represent bit '0' and a long gap between subsequent packets could represent bit '1'. Storage-based covert channels, on the other hand, operate by hiding data in unused fields of RFC-defined network protocols like IP and TCP.

At present, the literature on detecting timing-based covert channels is much more extensive than that on detecting storage-based covert channels. This paper aims to narrow this gap by taking several techniques that have been used successfully to detect timing-based channels, and evaluating their effectiveness for detecting storage-based channels. This is part of an overarching goal to develop methods to detect storage-based covert channels in real-time by observing network traffic. This will necessarily involve computationally efficient statistical tests to distinguish covert channels from ordinary network traffic.

The remainder of this paper is organized as follows. Section 2 provides a short overview of related work. This is followed by a categorization of the tests to be studied in Section 3. Next, the experimental methodology is detailed in Section 4 and the results are reported in Section 5. These results are discussed in Section 6 and, lastly, Section 7 concludes and highlights future work.

2 Related Work

When identifying the existence of covert channels in computer systems, particularly those which use network protocols as covert message carriers, anomaly detection has been among the most popular detection mechanisms. Anomaly detection refers to the detection of patterns in a given data set that do not conform to what is considered normal behaviour. Anomaly detection techniques are usually used in conjunction with machine learning and statistical approaches.

As previously indicated, much of the existing work on detecting covert channels has focused on timing-based covert channels. Cabuk et al. [3,4] and Gianvecchio and Wang [8] proposed a variety of statistical tests focussed on detecting timing-based covert channels. Naik et al. [20] proposed an entropy-based approach for detecting timing-based channels. Similarly, Li et al. [19] proposed yet more tests for detecting timing-based channels and combined them into a random forest classifier. Crespi et al. [6] studied different statistical anomaly detection methods, commonly used in network traffic analysis, to detect timing-based covert channels.

While there has been a focus on timing-based channels, approaches for storage-based covert channels have also been proposed. Sohn et al. [23] proposed an offline covert channel detection technique using a support vector machine to search for anomalies in the header fields of network packets. A similar technique was proposed by Tumoian and Anikeev [24] involving the interception of all TCP traffic and a model of the initial sequence number generation. The idea uses a neural network to create a model using only the intercepted TCP traffic without any knowledge of the data generation algorithm in an attempt to identify anomalies in the initial sequence numbers of the intercepted TCP segments. Berk et al. [1,2] investigated a methodology for detecting such channels based on a sta-

tistical measure of how well the capacity of a given channel is achieved. Jadhav and Kattimani [13] proposed a statistical detection method involving the capture of TCP segments from active network streams and analyzing the covert channel vulnerable fields of TCP headers. Zhai et al. [25] proposed a method based on a TCP Markov model and the Kullback-Leibler divergence [17] to verify the existence of anomalies in the TCP Flags field. Zhao and Shi [26] aimed to detect the existence of covert information embedded in TCP Initial Sequence Numbers using phase-space reconstruction to represent the dynamic nature of initial sequence numbers by building a four-dimensional space of the one dimensional initial sequence numbers.

Besides these, Gunadi and Zander [9–12] built an entire covert channel detection system covering aspects of both timing-based and storage-based channels. That said, their system considers far fewer tests than what we describe in this paper and it does not focus on analyzing which tests are most effective for detecting storage-based covert channels.

3 Test Classification

Statistical tests previously used to detect timing-based covert channels can be classified into two general categories: *Complexity Tests* and *Distributional Tests*.

3.1 Complexity Tests

Complexity tests attempt to measure the extent to which a stream of data is random or predictable and they work if the information transmitted via a covert channel is systematically different from normal network traffic in this way. This works with timing-based covert channels [4,8,12], and the concept should extend to storage-based covert channels, with some caveats. Firstly, it is not necessarily the case that traffic becomes less random when used as a covert channel[1]. This ought to be the case with some fields of network packet headers such as TCP Sequence numbers and IP Identification numbers that can be expected to be random under normal circumstances, but not with others such as IP Flags that would usually just be the same for most packets. The latter, of course, become more random once they have covert information stored in them. It should be possible to determine thresholds—ones that are different for each field—that will separate normal traffic from that when a covert channel is being used.

There are many ways to measure the complexity of a stream of data and thus there are many potential complexity tests to consider. Nonetheless, they can be divided into two types: compressibility-based tests and entropy-based tests.

The compressibility of a stream of data can be used as measure of its randomness—more compressible data is less random. This randomness, in turn,

[1] This is the case with timing-based covert channels because normal inter-packet delays are essentially random and those of covert channels cluster at either of the values used as symbols in the communication.

is informative as to the likelihood that the stream is being used as a covert channel. This technique was used by Cabuk et al. [4] to detect timing-based covert channels. They examined a covert channel created using the arrival times of IP packets; messages were sent by alternating long and short delays between packets to represent binary digits. They found that the arrival time pattern of this traffic could be separated from normal traffic because it was more compressible than normal, unaltered traffic.

Entropy is a measure of the information contained in data. As such, it is the most direct tool to search for covert information. Both unused network protocol fields and the covert messages that people may send contain some information, but there is no particular reason for them to contain the same amount of information per byte sent. If these happen to differ, measuring the entropy of unused network protocol fields has the potential to reveal covert channels if they exist.

All of the complexity tests evaluated in our experiments are ultimately based on either compressibility or entropy.

3.2 Distributional Tests

Much of the work of statistical science is to quantify the extent to which two or more entities are different in a world of limited observations. Fundamentally, this is the same as the task of identifying a covert channel so it stands to reason that the huge body of work that exists in statistics would be useful for detecting covert channels. In the literature, several different statistical features—ones that have nothing to do with entropy or compressibility—of inter-packet delay data have been used successfully to detect timing-based covert channels [3, 12, 19].

However, there is reason to believe that the usefulness of these tests will not transfer to storage-based covert channels. Underlying the logic of these distributional tests is the assumption that the data provided to them is measuring something; they involve ordinal comparisons between the elements in putative covert channels. This makes sense for timing-based covert channels where each element is an inter-packet delay, measured in seconds, and ordering inter-packet delays is obviously meaningful. The same cannot be said of storage-based covert channels, where each element is just the data observed in a network protocol header field. These are not measuring anything, are dimensionless, and ordering them in any way is spurious.

In spite of this, the applicability of distributional tests to storage-based covert channels cannot be so easily dismissed. Distributional tests seem less likely than complexity tests to transfer to storage-based covert channels, but abstract reasoning is no match for experiment, so a range of distributional tests will be considered and evaluated in our study.

4 Experimental Setup and Approach

4.1 Tests Included in the Experiments

Below is a brief overview and rationale for including the statistical tests that are considered in our experiments.

Complexity Tests

LZMA Compression (LZMA): The Lempel-Ziv Markov Chain compression algorithm is a widely used algorithm that offers a high compression ratio at the price of relatively low compression speed [5]. The compression ratio is what is reported as the output of the test. As with all the compression algorithms among the tests considered in our experiments, the rationale for studying LZMA to detect storage-based covert channels comes from [4].

LZ77 Compression (LZ77): Since the overall performance (compression ratio) of the compression is not directly relevant to the purpose of our experiments—all that matters is the difference between how it compresses covert channel traffic versus innocent traffic—it is worthwhile to test a simpler compression algorithm such as Lempel-Ziv 77 [27].

LZ78 Compression (LZ78): Lempel and Ziv also proposed another simple compression algorithm known as Lempel-Ziv 78 [28]. For the same reasons to consider LZ77 in our study, we also consider LZ78.

Lempel-Ziv Complexity (LZC): As precursor to their publications on practical compression algorithms, Lempel and Ziv studied a complexity measure for strings related to the number of distinct substrings and their abundance within the original string [18]. Seeing as the compression ratio given by LZ77 or LZ78 is expected to be a useful test, is only natural to also include a related idea that is meant specifically to be a complexity measure.

First-Order Entropy (FOE): Entropy is generally calculated based on a probability mass function. From any given sequence of values one can compute a histogram (i.e., count the occurrences of every unique value); this histogram can then be taken as an approximation of the probability mass function for the process generating the sequence and one can compute the resulting entropy. The result of approximating entropy in this way is termed first-order entropy and was used to detect timing-based covert channels in [9, 19].

Corrected Conditional Entropy (CCE): The entropy rate of a sequence is only truly defined for sequences of infinite length. Of course, this means that the entropy rate of any empirically obtained sequence cannot be computed so it must be estimated. Corrected Conditional Entropy is one method to produce such an estimate [22]. It was first imported into the field of covert channels by Gianvecchio and Wang [8].

Repetition (REP): This test simply counts the fraction of elements in a trace that are unique. This test is considered not so much a practical test, but rather as a "sanity check" that the other tests considered in our study are not performing worse than such a simple and easily computable measure.

Distributional Tests

Autocovariance: This is a measure of how similar a sequence is to itself at other points in the series. It was used to detect timing-based covert channels by Gunadi and Zander [9], as well as Li et al. [19].

Kolmogorov-Smirnov Test: This is a statistical test used determine whether two empirical cumulative distribution functions were sampled from the same source. It has also been used in [9, 19] to detect timing-based covert channels.

Wilcoxon Signed Rank: This is a statistical test that compares the rank of two sequences to assess their similarity. It was used by Li et al. [19] for timing-based covert channel detection.

Spearman Correlation: The correlation between the rank of two sequences can be used a measure of the similarity of these two sequences. This is known as Spearman Correlation. It has been used to detect timing-based channels in [19].

Regularity: This is a metric proposed by Cabuk et al. [3] specifically to detect timing-based covert channels. It effectively measures how much the standard deviation of a sequence changes over time.

4.2 Building the Dataset

To build the dataset for our experiments, we needed samples of network traffic with and without information covertly embedded into them. Samples of real normal network traffic were acquired from the Malware Capture Facility Project at the Czech Technical University in Prague [7].

To generate network traffic with covert information, we chose to build a storage-based covert channel embedding bits into the Identification (ID) field of the IP header. The purpose of this field is to identify packets that have been fragmented when the fragments are to be reconstructed. But today, networks tend to have large enough maximum transmissible units that IP packets do not need to be fragmented and these packets are usually sent with IP Flags set to "Don't fragment." Because of this, the ID field is rarely payed any attention, making it a good place to inject covert information. Furthermore, the field is included in every IP packet and is not expected to follow any particular pattern (unlike TCP Sequence numbers), so new covert information can be included in every packet sent. The field is 16 bits long, so a covert channel could transmit up to two bytes per packet.

For our experiments, a covert message had to be selected. The covert message ought to be something that any test would view as similar to real human communication. Text from a novel is appropriate for this purpose, and we chose to use the first few paragraphs of *Pride and Prejudice* by Jane Austen.

To create test data, the IP ID fields were extracted from each of the IP headers. Bits from the message were embedded into the these IP ID fields by replacing the least significant bits of the field with bits from the message[2]. The number of bits replaced was varied between 1 and 16, i.e., between changing just one bit of the field, and replacing them completely. Henceforth, the sequence of IP ID values extracted from the network before the message is embedded will be referred to as the *carrier*; once these carriers have message bits embedded into them they will be known as *traces*. Each 16-bit IP ID value (with or without

[2] A similar process can be adopted for other header fields of network packets.

message bits embedded) that form the traces/carrier will be known as an *element*. Note that to create the traces, the least significant bits of the carriers were replaced completely (i.e., AND-ed with zeroes and then OR-ed with the message bits). Using an exclusive-or operation to embed information does not work in this case because the message recipient does not know the value of the field in the carrier and so cannot recover the message if it is embedded via exclusive-or. For example, to embed the first two bits of the word "The" into the IP ID value 0x154A8FE0E, we set the two least significant bits to zero to get 0x154A8FE0C and OR that with '01'—the first two bits of 0x54, the value of 'T' in ASCII—to get 0x154A8FE0D. The result of this process is a set of 16 different traces, one for each number of bits. To ensure that comparisons between them are valid, all 16 traces are of the same length and they all contain the entire message. This requires that all traces except the 1-bit trace contain IP ID fields with no message content.

4.3 Conducting the Experiments

The code for generating the datasets and conducting our experiments is available at: `gitlab.com/CyberSEA-Public/CCStatTests`. The experiments are done in iterations. On each iteration the IP ID field is extracted from a certain number of different packets creating a carrier. The message is then embedded, creating 16 traces and the tests are applied to each trace. The result is one value per test per trace. The test is also applied to the carrier independently of the tests on the traces.

In the next iteration each of these tests is repeated. In total, 1000 iterations are performed and we calculate the mean and standard deviation for each trace/carrier. This whole process is repeated 3 times so as to vary the size of the message. Tests with 256, 16 and 1 byte messages are performed.

In addition to the repeated iterations and different message sizes, the process was repeated with the message encrypted. The Rijndael cipher of the Advanced Encryption Standard (AES) is the current state-of-the-art, so this is what we used. However, AES uses a block cipher with 16-byte blocks such that the smallest message that can be encrypted is 16 bytes long[3]. Consequently a stream cipher that can encrypt a single byte was used for the 1-byte messages, namely the Salsa20 cipher. This limitation is the reason for using 16 bytes as the second smallest message size in our experiments.

5 Experimental Results

5.1 Results for Tests Used in Isolation

The results of each experiment are presented here as a series of tables. First Tables 1–3 present the effect size for every trace for each message size for each test in our experiments. More specifically, they show the difference between the mean of the test's output on the traces and the mean output on the carrier.

[3] Padding the message would complicate interpretation of the results.

This difference is presented in units of the joint standard deviation of the carrier and trace outputs (i.e., the square root of the mean of the two variances). This is known as the *effect size* and it measures how clearly the test can distinguish each trace from the carrier.

The first thing to note is that complexity tests perform much better than distributional tests. No distributional test ever produced an effect size greater than 0.00189 (see Table 1) and, excluding the 1-byte messages (Table 3), all the complexity tests results are at least 0.445 (Table 1)—a difference of more than 2 orders of magnitude. If we go further and exclude Corrected Conditional Entropy on 256-byte messages the weakest results for a complexity test is much larger still at 2.58 (Table 2). As for the 1-byte messages, several test results were undefined (denoted by $\perp$). This means that the difference in the mean and joint standard deviations were both zero, i.e., the test result was the same for every iteration for both the carrier and at least one trace. In context, this means the test fails to differentiate the traces from the carrier, so $\perp$ is roughly equivalent to 0. That this is the result for many of the complexity tests means that a 1-byte message is not sufficient for them to usefully detect covert channels; even the tests that avoid this issue do not produce large effects.

Next, it is interesting to note that 1-bit traces are generally the least different from the carrier. The only exceptions to this, besides the distributional tests and the 1-byte message table where the tests are not effective, are some of the LZ77 results. Moreover, encryption made little difference and, counterintuitively, the tests mostly did better when the message was encrypted. This improvement is nonetheless quite small; LZ77 was affected the most and even it has a large effect everywhere (except on 1-byte messages) regardless of encryption.

One of the most interesting things about the result is that Repetition outperformed both First-Order Entropy and Corrected Conditional Entropy. This is odd because these both rely heavily on counting unique elements. The main difference is that entropy does so in a way that properly reflects the information contained in the traces and takes into account elements that are repeated more than once, whereas Repetition is ad hoc and should not be expected to be very meaningful. Nevertheless, Repetition performs better and is certainly more efficient than either of the entropy-based tests: the computations required for Repetition are a strict subset of those required for First-Order Entropy which are themselves a strict subset of those for Corrected Conditional Entropy.

As for the compression-based tests and Lempel-Ziv Complexity, all of them perform similarly and on par with Repetition on 256-byte messages. On 16-byte messages, LZ77 is notably worse and LZ78 is notably better, but all are outperformed by Repetition. This analysis tentatively pinpoints Repetition as the best test overall with LZ78 close behind, but there is no clear winner.

5.2 Results for Tests Used in Combination

Tests need not be used in isolation of each other and it is of interest to determine how they perform together. To do this, we show the correlation between the results of the tests across iterations in Tables 4 and 5. Because, in the previous

Table 1. Effect Sizes for 256-byte messages in an IP ID covert channel

	1-bit Trace		Minimum		Min. Index	
Encrypted?	Y	N	Y	N	Y	N
LZMA Compression	2.95	3	2.95	3	1	1
LZ77 Compression	6.3	4.62	3.59	3.55	3	2
LZ78 Compression	3.56	3.25	3.56	3.25	1	1
Lempel-Ziv Complexity	5.04	4.28	5.04	4.28	1	1
First-Order Entropy	3.37	3.13	3.37	3.13	1	1
Corr. Cond. Entropy	0.445	0.449	0.445	0.449	1	1
Repetition	4.58	3.94	4.58	3.94	1	1
Autocovariance	3.08e-08	9.12e-07	3.08e-08	4.23e-07	1	3
Kolm.-Smirnov Test	0.00189	0.00184	0.00189	0.00184	1	1
Wilcoxon Signed Rank	1.26e-05	2.53e-05	1.26e-05	2.53e-05	1	1
Spearman Correlation	3.46e-06	1.31e-05	3.46e-06	1.31e-05	1	1
Regularity	8.77e-07	3.82e-06	8.77e-07	3.82e-06	1	1

Table 2. Effect Sizes for 16-byte messages in an IP ID covert channel

	1-bit Trace		Minimum		Min. Index	
Encrypted?	Y	N	Y	N	Y	N
LZMA Compression	4.87	4.76	4.87	4.76	1	1
LZ77 Compression	3.24	3.15	2.58	3.15	2	1
LZ78 Compression	6.9	5.98	6.9	5.98	1	1
Lempel-Ziv Complexity	4.25	4.05	4.25	4.05	1	1
First-Order Entropy	6.24	5.55	6.24	5.55	1	1
Corr. Cond. Entropy	2.64	2.63	2.64	2.63	1	1
Repetition	8.25	6.7	8.25	6.7	1	1
Autocovariance	6.88e-07	1.19e-06	6.88e-07	1.19e-06	1	1
Kolm.-Smirnov Test	0.000624	0.00125	0.000624	0.00125	1	1
Wilcoxon Signed Rank	3.95e-06	8.82e-05	3.95e-06	8.82e-05	1	1
Spearman Correlation	0.00109	0.000826	0.00109	0.000826	1	1
Regularity	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$

Table 3. Effect Sizes for 1-byte messages in an IP ID covert channel

	1-bit Trace		Minimum		Min. Index	
Encrypted?	Y	N	Y	N	Y	N
LZMA Compression	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
LZ77 Compression	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
LZ78 Compression	0.135	0.165	0.135	0.165	1	1
Lempel-Ziv Complexity	0.0164	0.0106	0.0164	0.0106	1	1
First-Order Entropy	0.139	0.169	0.139	0.169	1	1
Corr. Cond. Entropy	0.139	0.171	0.139	0.171	1	1
Repetition	0.139	0.169	0.139	0.169	1	1
Autocovariance	3.32e-06	3.45e-07	6.66e-07	3.45e-07	2	1
Kolm.-Smirnov Test	0.00157	0	0	0	2	1
Wilcoxon Signed Rank	0	0	0	0	1	1
Spearman Correlation	0.000775	0.000996	9.73e-05	9.73e-05	7	7
Regularity	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$

Table 4. Correlation for 1-bit traces of 256-byte messages in an IP ID covert channel

	LZMA	LZ77	LZ78	LZC	FOE	CCE	REP
LZMA	1	0.25	0.23	0.19	0.18	0.18	0.16
LZ77	0.25	1	0.66	0.76	0.68	0.13	0.65
LZ78	0.23	0.66	1	0.87	0.94	0.15	0.94
LZC	0.19	0.76	0.87	1	0.92	0.17	0.94
FOE	0.18	0.68	0.94	0.92	1	0.14	0.99
CCE	0.18	0.13	0.15	0.17	0.14	1	0.14
REP	0.16	0.65	0.94	0.94	0.99	0.14	1

Table 5. Correlation for 1-bit traces of 16-byte messages in an IP ID covert channel

	LZMA	LZ77	LZ78	LZC	FOE	CCE	REP
LZMA	1	0.65	0.59	0.71	0.59	0.19	0.58
LZ77	0.65	1	0.50	0.71	0.51	0.064	0.51
LZ78	0.59	0.50	1	0.69	0.97	0.20	0.97
LZC	0.71	0.71	0.69	1	0.70	0.11	0.70
FOE	0.59	0.51	0.97	0.70	1	0.18	1
CCE	0.19	0.064	0.20	0.11	0.18	1	0.18
REP	0.58	0.51	0.97	0.70	1	0.18	1

section, the 1-bit traces were the hardest to detect, we restrict our analysis here to those and because encryption had no significant impact, we no longer keep it in our consideration. We also do not continue to analyze distributional tests and 1-byte messages, having concluded that a storage-based covert channel detector based on these ideas is not effective. To be clear, what is being compared is the correlation across all iterations in the difference between the output of the tests for 1-bit traces and the carrier.

Surprisingly, the compression-based tests are not clearly more correlated with each other than with the entropy-based ones. The general trend is that tests are moderately correlated with each other. There are, however, some definite outliers. First of all, First-Order Entropy and Repetition are very strongly correlated; this diminishes First-Order Entropy as a contender because Repetition outperforms it and gives very similar results from one iteration to the next. Secondly, Corrected Conditional Entropy and LZMA Compression have relatively weak correlation with the other tests. The weakness of these correlations suggest that two tests could be combined to make a more effective detector. In fact, except for First-Order Entropy and Repetition, any number of the tests could be combined because their correlations are not close to perfect.

5.3 Logistic Regression Detector

In this section, we evaluate the performance of covert channel detectors that use the tests evaluated herein. The algorithm used to create the detector is a simple logistic regression classifier. This technique takes in negative and positive examples and returns a model that estimates the probability that a certain sample is positive based on its Euclidean distance from a line—the number of

Table 6. Detector accuracy for IP ID covert channel for various message sizes

Message Size	256 bytes	64 bytes	16 bytes	4 bytes
LZMA	0.950 ± 0.0086	0.966 ± 0.0066	0.985 ± 0.0040	0.526 ± 0.038
LZ77	0.998 ± 0.0013	0.995 ± 0.0023	0.973 ± 0.0054	0.908 ± 0.0099
LZ78	0.976 ± 0.0056	0.992 ± 0.003	0.997 ± 0.0021	0.945 ± 0.0093
LZC	0.966 ± 0.11	0.995 ± 0.0024	0.969 ± 0.0061	0.74 ± 0.015
FOE	0.982 ± 0.0046	0.995 ± 0.0027	0.997 ± 0.0017	0.984 ± 0.012
CCE	0.996 ± 0.002	0.998 ± 0.0015	0.930 ± 0.0086	0.982 ± 0.0046
REP	0.984 ± 0.0043	0.995 ± 0.0024	0.999 ± 0.0013	0.953 ± 0.021
All	0.992 ± 0.0038	0.995 ± 0.0025	0.991 ± 0.0038	0.982 ± 0.0045

tests determines the dimensionality of the space in which the linear classifier exists. In this instance, the positive examples are the test results for the 1-bit traces and the negative examples are the test results for the carrier. The examples are split into a training set and a test set so that the model can be evaluated on examples it has not yet seen. 70% of examples are in the training set leaving 30% for the test set. There are 2000 examples in total; two—a negative and a positive—for each of the 1000 iterations. This yields a training set of 1400 examples and a test set of 600 examples; both are balanced (i.e., they contain roughly the same number of positive and negative examples). All this is repeated for four message sizes: 256, 64, 16, and 4 bytes. The creation of the classifier was repeated 1000 times, randomizing the examples that were included in the test set in order to average out any effect of how the dataset is split; both the mean and the standard deviation are reported.

The accuracy of the classifier for an IP ID covert channel is presented in Table 6. Each column represents the accuracy for a different message size while each row except for the last represents the accuracy of a detector that uses only one of the complexity tests. The bottom row (All) shows the accuracy of a detector using all the tests together. The first thing to note about the results is that a detector using just one of these tests works very well: most of the detectors are greater than 90% accurate even for messages as small as 4 bytes and some are greater than 95% accurate. For messages of 16 bytes or more, the accuracy climbs to over 99% in most cases. The second thing to note is that the detector using all the tests does not outperform those that use just one test. Given this, it seems that combining more than one of the tests to create a more effective detector just leads to unnecessary complexity in the detector design.

6 Discussion

Having performed this experiment on storage-based covert channels using the IP ID field it seemed natural to investigate covert channels using other fields such as TCP Initial Sequence Numbers (ISN). This was done to underwhelming results as shown in Table 7. Effect sizes for complexity tests were generally less than 0.01 and the resulting logistic regression detector was no better than chance. The likely reason for this is that TCP ISNs can be truly random—no vestigial need to be unique in order to reconstruct fragmented IP packets unlike

Table 7. Detector accuracy for TCP ISN covert channel for various message sizes

Message Bytes	256 bytes	64 bytes	16 bytes	4 bytes
LZMA	0.415 ± 0.061	0.412 ± 0.062	0.408 ± 0.064	0.407 ± 0.069
LZ77	0.415 ± 0.06	0.412 ± 0.063	0.410 ± 0.063	0.403 ± 0.064
LZ78	0.412 ± 0.063	0.408 ± 0.064	0.409 ± 0.064	0.412 ± 0.063
LZC	0.412 ± 0.063	0.411 ± 0.065	0.404 ± 0.075	0.392 ± 0.087
FOE	0.414 ± 0.061	0.411 ± 0.062	0.410 ± 0.065	0.413 ± 0.066
CCE	0.410 ± 0.065	0.400 ± 0.078	0.430 ± 0.098	0.417 ± 0.075
REP	0.414 ± 0.061	0.408 ± 0.068	0.412 ± 0.063	0.409 ± 0.064
All	0.415 ± 0.059	0.384 ± 0.071	0.403 ± 0.097	0.378 ± 0.089

Table 8. Detector accuracy for IP TTL covert channel for various message sizes

Message Size	256 bytes	64 bytes	16 bytes	4 bytes
LZMA	0.975 ± 0.0072	0.921 ± 0.015	0.897 ± 0.028	0.787 ± 0.014
LZ77	0.924 ± 0.0098	0.906 ± 0.01	0.921 ± 0.0092	0.919 ± 0.0096
LZ78	0.669 ± 0.18	0.636 ± 0.18	0.895 ± 0.014	0.854 ± 0.012
LZC	1 ± 0	1 ± 0.00097	0.986 ± 0.0065	0.936 ± 0.0082
FOE	0.562 ± 0.22	0.497 ± 0.14	0.776 ± 0.019	0.836 ± 0.014
CCE	0.998 ± 0.0017	0.976 ± 0.0062	0.897 ± 0.013	0.810 ± 0.019
REP	0.746 ± 0.24	1 ± 0.00067	0.997 ± 0.0019	0.955 ± 0.007
All	1 ± 0	1 ± 0.00087	0.988 ± 0.0039	0.942 ± 0.0088

IP ID. The small fragments of messages that we embed are also very close to random, so tests relying on information content cannot distinguish the two. This observation reveals something of a trade-off in how systems should choose their TCP ISNs. The conventional wisdom is that they should be truly random so as to make it difficult for an attacker to spoof a connection, but doing this creates an opportunity for a nearly undetectable covert channel (at least by the statistical tests considered in this paper). That said, this is not much of a trade-off because almost any system will have much greater exposure from spoofed connections than from covert channels.

Despite this, there is still a reason why an attacker might use an IP ID covert channel and thus why one might want to detect them. Most obviously there is bandwidth: machines send out one IP ID per IP packet but only one TCP ISN per TCP connection and, since one TCP connection generally comprises many packets, many more IP IDs are transmitted than TCP ISNs. For instance, the dataset used in this paper contains 76 times more IP IDs than TCP ISNs. Furthermore, using TCP ISN requires more access to the machine than IP IDs. With IP IDs an attacker can just change what IP IDs the machine sends and if the packet is never fragmented (the usual case) no one is likely to notice; with TCP ISN the attacker would not only have to change what is sent, but also what sequence number the machine expects to receive for subsequent segments.

This limited transferability does not mean that the statistical tests evaluated in this paper are specific to IP ID covert channels. IP Time-to-Live (TTL) is another network protocol field that can be used to build a covert channel, and was one of many studied by Gunadi and Zander [10]. Using the same methodology

as above, we trained a classifier to detect IP TTL covert channels. The results are presented in Table 8 and are similar to those for IP ID covert channels. Note that for IP TTL it was the 8-bit traces that were hardest to classify (i.e., produced the smallest effect sizes), and therefore it is the accuracy on those that are presented in Table 8.

7 Concluding Remarks

This paper evaluated the effectiveness and applicability of several statistical tests to detect storage-based covert channels. The tests were selected based on their past success in being effective to detect timing-based covert channels. In particular, we conducted several experiments on sequences of IP IDs with and without information embedded into them. The results of the experiments show that complexity tests are much more effective that distributional tests for detecting storage-based covert channels. Many of the tests were determined to be able to detect covert channels on their own and combining multiple tests into a multi-dimensional classifier did not bring any significant improvement which means that simple covert channel detectors can be built using a single test.

In our ongoing and future work, we seek to build off of the results presented in this paper by exploring the practicality and effectiveness of more techniques for detecting storage-based covert channels such as those in [14, 16]. The goal is to adapt a collection of these tests into a high-performance platform to create a practical real-time storage-based covert channel detector.

References

1. Berk, V., Giani, A., Cybenko, G.: Covert channel detection using process query systems. In: 2nd Annual Conference for Network Flow Analysis (September 2005)
2. Berk, V., Giani, A., Cybenko, G.: Detection of covert channel encoding in network packet delays. Tech. Rep. TR2005-536, Dartmouth College, Hanover, NH, U.S.A. (August 2005)
3. Cabuk, S., Brodley, C.E., Shields, C.: IP covert timing channels: design and detection. In: 11th ACM Conference on Computer and Communications Security. pp. 178–187. ACM (2004)
4. Cabuk, S., Brodley, C.E., Shields, C.: IP covert channel detection. ACM Transactions on Information and System Security **12**(4), 22 (2009)
5. Collin, L.: A quick benchmark: Gzip vs. Bzip2 vs. LZMA. `https://tukaani.org/lzma/benchmarks.html` [Accessed: 22 October 2019] (2005)
6. Crespi, V., Cybenko, G., Giani, A.: Engineering statistical behaviors for attacking and defending covert channels. IEEE Journal of Selected Topics in Signal Processing **7**(1), 124–136 (February 2013)
7. Garcia, S.: Normal captures. `https://stratosphereips.org` (2017), Malware Capture Facility Project
8. Gianvecchio, S., Wang, H.: An entropy-based approach to detecting covert timing channels. IEEE Transactions on Dependable and Secure Computing **8**(6), 785–797 (2010)
9. Gunadi, H., Zander, S.: Bro covert channel detection (BroCCaDe) framework: Design and implementation. Tech. Rep. 20171117B, Murdoch University (2017)

10. Gunadi, H., Zander, S.: Bro covert channel detection (BroCCaDe) framework: Scope and background. Tech. Rep. 20171117A, Murdoch University (2017)
11. Gunadi, H., Zander, S.: Extending bro covert channel detection (BroCCaDe) with new plugins. Tech. Rep. 20171207A, Murdoch University (2017)
12. Gunadi, H., Zander, S.: Performance evaluation of the bro covert channel detection (BroCCaDe) framework. Tech. Rep. 20180427A, Murdoch University (2018)
13. Jadhav, M., Kattimani, S.: Effective detection mechanism for TCP based hybrid covert channels in secure communication. In: 2011 International Conference on Emerging Trends in Electrical and Computer Technology. pp. 1123–1128 (2011)
14. Jaskolka, J.: Modeling, Analysis, and Detection of Information Leakage via Protocol-Based Covert Channels. Master's thesis, McMaster University, Hamilton, ON, Canada (September 2010)
15. Jaskolka, J., Khedri, R.: Exploring covert channels. In: 44th Hawaii International Conference on System Sciences. pp. 1–10 (January 2011)
16. Jaskolka, J., Khedri, R., Sabri, K.: A formal test for detecting information leakage via covert channels. In: 7th Annual Cyber Security and Information Intelligence Research Workshop. pp. 1–4 (October 2011)
17. Kullback, S., Leibler, R.: On information and sufficiency. Annals of Mathematical Statistics **22**(1), 79–86 (1951)
18. Lempel, A., Ziv, J.: On the complexity of finite sequences. IEEE Transactions on Information Theory **22**(1), 75–81 (1976)
19. Li, Q., Zhang, P., Chen, Z., Fu, G.: Covert timing channel detection method based on random forest algorithm. In: 17th IEEE International Conference on Communication Technology. pp. 165–171 (2017)
20. Naik, B., Boddukolu, S., Sujatha, P., Dhavachelvan, P.: Connecting entropy-based detection methods and entropy to detect covert timing channels. In: Meghanathan, N., Nagamalai, D., Chaki, N. (eds.) 2nd International Conference on Advances in Computing and Information Technology, Advances in Intelligent Systems and Computing, vol. 176, pp. 279–288. Springer (2012)
21. Ponemon Institute: 2018 cost of a data breach study: Global overview. Tech. rep., IBM Security (2018)
22. Porta, A., Baselli, G., Liberati, D., Montano, N., Cogliati, C., Gnecchi-Ruscone, T., Malliani, A., Cerutti, S.: Measuring regularity by means of a corrected conditional entropy in sympathetic outflow. Biological cybernetics **78**(1), 71–78 (1998)
23. Sohn, T., Seo, J., Moon, J.: A study on the covert channel detection of TCP/IP header using support vector machine. In: Qing, S., Gollmann, D., Zhou, J. (eds.) Information and Communications Security, LNCS, vol. 2836, pp. 313–324. Springer Berlin/Heidelberg (2003)
24. Tumoian, E., Anikeev, M.: Network based detection of passive covert channels in TCP/IP. In: 30th IEEE Conference on Local Computer Networks. pp. 802–807 (2005)
25. Zhai, J., Liu, G., Dai, Y.: A covert channel detection algorithm based on TCP markov model. In: 2nd International Conference on Multimedia Information Networking and Security. pp. 893–897 (2010)
26. Zhao, H., Shi, Y.: A phase-space reconstruction approach to detect covert channels in TCP/IP protocols. In: 2010 IEEE International Workshop on Information Forensics and Security. pp. 1–6 (2010)
27. Ziv, J., Lempel, A.: A universal algorithm for sequential data compression. IEEE Transactions on Information Theory **23**(3), 337–343 (1977)
28. Ziv, J., Lempel, A.: Compression of individual sequences via variable-rate coding. IEEE Transactions on Information Theory **24**(5), 530–536 (1978)