NEP-IDS: a Network Intrusion Detection System Based on Entropy Prediction Error

A. Augello, G. Lo Re, D. Peri, P. Thiyagalingam

In Proceedings of the 2024 IEEE 49th Conference on Local Computer Networks (LCN)

DOI: https://doi.org/10.1109/LCN60385.2024.10639755

NEP-IDS: a Network Intrusion Detection System Based on Entropy Prediction Error

Andrea Augello, Giuseppe Lo Re, Daniele Peri, and Partheepan Thiyagalingam
{andrea.augello01, giuseppe.lore, daniele.peri, partheepan.thiyagalingam}@unipa.it
Engineering Department, University of Palermo, Viale delle Scienze, Ed. 6, 90128 Palermo, Italy

***Abstract*—Intrusion Detection Systems (IDSs) are used to intercept unauthorized access and malicious activity in computer networks. However, cyber-attacks are becoming more sophisticated, using evasion techniques to prevent signature-based detection. The rise of previously unseen attacks poses a critical challenge to IDSs. In this work, we present a lightweight approach to anomaly detection in network traffic that exploits the entropy of packet header features to reveal attacks. Detection is performed through a predictive model and a sliding window cumulative sum algorithm. The experimental evaluation, conducted on various attacks, indicates our system's effectiveness in detecting attacks generating both high and low amounts of traffic, maintaining a low false alarm rate.**

I. Introduction

The rapid evolution of the digital landscape has brought with it an increase in the complexity and frequency of cyberattacks. Numerous cyber-attacks are reported every year and attackers create more and more sophisticated techniques to break into computer networks [1]. These attacks, which are often sophisticated and stealthy [2], pose significant threats to information systems worldwide. At the same time technology systems are becoming more pervasive than ever before. As such, the development of effective detection methods is of paramount importance. Network traffic analysis is a critical component of cybersecurity, monitoring network availability and activity to identify anomalies and potential security issues [3]. However, traditional methods of network traffic analysis often fall short in the face of modern, complex cyber threats, thus requiring the use of Intrusion Detection Systems (IDSs) [4] to identify suspicious patterns or anomalies. IDSs provide real-time monitoring and analysis of network traffic to detect and prevent unauthorized access and malicious activities.

Due to evolving attacks, many challenges need to be addressed in the field of IDS, notably: reducing computational costs, improving transparency of results, and improving effectiveness against new types of attacks [5].

In this work, we present a Network Entropy Prediction Error-based Intrusion Detection System (NEP-IDS). Entropy, a concept borrowed from information theory, provides a measure of randomness or uncertainty in a set of data. In the context of network traffic, entropy can be used to quantify the unpredictability of traffic patterns. Anomalies, such as those caused by cyberattacks, often disrupt these patterns, leading to changes in entropy. By monitoring these changes, cyberattacks can be effectively detected in their early stages.

The main contributions of this work are the following:

- The system is based on a predictive model that can be trained on a relatively small amount of data and can be quickly deployed on a new network.
- Unlike most entropy-based IDSs, NEP-IDS's predictive model extends its effectiveness beyond typical high-volume Denial of Service (DoS) attacks.
- NEP-IDS is lightweight, requiring only a minimal set of features, and can be used online, with constant time and memory requirements.
- NEP-IDS is suitable for semi-supervised settings, highlighting traffic for further inspection or manual labeling, thanks to its interpretable output and low false alarm rate.

We show the applicability and effectiveness of NEP-IDS using a public dataset.

The rest of this work is organized as follows. Section II broadly reviews the state of the art in the field of IDS. Section III outlines our IDS components and operation. Section IV presents the experimental results. Finally, Section V reports our conclusions and discusses future research directions.

II. Related works

An IDS is a system security component that identifies malicious network traffic [6]. IDSs can be broadly classified into two categories [7]: Signature-based IDSs (SIDSs) and Anomaly-based IDSs (AIDSs). SIDSs detect attacks using pattern-matching techniques [8]. They store information from previously detected attacks in a database and raise an alarm when the oncoming network traffic matches a known attack signature. With labeled data, a model can be trained for accurate traffic classification [9]. Supervised IDSs, however, require large amounts of labeled data, and cannot adapt to new threats. The increasing frequency of polymorphic variants of malware and targeted attacks negatively affects their real-world effectiveness [1]. Thus, even if they perform well, effective techniques to recognize unknown attacks are still required [10].

AIDSs overcome this limitation by modeling normal system behavior and detecting attacks as deviations from this model. Since using human experts to label traffic data is time-consuming and expensive, and collecting large real-world datasets containing a wide range of attacks is difficult, keeping supervised IDSs up to date is a challenging task. An attractive solution to this issue is using an AIDS in parallel with the supervised IDS. The anomaly detection system can identify malicious traffic that eludes the classifier. Then this traffic can be manually labeled and used to update the classifier [11].

AIDSs can be classified by the methods used to construct behavior models [7]. *Knowledge-based techniques* use sources of information such as human experts to create rules defining normal system activity. However, inconsistencies may arise when merging information from multiple sources [12]. *Machine Learning AIDS*, on the other hand, are trained on large quantities of data to improve accuracy and require less human dependence. In the right configuration, they can be run on devices with limited computational capabilities while maintaining good detection accuracy [13]. However, machine learning models are often black-boxes, their decision-making process is opaque [14], and interpreting the reason of a packet classification is difficult.

AIDS can also be classified by the data they analyze. *Flow counting schemes*, primarily used for DoS detection [15], summarize network information in IP flows [16] but struggle with real-time analysis of encrypted traffic [17]. *Information theory-based schemes* analyze traffic using metrics such as entropy to measure the randomness of incoming traffic [18] and to assess whether the state of a network changed compared to some normalcy baseline. These approaches typically have low computational overhead, can handle a large amount of data in real-time, have few false positives, and need few assumptions on the underlying traffic generating mechanisms [19]. While fast at detecting large-scale events like DoS attacks [20], [21], [22], they struggle with small-scale, low-volume attacks [23].

Entropy-based IDSs typically set an entropy threshold that, when passed, labels traffic as malicious. For example, Distributed Denial of Service (DDoS) attacks decrease the entropy of the packet destination IP, revealing the ongoing attack [24]. Additionally, multiple thresholds can be combined to perform complex decisions, e.g., by using the entropy of different features to differentiate DDoS attacks from flash events [25]. Various entropy metrics, even when computed for the same features, can provide a richer context to classifiers, enhancing their performance against botnet-like malware attacks [26]. However, this approach is effective mainly against high-traffic attacks.

Fixed entropy thresholds require case-by-case tuning and regular updates due to changing network and attack characteristics, limiting their widespread use. Instead, adaptive bounds based on recent samples can improve DDoS detection [27]. Using multiple time windows can account for periodic traffic variations. Entropy is computed in various reference time frames, and new samples are compared to these baselines. Significant deviations can indicate abnormal states [26]. Good performances for DDoS detection was also achieved using probability distribution divergences of the incoming traffic compared to baseline network traffic [28]. Similarly to the proposed NEP-IDS, these techniques rely on establishing a baseline for the normal traffic and detecting deviations from it. However, since they do not use a predictive model, they are not as flexible in adapting to realistic network traffic variations. Additionally, low-volume attacks tend to escape detection, as in any given instant they do not significantly alter the entropy of the network traffic, and their effect is only noticeable after compounding over time.

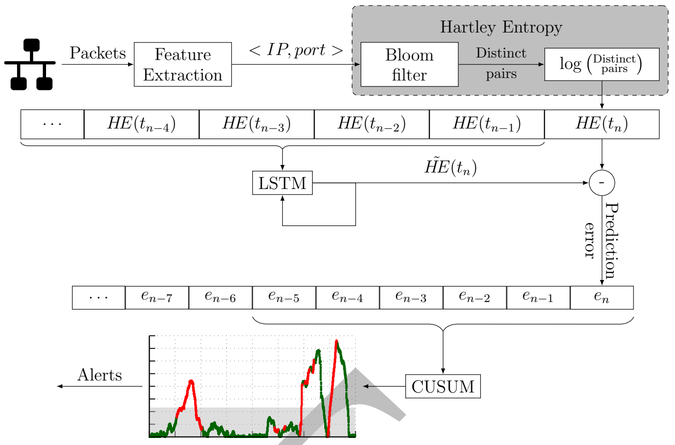


Fig. 1: Schematic diagram of the proposed system. The Long Short-Term Memory (LSTM) is first trained on benign traffic data. Then, the cumulative sum (CUSUM) algorithm is used to detect anomalies in the prediction error.

III. NEP-IDS Architecture

NEP-IDS is designed to detect anomalies in network traffic by leveraging the packet header information entropy. The system's architecture and components are illustrated in Fig. 1. NEP-IDS uses a predictive model to estimate future network traffic entropy, comparing predictions with actual entropy to identify anomalies (Section III-B). The predictive model is a Long Short-Term Memory (LSTM), a type of Recurrent Neural Network that can learn long-term dependencies in sequential data, making it resilient to input noise and ideal for time series prediction.

NEP-IDS operates in real-time, extracting header information from all the packets in transit through the network. The destination port and IP destination address of each packet are recorded and used to compute the Hartley Entropy (HE) of the "packet destination" variable. The entropy computation pipeline is highlighted in gray in Fig. 1 and is analyzed in greater detail in Section III-A.

The sequence of the computed entropy values is then used by the LSTM model to predict upcoming entropy values. The prediction is compared to the actual entropy, and the prediction errors are used as input for a sliding window cumulative sum (CUSUM) algorithm. The CUSUM chart is a statistical method used to detect anomalies. When the cumulative sum of the prediction errors exceeds a threshold, the system raises an alert, indicating a potential cyberattack.

In the following subsections, the main components of NEP-IDS and the rationale behind their design are further analyzed.

A. Hartley Entropy

Entropy is an information theory concept that quantifies the expected amount of information produced by a random variable. Usually, the entropy of a random variable X is computed using

Shannon's entropy as $H(X) = -\sum_{i=1}^{|X|} p_i \log(p_i)$, where p_i is the probability of the variable taking the i-th value. However, Shannon's entropy is not suitable for network anomaly detection and is rarely used in IDSs [29]. The contribution of an event to Shannon's entropy is small for events with very high or low probability as p_i and $\log(p_i)$ are close to 0, respectively. For network anomaly detection, however, multiple rare events (e.g., those generated by a port scan) and events with high frequency (e.g., the effect of a DDoS attack) are those linked to attacks. Moreover, attacks with a small volume footprint are almost negligible from a Shannon entropy point of view, making this entropy unsuitable for all low-volume attacks. Thus, IDSs often use Rényi's generalized entropy as in Eq. (1), which has a tunable parameter α to adjust sensitivity to rare/common events.

$$H_\alpha(X) = \frac{1}{1-\alpha} \log \sum_{i=1}^{|X|} p_i^\alpha \tag{1}$$

Positive α values highlight frequent events, while negative values emphasize rare ones. For $\alpha = 0$, Rényi's entropy is also known as Hartley Entropy (HE), and is computed as per Eq. (2).

$$HE(X) = H_0(X) = \frac{1}{1-0} \log \sum_{i=1}^{|X|} p_i^0 = \log |X| \tag{2}$$

HE equally weights all nonzero probability events, making it a general-purpose detection metric, not tailored for a specific attack class, and thus suitable for our aims. In NEP-IDS, we compute the entropy over the destination <IP, port> pairs. These features are a subset of some commonly used features in entropy-based IDSs [28]. Ignoring additional header fields speeds up computation and reduces the memory footprint. We focus on packet destinations, as sources can be easily spoofed or vary widely in attacks, while the destinations inside the security perimeter remain consistent. It is worth noting that most works tend to exclude these features, as a supervised classifier might learn to associate specific machines with attacks [30]. NEP-IDS is not subject to this limitation, as the header information is not directly fed to the LSTM model. Instead, all the pairs recorded in a time interval are used to compute a single entropy value that reflects the overall network state.

The computation of HE only requires the number of possible events $|X|$, not their probability distribution. Since only the number of distinct <IP, port> pairs is needed to compute the HE of the "packet destination" variable, the system can be optimized to require constant time and memory to compute the entropy each time interval. NEP-IDS uses a Bloom filter to efficiently select distinct <IP, port> pairs in each time interval, ensuring constant time and memory costs for HE computation. A Bloom filter is a space-efficient probabilistic data structure that tests set membership using a bit array and multiple hash functions [31]. Initially, the Bloom filter is a bit array of length m set to zero. A set of k hash functions is defined, each of which maps some set element to one of the m array positions, generating a uniform random distribution. To insert an element,

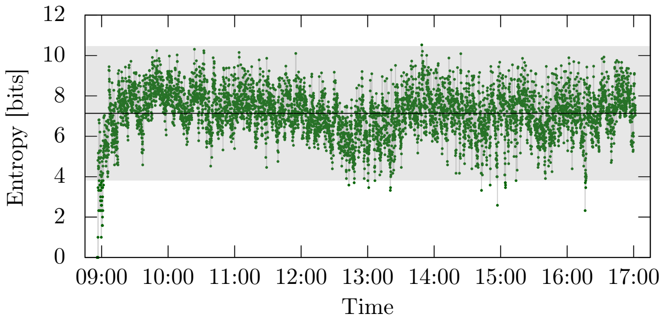


Fig. 2: HE of legitimate traffic on the first day of the dataset. The line represents the mean entropy, and the highlighted area spans six standard deviations.

it's hashed by the k functions, and the corresponding positions (modulo m) are set to one. k and m are chosen through Eq. (3), where p is the desired false positive probability for n elements.

$$k = -\log_2(p), \qquad m = -\frac{n \log_2(p)}{\ln(2)} \tag{3}$$

When an <IP, port> pair is received, it is hashed and the bits at the identified positions are checked. This operation requires constant time, regardless of the number of elements in the filter. If any bits are zero, the pair is added to the filter, and the distinct pair counter is incremented. Otherwise, the pair is considered present and ignored. Periodically, the HE is computed as the logarithm of the counter. After this computation, the counter and the Bloom filter are reset.

B. Prediction error

While many works use static entropy thresholds, this approach falls short in real-world scenarios where network behavior varies significantly throughout the day. This variation is to be expected, since network traffic is influenced by time-dependent user behavior and scheduled bandwidth-intensive processes. Network entropy alone, however, is not informative enough to understand the pattern of the network traffic [32]. The high entropy variability, shown in Fig. 2, complicates threshold setting. A low threshold causes many false alarms, while a high one misses attacks. It is also worth noting that, even with the wide bounds of the six standard deviations interval in the figure, many instances of legitimate traffic fall outside the interval.

To address this, rather than directly using the HE value for anomaly detection, NEP-IDS uses a predictive model to estimate future entropy. Said predictive model is an LSTM model with three recurrent layers having hidden size 5, and two fully connected layers with the same hidden size and Rectified Linear Unit (ReLU) activation. The LSTM is trained on entropy sequences from attack-free periods to learn the network's behavior.

At time t_{n+1} the LSTM has processed the observed entropy for each interval in the t_0–t_n range. We expect the predicted entropy $\tilde{HE}(t_{n+1})$ to be, on average, closer to the actual value $HE(t_{n+1})$ when the network is not influenced by ongoing attacks. We assume attacks alter network dynamics, causing the model, trained on legitimate traffic, to predict less accurately. The prediction error $e_t = \tilde{HE}(t) - HE(t)$ can indicate anomalies more effectively than raw entropy.

C. Anomaly detection module

In order to detect anomalies, NEP-IDS employs a sliding window cumulative sum (CUSUM) algorithm. The CUSUM algorithm is a sequential statistical hypothesis testing technique that detects shifts in the mean of a monitored variable, in our case the LSTM prediction error $e_t = \tilde{HE}(t) - HE(t)$. This algorithm entails computing the cumulative sum of differences between expected and observed values for all samples from the process start to the current sample under analysis. An alarm is raised when the sum exceeds a threshold. If the variable is normally distributed, the CUSUM score is near zero in normal conditions but increases when an anomaly shifts the mean.

To quickly return to normal levels post-anomaly, we use a sliding window mechanism [33] that only considers recent samples. At startup, all available samples are used until k samples are available. Then, the window starts sliding, computing the sum from $e_{\max(0,n-k)}$ instead of e_0 and using only the latest k samples. In the experiments, k is set to consider the latest 45 minutes of data.

NEP-IDS detects negative shifts in prediction error, assuming that an ongoing attack will increase HE over the LSTM prediction, which is trained on normal traffic. The CUSUM score to detect negative shifts is computed as in Eq. (4).

$$\begin{aligned} S_{\max(0,n-k)} &= 0 \\ S_{n+1} &= \max(0, S_n + \mathbf{E}[e] - e_{n+1}) \end{aligned} \tag{4}$$

The CUSUM score, initialized at zero, updates with each sample e_{n+1} by adding the difference between the expected and observed prediction errors. Traffic is labeled as anomalous if the CUSUM score S_t in a given time frame t exceeds a predefined threshold, rejecting the null hypothesis of no attack.

Finally, the complete NEP-IDS algorithm is summarized in Algorithm 1

Algorithm 1 NEP-IDS anomaly detection algorithm

```
Input: traffic stream, LSTM model, average prediction error,
       threshold, window size
Output: Anomaly alert
 1: Create an empty Bloom filter $BF$
 2: $distinctCounter \leftarrow 0$
 3: $errors \leftarrow []$
 4: for packet in traffic stream do
 5:     Extract packet header information: destination port and IP
 6:     if <IP, port> $\notin BF$ then
 7:         $distinctCounter \leftarrow distinctCounter + 1$
 8:         Add <IP, port> to $BF$
 9:     if interval has ended then
10:         $HE \leftarrow \log(distinctCounter)$
11:         $distinctCounter \leftarrow 0$
12:         RESET $BF$
13:         $\tilde{HE} \leftarrow$ LSTM.PREDICT_ENTROPY()
14:         $e \leftarrow \tilde{HE} - HE$
15:         APPEND $e$ to $errors$
16:         $S \leftarrow 0$
17:         for $i \leftarrow 0$ to $\min(windowSize, |errors|)$ do
18:             $S \leftarrow \max(0, S + \mathbf{E}[e] - errors[i])$
19:         if $S > threshold$ then
20:             return Anomaly alert
21:         LSTM.UPDATE_STATE($HE$)
```

IV. Experimental Evaluation

To demonstrate NEP-IDS' effectiveness, we need a dataset with diverse attacks and normal traffic in a network with multiple devices, containing raw network traffic for extracting packet header information and timestamps. Therefore, we evaluated NEP-IDS on the CIC-IDS2017 dataset [34]. This open-source dataset is a comprehensive and widely used dataset for network intrusion detection systems. It comprises ample realistic data, containing benign traffic, common attacks from various operating systems, and raw network traffic captures in a packet-based format.

The dataset spans five days, with benign traffic on Monday and diverse attacks thereafter. We use Monday's data, free of attacks, to train and tune our system. This approach is more representative of real-world scenarios than using attack-free intervals from across the entire dataset, as the training data in a deployed system would be collected pre-deployment. This test configuration illustrates that NEP-IDS can be quickly trained and deployed on a new network in a few hours without requiring extensive training data. To implement this configuration, the LSTM was initially trained on the first 5h 30m of Monday's traffic. The training was performed using the Adam optimizer and mean squared error loss function. Subsequently, the remaining 2h 30m were used as test data to characterize the statistical properties of the LSTM reconstruction error on previously unseen data.

Although setting thresholds at 2 or 3 standard deviations is a common practice in anomaly detection, percentile-based thresholding is more effective when the standard deviation is much larger than the mean, as with our CUSUM scores [29]. Thus, the threshold is set to classify 98% of the Monday test data as benign. This specific threshold value was selected according to the procedure described by the authors of [35].

In the following subsections, we evaluate the effectiveness of the NEP-IDS algorithm comparing it to a system using only entropy to classify incoming traffic (Singh et al. [28]). Then we perform an ablative analysis to assess the impact of the various components of the system. First, we keep the LSTM prediction error but ignore the historical error information leveraged by the CUSUM algorithm. Finally, we test the full system with the removal of the sliding window mechanism, using the entire dataset to compute the CUSUM score. These comparisons underscore our integrated architecture's effectiveness, outperforming the isolated components typically used in

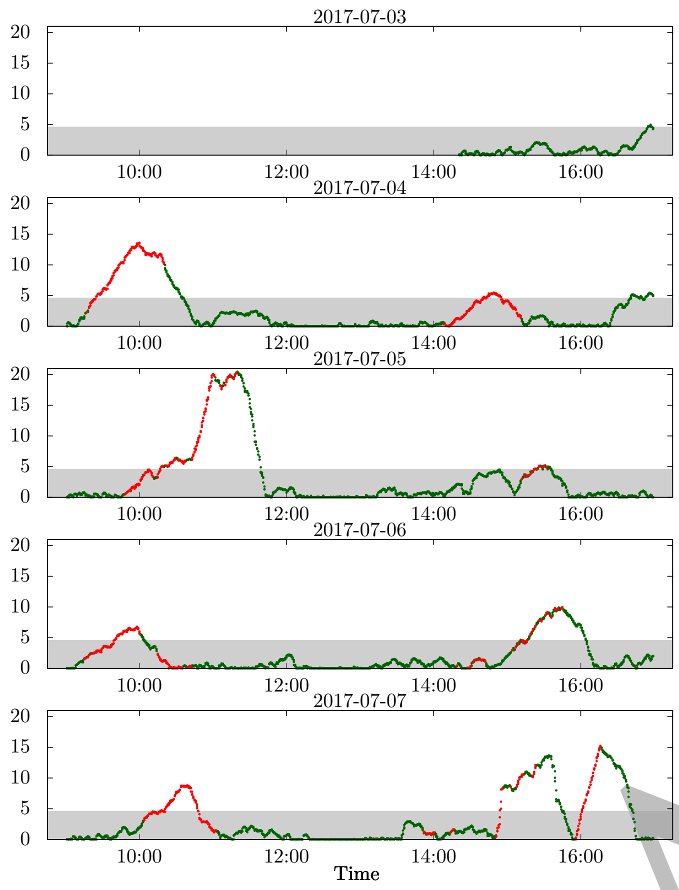


Fig. 3: Distribution of the CUSUM score as computed through Eq. (4). Points outside the shaded area are deemed malicious and are clearly distinguishable from normal traffic.

the literature.

A. Performance of NEP-IDS

We first provide a qualitative evaluation of NEP-IDS by showing the CUSUM score across all test data intervals in Fig. 3. This visual representation provides an insight into NEP-IDS's ability to detect attacks. In this representation, green and red points denote intervals without and with malicious activity, respectively. The shaded area represents the 98th percentile of the training data's CUSUM score, setting the detection threshold. Points outside this area are flagged as anomalies.

The figure shows that the CUSUM score of malicious traffic is easily distinguishable from normal traffic, with most attacks corresponding to sharp peaks in the CUSUM score. Conversely, the score assigned to benign traffic appears to be mostly flat, with occasional increases being small and short-lived. This ease of interpretation is a desirable property for an IDS as it allows a human operator to quickly assess the situation, estimate the beginning of the attack, and take appropriate action. Additionally, if human experts are tasked with labeling the traffic, by providing a clear visual representation of the system's output, NEP-IDS can help them quickly identify the intervals that lead up to an attack which require further inspection.

For a quantitative evaluation of the performance, in total, out of 35.5 hours of test data, 06:03:40 were labeled by our system as anomalous. Among the intervals classified as anomalous, only one 21-minute interval did not include any malicious traffic. Out of the total 09:05:00 of malicious traffic, attacks corresponding to 08:43:00 of traffic had at least one corresponding interval labeled as anomalous, resulting in a weighted detection rate of 95.9%.

Table I presents the time before each attack was detected by NEP-IDS. NEP-IDS failed to detect only the XSS and SQL Injection attacks. Despite the SQL Injection attack's distinct entropy distribution (significant with $p < 0.05$), its short duration (2 minutes) didn't allow the CUSUM score to surpass the alert threshold. NEP-IDS did not detect the XSS attack as its entropy and prediction error were statistically similar to normal traffic ($p > 0.1$). In general, the median delay from attack start to anomaly detection was 11 minutes and 30 seconds, resulting in more than half of the total attack duration occurring after the alert.

The longest detection time was for the two-stage PortScan attack on Friday 2017-07-07 from 13:55 to 15:29. In the first stage from 13:55 to 14:35 the firewall rules prevented additional traffic, so NEP-IDS could not detect the attack. With firewall rules off from 14:51 to 15:29, the second stage was detected in 6 minutes. Given the high incidence of port scan attacks in the wild [36], this is a desirable outcome, as NEP-IDS only raised an alert for the successful scan, while no alert was needlessly raised when the attack was intercepted by other security mechanisms in the network. Thus, NEP-IDS can be easily used as a complementary system to existing security measures, providing additional complementary insights into network traffic.

B. Comparison with entropy thresholding approaches

Setting entropy thresholds to raise alarms is a common approach in network anomaly detection, thus we consider a strategy that relies on this technique as a baseline for comparison. Specifically, we implement the approach described by Singh et al. in [28] that computes the Information Distance (ID) between the incoming traffic and the baseline traffic. First, this technique establishes a normalcy baseline using the mean and standard deviation of the entropy of benign traffic during the training period. Then, for the test data, the ID is computed as $ID = HE - \mu$. Values of ID outside the $\pm 3\sigma$ range are labeled as anomalies.

Fig. 4a shows the entropy distributions during normal and attack activities. The large overlap between these distributions makes ID, and by extension entropy, an ineffective anomaly detection metric. Only the DoS Hulk and PortScan attacks ever had an information distance greater than three standard deviations (3σ). Additionally, just 6.8% of all data points beyond the 3σ bounds were associated with malicious activities, highlighting entropy's limited discriminatory power. At the same time, 93.2% of all the data points outside the 3σ range

TABLE I: Time before NEP-IDS detected each attack and attack duration.

Attack	Time before detection (NEPEIDS)	Time before detection (Error Thresholding)	Attack duration
FTP-Patator	00:07	Not detected	01:03
SSH-Patator	00:31	00:04	01:02
DoS Slowloris	00:20	00:22	00:23
DoS Slowhttptest	00:02	Not detected	00:22
DoS Hulk	00:00	00:00	00:24
DoS GoldenEye	00:00	Not detected	00:09
Heartbleed	00:10	Not detected	00:20
Brute Force	00:25	00:22	00:45
XSS	Not detected	**00:20**	00:20
SQL Injection	Not detected	**00:01**	00:02
Infiltration	00:57	00:53	01:26
Bot	00:13	Not detected	00:58
PortScan	01:02	00:59	01:31
DDoS	00:05	Not detected	00:20

were not associated with any malicious traffic. Therefore, using entropy thresholds for anomaly detection leads to high false alarms and low detection rates, underscoring its unsuitability for general-purpose intrusion detection.

The distinct entropy distribution experienced during the DDoS also aligns with literature that primarily uses entropy to detect DoS and high-volume DDoS attacks. Nevertheless, this entropy stayed within normal ranges, evading detection. This confirms our hypothesis that realistic network traffic's natural variability makes entropy-based anomaly detection thresholds too lax. By contrast, the score from NEP-IDS (Fig. 4b) is distinct from the normalcy baseline for most attacks, reducing overlap with legitimate traffic and improving distinguishability.

We quantify this improvement in distinguishability through the detectability index. The detectability index d'_e [37] measures the separation between two distribution means in units of the average standard deviation as $d'_e = \frac{2\cdot|\mu_1-\mu_2|}{\sigma_1+\sigma_2}$, with μ_1 and μ_2 representing the means, and σ_1 and σ_2 the standard deviations of the two distributions being compared. Computing d'_e provides a quantitative measure of how well the distributions can be differentiated. d'_e values near 0 suggest indistinguishable distributions, while higher values indicate easier sample differentiation. This analysis is reported in Table II. Except for the DoS Slowloris, all attacks' detectability improved or remained nearly unchanged by using the NEP-IDS score instead of ID, with DoS GoldenEye's increasing sixfold.

C. Comparison with plain prediction error

The prediction error of the LSTM model is a better indicator of malicious activity than the entropy value even without further processing. Contrary to using the entropy alone, when relying only on the prediction error and setting the same 3σ threshold more than half of the attacks in the test data are correctly detected.

Table I also reports the time elapsed before the detection of each attack using the prediction error. Interestingly, XSS and SQL Injection attacks, the only two categories that NEP-IDS could not detect, were detected by naive error thresholding. Nevertheless, NEP-IDS was able to detect more attacks with a lower false alarm rate. Moreover, the CUSUM mechanism in NEP-IDS does not significantly delay the detection of attacks: for most of the detected attacks there was no substantial increase in the time elapsed before an attack was detected using NEP-IDS compared to prediction error thresholding. This result demonstrates that the detection performance improvement brought by the CUSUM mechanism in NEP-IDS does not come at the cost of a decreased responsiveness and real-time capabilities of the system.

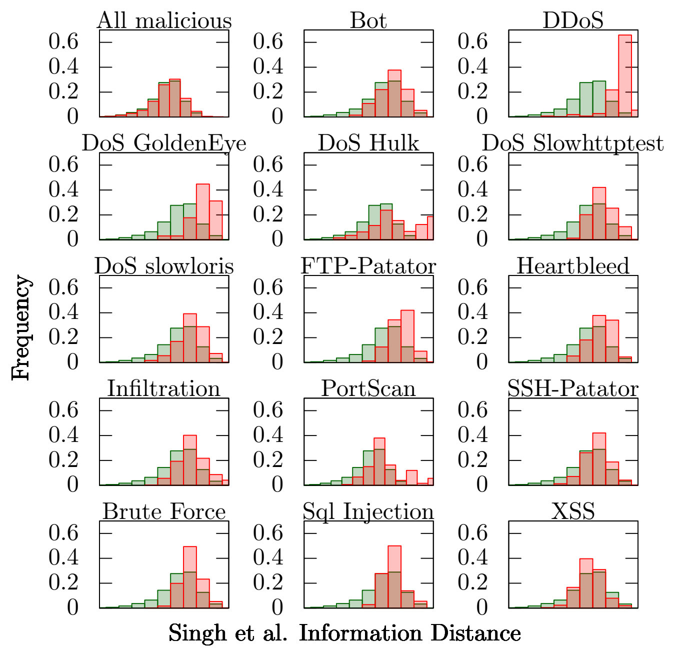
TABLE II: Detectability index for various attacks, comparing Singh et al. and NEP-IDS scores with and without the sliding window mechanism.

	Singh et al. [28]	NEP-IDS	No sliding window
FTP-Patator	1.10	2.42	1.22
SSH-Patator	**0.56**	0.52	0.26
DoS slowloris	**0.74**	0.32	**0.48**
DoS Slowhttptest	0.91	**1.84**	0.35
DoS Hulk	1.01	**3.05**	2.16
DoS GoldenEye	1.51	**9.31**	5.84
Heartbleed	0.79	**1.09**	0.20
Web Attack – Brute Force	0.83	**1.09**	0.06
Web Attack – XSS	0.12	**0.57**	0.01
Web Attack – SQL Injection	0.61	**0.91**	0.33
Infiltration	0.77	**0.90**	0.12
Bot	0.54	**1.27**	0.55
PortScan	**0.89**	0.86	0.14
DDoS	1.96	**1.59**	**2.04**
All malicious	0.79	**1.05**	0.48

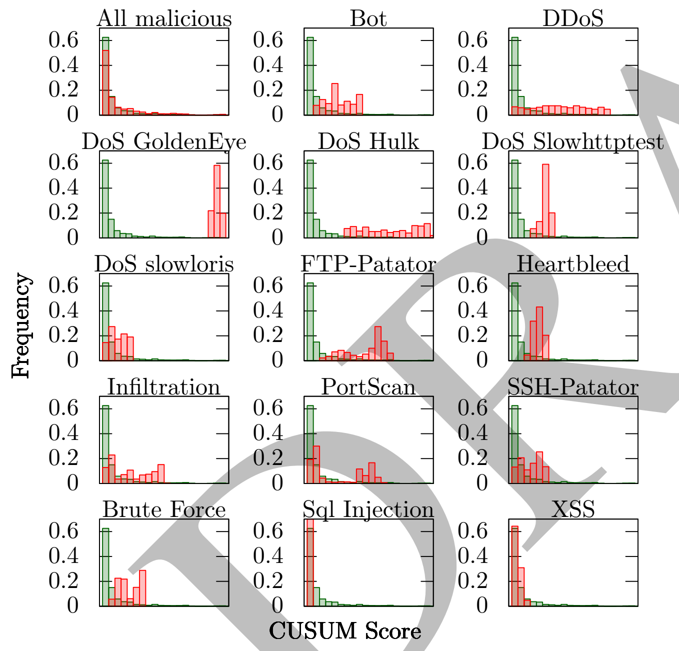
The prediction error of the LSTM model for all the test data is reported in Fig. 5. The shaded area denotes a 6σ interval around the mean error. From the figure it is possible to see that, despite the prediction error being more effective than the entropy value at detecting attacks, it still does not easily distinguish normal traffic from traffic influenced by the presence of malicious activities. A more in-depth analysis shows that most malicious traffic falls within the 6σ interval and, of all the data points that exceeded the threshold, 74.4% were still false positives.

D. Effect of the sliding window in the anomaly detection module

Finally, we assessed the sliding window mechanism's impact on the CUSUM algorithm. A comparison of the detectability with and without the mechanism is reported in Table II. Without the sliding window mechanism, the CUSUM algorithm's slower



(a) HE-based Information Distance distribution of benign (green) and malicious (red) traffic.



(b) NEP-IDS CUSUM score distribution of benign and malicious traffic.

Fig. 4: Distribution of the Information Distance and CUSUM score assigned by NEP-IDS for benign and malicious traffic.

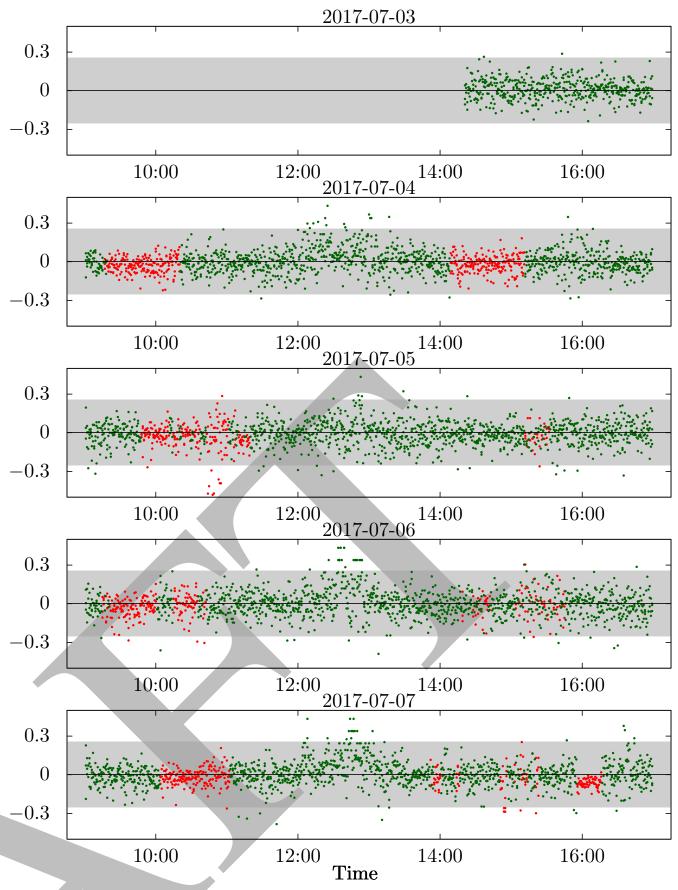


Fig. 5: Distribution of the LSTM prediction error. The shaded area is a 6σ interval around the mean. Most malicious traffic falls within this interval.

return to normalcy worsens most attacks' detectability. The slower return to normalcy results from the score incorporating prediction errors from intervals with attacks. This results in the system remaining in an alert state for a long time after the attack is over, potentially hindering the detection of subsequent attacks and increasing the false alarm rate. If the sliding window mechanism is not included in the CUSUM algorithm, the false alarm rate rises to 14.4% of the considered intervals, demonstrating the usefulness of this addition.

The limitations of the normal CUSUM are also visible in Fig. 6, where the performance of the algorithm without the sliding window mechanism is compared to the full NEP-IDS detection mechanism, shown in black. The figure illustrates how the CUSUM score of the algorithm without the sliding window mechanism is slower to return to normalcy after the attack is over, resulting in a higher false alarm rate. On the other hand, when the score is rising for an ongoing attack, the two algorithms have similar performance. Therefore, the addition of the sliding window mechanism does not constitute a trade-off between detection performance and responsiveness, as it improves the system's ability to return to normalcy post-attack without hindering the detection of ongoing attacks.

Two notable examples of the importance of the sliding window mechanism are on Tuesday 2017-07-04 for the FTP-Patator attack at 9:20-10:20 and the DDoS attack on Friday at 15:56-16:16. NEP-IDS returns to normalcy in 30 minutes post-Patator attack, while the system without the sliding window mechanism stays on alert for over 2 hours. Such a needlessly long alert state would result in considerable confusion for a

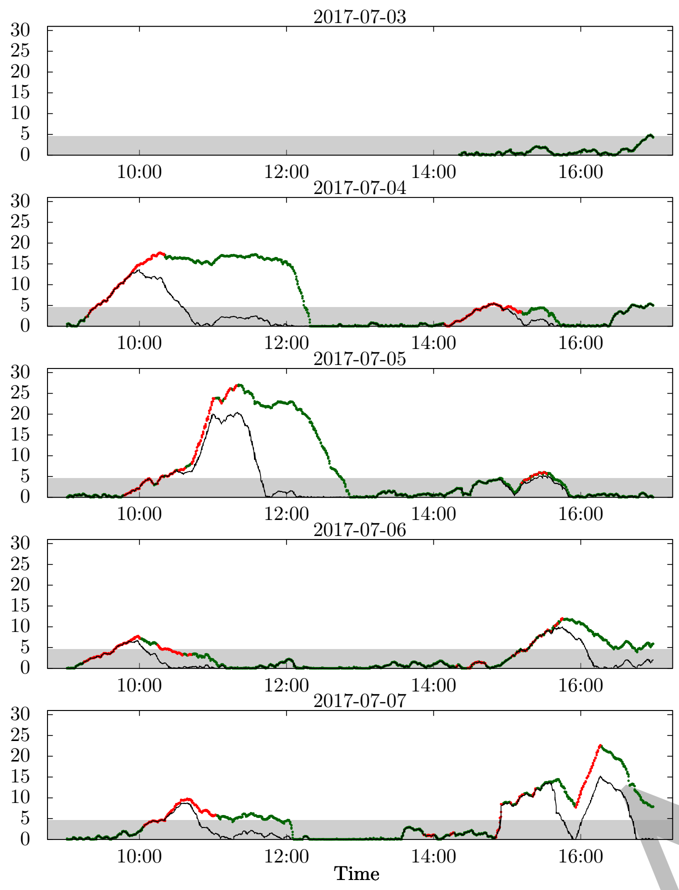


Fig. 6: Performance comparison with and without the sliding window mechanism. NEP-IDS (black) has fewer false alarms and recovers faster post-attack.

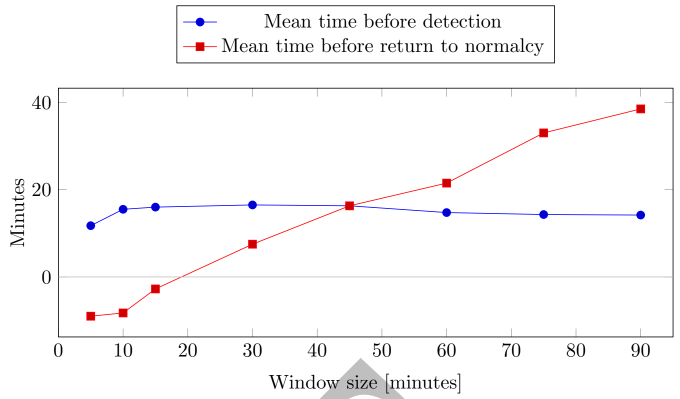


Fig. 7: Effect of the window size on the time before detection and extra alert duration.

human operator and unnecessary resource consumption for manual inspection.

In the case of the DDoS attack, the system without the sliding window mechanism is still in the alert state caused by the PortScan attack that ended at 15:29. A user inspecting the system at 15:56 would thus assume that the previous attack is still ongoing, potentially not noticing that a subsequent DDoS attack is now causing the system to remain in an alert state. Conversely, the full NEP-IDS detection mechanism correctly separates the two attacks, returning to normalcy after the PortScan attack and then raising a new alert for the DDoS.

Fig. 7 shows a sensitivity analysis of the window size. Increased window size does not significantly affect the responsiveness of the system, as the time before detection remains stable. On the other hand, the extra alert duration increases almost linearly with the window size, as the system takes longer to return to normalcy after an attack. The 45 minutes window size used in the experiments was chosen as the smallest window that prevented the system from raising multiple alerts for the same attack.

V. Conclusions

This work introduces NEP-IDS, an anomaly-based network IDS that leverages entropy prediction error, addressing the growing challenge of identifying novel cyber-attacks that would bypass traditional signature-based detection techniques.

NEP-IDS leverages packet header entropy for anomaly detection in network traffic, incorporating a predictive model and a statistical testing algorithm to detect ongoing attacks. We evaluated NEP-IDS on a diverse set of attacks, revealing its effectiveness in maintaining a low false alarm rate, while still detecting a broader range of attacks compared to conventional entropy-based systems.

NEP-IDS is very lightweight, requiring only minimal features, packet destination IP and port, without needing deep packet inspection or flow reconstruction. NEP-IDS is suitable for real-time use due to its constant resource needs. In our experimental evaluation, NEP-IDS outperformed traditional entropy-thresholding methods, expanding the repertoire of detectable attacks while maintaining efficient performance. The system demonstrates adaptability to evolving network conditions, making it a valuable addition to the field of intrusion detection.

As cyber threats continue to evolve, the need for more sophisticated and robust intrusion detection systems will continue to grow. NEP-IDS is a promising step in this direction, providing transparent, robust, and efficient detection against unknown attacks. Future works will investigate the application of our approach to stealthier attacks and the use of more complex change detection techniques, and will attempt to decrease the time before attack detection and the alert duration after the attack ends.

References

[1] Symantec, "Internet security threat report volume 22."

[2] M. Imran, H. U. R. Siddiqui, A. Raza, M. A. Raza, F. Rustam, and I. Ashraf, "A performance overview of machine learning-based defense strategies for advanced persistent threats in industrial control systems," *Computers & Security*, vol. 134, p. 103445, 2023.

[3] A. Shahraki, M. Abbasi, A. Taherkordi, and A. D. Jurcut, "A comparative study on online machine learning techniques for network traffic streams analysis," *Computer Networks*, vol. 207, p. 108836, 2022.

[4] T. Sommestad, H. Holm, and D. Steinvall, "Variables influencing the effectiveness of signature-based network intrusion detection systems," *Information Security Journal: A Global Perspective*, vol. 31, no. 6, pp. 711–728, 2022.

[5] G. Kocher and G. Kumar, "Machine learning and deep learning methods for intrusion detection systems: recent developments and challenges," *Soft Comput.*, vol. 25, no. 15, pp. 9731–9763, 2021.

[6] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.

[7] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersec.*, vol. 2, no. 1, pp. 1–22, 2019.

[8] V. Agate, F. M. D'Anna, A. De Paola, P. Ferraro, G. Lo Re, and M. Morana, "A Behavior-Based Intrusion Detection System Using Ensemble Learning Techniques," in *ITASEC 2022*, 2022.

[9] C. Yang, "Anomaly network traffic detection algorithm based on information entropy measurement under the cloud computing environment," *Cluster Comput.*, vol. 22, pp. 8309–8317, July 2019.

[10] D. Liao, R. Zhou, H. Li, M. Zhang, and X. Chen, "GE-IDS: an intrusion detection system based on grayscale and entropy," *Peer-to-Peer Networking and Applications*, vol. 15, pp. 1521–1534, May 2022.

[11] X. Meng, Y. Wang, S. Wang, D. Yao, and Y. Zhang, "Interactive Anomaly Detection in Dynamic Communication Networks," *IEEE/ACM Transactions on Networking*, vol. 29, pp. 2602–2615, Dec. 2021.

[12] L. Bouzar-Benlabiod, L. Meziani, A. Chebieb, N.-E. Rim, and Z. Mellal, "Experts' knowledge merging to reduce ids alerts number," in *2016 International Conference on Collaboration Technologies and Systems (CTS)*, pp. 418–423, 2016.

[13] K. Liu, Z. Fan, M. Liu, and S. Zhang, "Hybrid intrusion detection method based on k-means and cnn for smart home," in *2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 312–317, 2018.

[14] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead" *Nat. Mach. Intell.*, vol. 1, pp. 206–215, 05 2019.

[15] T. Jafarian, M. Masdari, A. Ghaffari, and K. Majidzadeh, "A survey and classification of the security anomaly detection mechanisms in software defined networks," *Cluster Comput.*, vol. 24, no. 2, pp. 1235–1253, 2021.

[16] M. F. Umer, M. Sher, and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Computers & Security*, vol. 70, pp. 238–254, 2017.

[17] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, pp. 1–99, 2018.

[18] Y. Chen, "Anomaly network traffic detection algorithm based on information entropy measurement under the cloud computing environment [j/ol]," 2018.

[19] G. Fernandes, J. J. P. C. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi, and M. L. Proença, "A comprehensive survey on network anomaly detection," *Telecommunication Systems*, vol. 70, pp. 447–489, Mar. 2019.

[20] A. S. S. Navaz, V. Sangeetha, and C. Prabhadevi, "Entropy based anomaly detection system to prevent ddos attacks in cloud," *CoRR*, vol. abs/1308.6745, 2013.

[21] İ. Özçelik and R. R. Brooks, "Deceiving entropy based dos detection," *Computers & Security*, vol. 48, pp. 234–245, 2015.

[22] A. Wagner and B. Plattner, "Entropy based worm and anomaly detection in fast ip networks," in *14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)*, pp. 172–177, 2005.

[23] M. A. Aladaileh, M. Anbar, A. J. Hintaw, I. H. Hasbullah, A. A. Bahashwan, T. A. Al-Amiedy, and D. R. Ibrahim, "Effectiveness of an entropy-based approach for detecting low- and high-rate ddos attacks against the sdn controller: Experimental analysis," *Applied Sciences*, vol. 13, p. 775, Jan 2023.

[24] R. Swami, M. Dave, and V. Ranga, "Defending DDoS against Software Defined Networks using Entropy," in *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pp. 1–5, Apr. 2019.

[25] S. Behal and K. Kumar, "Detection of DDoS attacks and flash events using novel information theory metrics," *Computer Networks*, vol. 116, pp. 96–110, Apr. 2017.

[26] P. Bereziński, B. Jasiul, and M. Szpyrka, "An Entropy-Based Network Anomaly Detection Method," *Entropy*, vol. 17, pp. 2367–2408, Apr. 2015.

[27] J. David and C. Thomas, "DDoS Attack Detection Using Fast Entropy Approach on Flow- Based Network Traffic," *Procedia Computer Science*, vol. 50, pp. 30–36, Jan. 2015.

[28] J. Singh and S. Behal, "A Novel Approach for the Detection of DDoS Attacks in SDN using Information Theory Metric," in *2021 8th International Conference on Computing for Sustainable Global Development*, pp. 512–516, Mar. 2021.

[29] A. A. Amaral, L. d. S. Mendes, B. B. Zarpelão, and M. L. P. Junior, "Deep IP flow inspection to detect beyond network anomalies," *Computer Communications*, vol. 98, pp. 80–96, Jan. 2017.

[30] X. Lin, G. Xiong, G. Gou, Z. Li, J. Shi, and J. Yu, "Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification," in *Proceedings of the ACM Web Conference 2022*, pp. 633–642, 2022.

[31] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, p. 422–426, jul 1970.

[32] C. Wang, X. Li, and E. Bertino, "Network temperature: A novel statistical index for networks measurement and management," *ACM T. Internet Techn.*, vol. 22, no. 3, pp. 1–20, 2022.

[33] E. Ahmed, A. Clark, and G. Mohay, "A Novel Sliding Window Based Change Detection Algorithm for Asymmetric Traffic," in *2008 IFIP International Conference on Network and Parallel Computing*, pp. 168–175, 2008.

[34] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization.," in *ICISSp*, vol. 1, pp. 108–116, 2018.

[35] M. T. Tun, D. E. Nyaung, and M. P. Phyu, "Network anomaly detection using threshold-based sparse," in *Proceedings of the 11th International Conference on Advances in Information Technology*, pp. 1–8, 2020.

[36] B. Hartpence and A. Kwasinski, "Combating tcp port scan attacks using sequential neural networks," in *2020 International Conference on Computing, Networking and Communications (ICNC)*, pp. 256–260, IEEE, 2020.

[37] A. J. Simpson and M. J. Fitter, "What is the best index of detectability?," *Psychol. Bull.*, vol. 80, no. 6, p. 481, 1973.