1-1-2021

SnapCatch: Automatic Detection of Covert Timing Channels Using Image Processing and Machine Learning

Shorouq Al-Eidi
Memorial University of Newfoundland

Omar Darwish
Ferrum College

Yuanzhu Chen
Memorial University of Newfoundland

Ghaith Husari
East Tennessee State University, husari@etsu.edu

Citation Information

Al-Eidi, Shorouq; Darwish, Omar; Chen, Yuanzhu; and Husari, Ghaith. 2021. SnapCatch: Automatic Detection of Covert Timing Channels Using Image Processing and Machine Learning. *IEEE Access*. Vol.9 177-191. https://doi.org/10.1109/ACCESS.2020.3046234

Received December 6, 2020, accepted December 16, 2020, date of publication December 21, 2020,
date of current version January 4, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3046234

SnapCatch: Automatic Detection of Covert Timing Channels Using Image Processing and Machine Learning

SHOROUQ AL-EIDI[1], OMAR DARWISH[2], YUANZHU CHEN[1], AND GHAITH HUSARI[3]
[1]Computer Science Department, Memorial University of Newfoundland, St. John's, NL A1B 3X9, Canada
[2]Computer Technology and Information System Department, Ferrum College, Ferrum, VA 24088, USA
[3]Department of Computing, East Tennessee State University, Johnson City, TN 37614, USA

Corresponding author: Shorouq Al-Eidi (shorouqa@mun.ca)

ABSTRACT With the rapid growth of data exfiltration carried out by cyber attacks, Covert Timing Channels (CTC) have become an imminent network security risk that continues to grow in both sophistication and utilization. These types of channels utilize inter-arrival times to steal sensitive data from the targeted networks. CTC detection relies increasingly on machine learning techniques, which utilize statistical-based metrics to separate malicious (covert) traffic flows from the legitimate (overt) ones. However, given the efforts of cyber attacks to evade detection and the growing column of CTC, covert channels detection needs to improve in both performance and precision to detect and prevent CTCs and mitigate the reduction of the quality of service caused by the detection process. In this article, we present an innovative image-based solution for fully automated CTC detection and localization. Our approach is based on the observation that the covert channels generate traffic that can be converted to colored images. Leveraging this observation, our solution is designed to automatically detect and locate the malicious part (i.e., set of packets) within a traffic flow. By locating the covert parts within traffic flows, our approach reduces the drop of the quality of service caused by blocking the entire traffic flows in which covert channels are detected. We first convert traffic flows into colored images, and then we extract image-based features for detection covert traffic. We train a classifier using these features on a large data set of covert and overt traffic. This approach demonstrates a remarkable performance achieving a detection accuracy of 95.83% for cautious CTCs and a covert traffic accuracy of 97.83% for 8 bit covert messages, which is way beyond what the popular statistical-based solutions can achieve.

INDEX TERMS Covert timing channels, detection, entropy, image processing, machine learning.

I. INTRODUCTION

Covert channels provide effective methods to exfiltrate sensitive data from the targeted networks. This type of exfiltration is particularly effective because it uses existing system resources, which were not originally designed to transmit sensitive data for the purpose of communication. By doing this, the transfer of the covert data becomes undetectable by traditional detection methods such as firewalls and intrusion detection systems. Due to the ability to transmit data without being detected, covert channels have become a serious threat to the professional domain as well as the general community of internet users. In addition to the fact that covert channels can be used to leak confidential information, they can be utilized by malicious parties to communicate and exchange information in order to coordinate devastating Distributed Denial of Service (DDoS) attacks [35].

Covert channels can be divided into two broad categories based on network resources used to transmit covert messages: covert storage channels and covert timing channels. In Covert Storage Channels (CSCs), the sender communicates with the receiver by writing covert information into a particular storage location, and then the receiver reads from that particular storage location such as a hard drive [16]. Many applications that are based on TCP, IP, and HTTP protocols can be used to establish covert storage channels. This type of cyber attack takes advantage of unused packet fields, such as TCP initial sequence number field, to maliciously

The associate editor coordinating the review of this manuscript and approving it for publication was Jeon Gwanggil.

embed covert messages that provide attackers with important information about the targeted system.

An excellent example of covert storage channels in a popular application is the ICMP error message. Many IP implementations use the packet's memory for storage. Some ICMP error packet fields, which are intended to echo back parts of the received message, may be utilized to store and leak information about the target's identity and/or determine if the targeted system contains any vulnerabilities.

The second type of covert channel is the Covert Timing Channels (CTCs). In this attack, the sender manipulates the timing of network events (e.g., packet arrival times) in a specific way to transfer sensitive information from the targeted system. The receiver observes the covert traffic and extracts the timing information to decode the covert bits that contain the covert information (e.g., user passwords) [18], [28]. CTCs have stochastic distribution, and hence detecting them is more challenging. Therefore, urging the need to design sophisticated protection techniques to defend against CTCs.

There are two main approaches to defend against covert channels: 1) blind thwarting and 2) thwarting after detection. Blind thwarting means limiting or blocking the possible presence of active covert channels without detecting them. This type of defense approach is enforced at the network boundary and applied to all network traffic to disrupt traffic streams that potentially carry covert information [6]. While these techniques can be used to disrupt CTCs effectively, they also have a serious impact on applications that have high Quality of Service (QoS) requirements, such as streaming video and remote desktop applications.

The second approach aims at detecting CTCs prior to thwarting (or blocking) them. Due to its ''detect before block'' nature, this defense strategy causes less impact on the quality of service, which makes it more suitable for applications with high QoS requirements. In general, the detection approaches of CTCs use statistical tests to distinguish between covert and overt traffic. These tests use measures such as inter-arrival time distributions and entropy to distinguish between covert from overt traffic. However, due to the high variation of overt traffic, these standard statistical tests are not accurate nor robust enough to identify CTCs. There have been various research efforts on detecting CTCs over the Internet. However, some of these detection techniques are designed only to detect one or two types of CTCs and are not applicable to detect other types of CTCs. The broader detection techniques are too sensitive to the high variance of the network traffic. Moreover, some of the detection techniques are computationally complex, which makes them inapplicable to networks with high QoS requirements.

In this article, we propose a new detection approach for CTCs to address the existing statistical-based techniques limitations. The proposed approach utilizes image processing and machine learning techniques to detect CTCs by converting the inter-arrival times of packets into colored images. This conversion allows for utilizing more accurate and robust image-based features. Once features are extracted from these images, our approach uses them to construct CTC detection models using the popular machine learning algorithms to classify traffic flows into overt or covert. To evaluate the image-based approach, we conduct a series of experiments to measure its accuracy in detecting CTCs. In addition, we evaluate the model's ability to locate the part of traffic flows where covert occurs. Moreover, we investigate our approach's robustness to detect CTCs that utilize different sizes of covert messages. Based on our best knowledge, image processing and machine learning techniques have not been applied in the literature to detect CTCs. In addition, there is no approach that detects the location of covert data within traffic flows.

In summary, our contributions can be summarized below:

- We propose a technique for converting CTCs inter-arrival times into two-dimensional (2D) colored images to generate robust features that we utilize for training machine learning classifiers.
- We present a novel method for detecting CTCs using image processing techniques and machine learning.
- We propose a mechanism that uses our trained classifiers to accurately pinpoint the covert part of the traffic that contain covert messages within a traffic flow. This allows our approach to alert the existence of CTCs to start covert mitigation defend mechanisms such as those mentioned in [4], [14], [25] or traffic blocking applications, which significantly improves the quality of service that gets impaired upon dropping the entire traffic flow.

The rest of this article is organized as follows. In Section 2, we review existing literature related to the detection of CTC and examine their effectiveness in comparison to the proposed approach. In Section 3, we describe our detection approach's design and further detail the proposed image processing techniques and feature extraction for training machine learning classifiers. In Section 4, we discuss the performance evaluation measures for CTC detection. Finally, in Section 5, we detail the conducted experiments using our proposed approach and discuss its effectiveness in detecting covert channels and in locating CTC sub-flows within traffic flows.

II. RELATED WORK

In this section, we discuss CTC detection and prevention approaches proposed in the literature. The research works that we have considered for the design and evaluation of our proposed approach can be grouped into two main categories: statistical-based CTC detection, and machine learning-based CTC detection. First, we provide an overview of each of these categories, then we discuss the relevant image processing-based approaches and applications.

A. STATISTICAL-DRIVEN CTC DETECTION

CTC detection methods are mostly focused on the analysis of network traffic. The majority of CTC detection methods observe network traffic behavior and extract statistical properties of covert and overt traffic and compare those properties to recognize anomalies and detect covert communication.

Several statistical tests are introduced in the literature, and their effectiveness for detecting CTCs is investigated. In [9], the authors detected a binary CTC by comparing overt traffic and covert traffic distribution. Similarly, in [7], a binary CTC was detected using overt traffic and covert traffic histogram. Berk *et al.* [5] investigated a simple statistical method for detecting CTCs. This method assumes that a stream of network traffic roughly fits a normal distribution; a stream with a bimodal or multi-modal distribution would suggest the presence of a covert timing channel. Thus, the method was among the first to focus on irregularities in the shape of network traffic distribution. Although simple tests such as distribution and histogram of traffic can detect simple CTC algorithms, these simple tests do not effectively detect robust and complicated CTC algorithms. Consequently, more effective statistical tests were introduced and investigated to detect CTCs.

Peng *et al.* [40] used the Kolmogorov-Smirnov (K-S) test to quantify the distance between two empirical distribution functions for traffic sample and the expected overt traffic sample. Their results demonstrated that when the distance between the test sample and training set was large, it indicates a possible CTC occurrence. Liu *et al.* [30] used the variance test to determine whether there are existing CTCs. In their approach, the variance test indicated a high distribution of transmitted data as evidence of existing CTCs, whereas a low data distribution indicated their absence.

Cabuk *et al.* [9] designed a detection method based on the variance of inter-arrival times of network stream to measure traffic's regularity score to detect covert timing channels. This approach interpreted low regularity scores as an indication of existing CTCs. Gianvecchio and Wang [17] used the entropy test as a metric to detect CTCs. The entropy test provides significant evidence of the existence of patterns inside the data signaling the presence of a covert channel with high probability. High entropy implies high randomness in the data and the absence of patterns. In contrast, low entropy implies the existence of a particular ordering in the data indicating the increased possibility of a covert channel. In [3], the authors extended Gianvechhio *et al.*'s work by introducing two additional statistical tests for detecting CTCs—the K-L divergence and Welch's t-test. They showed that Welch's t-test was better suited to detect the Jitterbug algorithm as it was a non-parametric test based on the difference of means. Shrestha *et al.* [43] introduced autocorrelation of the traffic distribution as an additional statistical test that can indicate the presence of covert communication. They showed that using a classifier system with autocorrelation properties of traffic as feature points outperforms other detection methods, providing reliable detection and generality.

To overcome the drawbacks of using flat statistical-based analysis for detecting CTCs, Darwish *et al.* [12], proposed hierarchical entropy analysis to detect covert timing channels. Moreover, they designed a solution that utilizes hierarchical entropy to divide the stream of inter-arrival times to identify the time-scale that best reveals the existence of a CTCs. Their results showed that the solution achieves significantly better accuracy than the solutions that use flat entropy. Darwish *et al.* [13] utilized the MapReduce technique to measure traffic flows' hierarchical entropy to improve CTC detection speed.

In summary, many approaches that use statistical tests have been proposed in the literature for detecting CTCs. However, due to the high variation and complexity of overt traffic, detection approaches that depend on statistical tests are not reliable nor robust for detecting CTCs.

Given the rapid increase of cyber attacks and their utilization of covert time channels to exfiltrate stolen data, this article aims to develop a new technique that utilizes image processing and machine learning techniques to detect CTCs accurately and robustly. Furthermore, we propose a mechanism to pinpoint the part of traffic flow (sub-flow) that contains covert traffic. Also, we investigate CTC detection for scenarios where the size of a covert message is small, which presents a challenging task for CTC detection methods in the literature.

B. MACHINE LEARNING BASED CTC DETECTION

Machine learning algorithms have been used in many CTC detection approaches because of their ability to effectively identify covert timing channels. In general, these approaches use various metrics (or features) to train and construct machine learning models using a labeled set of overt and covert traffic flows. Then, these models are used to classify new traffic flows to either overt or covert traffic. Zander *et al.* [47] proposed an approach that utilizes the decision tree learning algorithm. In their approach, the decision tree was trained using multiple statistical features extracted from traffic flows. The model's effectiveness in detecting the pattern of inter-arrival times of CTC packets was then tested using a set of both overt and covert traffic. The evaluation results of this work showed that the model was effective in detecting CTCs. Similarly, Iglesias *et al.* [23] using a set of statistical properties of traffic as features to build a decision tree model that can detect CTCs. The model was trained offline to avoid high computational costs during the detection process. Decision trees have also been applied in [22] to detect CTCs. The decision tree classifier was used to fine-tune parameters of a set of Descriptive Analytics of Traffic (DAT) detectors previously proposed by the same authors to detect covert channels in general. DAT detectors used a set of descriptive analytics-related traffic flow features such as the number of unique values, the sum of autocorrelation coefficients, and the total number of packets. Their results showed that descriptive analytics grounded the detectors; they exhibited high flexibility and easily incorporated new traffic features for future detection methodologies.

In [24], the authors analyzed if CTCs can be detected as strong anomalies according to their statistical properties in unsupervised machine learning (i.e., outlier detection algorithms). Their results disclosed that flows containing CTCs do not usually show extremes values or suspicious density differences. All covert flows are actually (mild) distance-outliers, but not all distance-outliers are covert flows; therefore, their detection with unsupervised methods is unsatisfactory. In [24], the authors proposed a CTC detection method based on using supervised and unsupervised machine learning algorithms. Their results demonstrated that CTCs showed high histogram distance-based outliers but insufficient to distinguish them from regular traffic due to the high shape variability of normal traffic. Therefore, the authors concluded that the combination of supervised and unsupervised methods (i.e., semi-supervise methods) appeared in the right direction to follow to develop accurate and reliable CTC detectors.

The Support Vector Machine (SVM) learning algorithm has also been widely used in the CTCs detection research domain. This is due to the fact that SVM has an exploration and classification power that goes beyond checking for statistical properties of traffic. Sohn *et al.* [45] demonstrated that simple covert channels encoded demonstrated that a simple covert channel encoded in the sequence number fields of TCP/IP protocol headers could be accurately detected using an SVM classifier. Recently, Shrestha *et al.* [44] proposed a reliable detection approach based on the SVM classifier for detecting CTCs. Their approach utilized four types of statistical features generated from four CTCs algorithms to build their model. The evaluation results of the method showed that machine learning techniques performed well in detecting CTCs. In [36], a new detection approach was proposed based on wavelet transform as features and SVM classifiers to detect various kinds of CTCs. Their approach relied on extracting maximum entropy features at different wavelet levels, and then these features were fed to SVM. In addition, the authors designed a sliding window-based scheme to detect complex traffic with several types of CTCs.

Artificial Neural Networks (ANN) learning algorithm has also been used for CTC detection. This is not surprising given the ANN effectiveness in recognizing patterns. Darwish *et al.* [15] proposed an approach that uses deep neural networks to build a CTC detection classifier. This classifier was trained using a set of statistical features extracted from the flow of inter-arrival times using the hierarchical statistical-based method. Their results showed that deep neural networks achieved higher accuracy compared to the SVM models.

Unfortunately, most of the machine learning-based approaches in the literature are based on statistical tests extracted from the network traffic, such as CCE values of the inter-arrival time of packets. Such metrics are extremely limited in CTC detection, especially when the covert message size is small (e.g., 8 bits). Another key limitation of the current approaches is that they become less reliable (and unpredictable) when the attacker tries to imitate benign traffic to evade detection.

To address these challenges, we propose a technique that yellowconverts the inter-arrival times of packets to colored images. Then, we utilize wildly popular image processing techniques to extract more robust image-based features from the colored images. We evaluate our approach's performance using three types of covert channels to show more complex cyber-attacks that are more difficult to detect. Also, we use different covert message sizes ranging from larger to smaller sizes (128 to 8 bits). Our approach shows remarkable performance to detect CTCs used by simple and more complex attacks, as well as small and larger sizes of covert messages. Our approach also shows a reasonable accuracy in pinpointing the location (i.e., set of packets) of covert messages in traffic flow. Based on our best knowledge, no approach in the literature can accurately do this.

C. IMAGE PROCESSING APPLICATIONS

Image processing techniques have been extensively studied and integrated with machine learning algorithms by the research communities in various domains. These techniques' popularity is due to their accuracy and efficiency in solving complex problems and detecting image patterns. This section provides a brief overview of the research and applications of image processing techniques in the healthcare, transportation, and cybersecurity domains. Research works at the intersection of these domains provide interesting and innovative integration of image processing and machine learning for medical diagnosis, monitoring vehicle movements, and network threats detection.

1) HEALTHCARE

Image processing is a widely used methodology in various medical sectors. For example, image analysis is very helpful in the early detection of various cancers. Paul *et al.* [39] proposed a solution integrating image processing techniques with deep neural networks to predict the odds of survival for lung cancer patients after a diagnosis is determined from CT scan images. A similar work [46] studied breast cancer detection using a combination of image processing techniques and a hybrid machine learning approach of deep neural networks and multi-criteria decision making. Their method achieved a reasonable accuracy (84.33%) compared to the classical machine learning algorithms, such as SVM. Moreover, Najadat *et al.* [37] used image processing techniques to detect abnormalities in the human brain using an average front-back CT image. This approach was able to detect general abnormalities from the whole CT image with high accuracy.

2) TRANSPORTATION

Traffic monitoring and analysis based on computer visualization techniques became valuable and essential due to the transportation evolution in recent decades. For example, Ma *et al.* [32] presented an image-based traffic speed prediction model. This model evaluates the traffic information extracted from vehicle trajectories to help the people choose better routes, predict traffic congestion, and support the traffic managers in controlling the road network and systematically allocating resources. The method of the speed prediction model contains two main procedures. The first procedure involves converting network traffic to images representing time and space dimensions of a transportation network as 2D of an image. The second procedure was to employ machine learning techniques for traffic prediction. The model results showed significant improvement within an acceptable execution time. On the other hand, our proposed model detects the CTCs as security threats in the network environments. Moreover, image processing and machine learning techniques are used to detect enemies in vehicle movements such as standing and traveling in the reverse direction. For instance, Sarikan and Ozbayoglu [42] presented a vehicle flow detection approach to distinguish traffic anomalies. In their model, image processing and machine learning technique k-nearest neighbor were adapted using different illumination levels of vehicle images. The proposed method was evaluated on a public highway and achieved a high detection accuracy.

3) CYBERSECURITY

Recently, various cyber-attacks detection techniques have been proposed to detect various network attacks. Most of these techniques utilize visualization models to monitor and detect malicious traffic patterns. For example, Anderson *et al.* [2] proposed a visualization model to detect malware malicious software samples based on the similarity between malware heatmap images. Nataraj *et al.* [38] used image processing techniques to transform binary code samples structure into 2D gray-scale images. Then extracted various features from these images to train machine learning classifiers to detect malicious software. The classifier achieved an accuracy of 97.18% in detecting multiple types of malware. Similarly, Han *et al.* [20] proposed a colored visualization model to detect complex malware software types and achieve high detection accuracy. Makandar and Patrot [34] converted malware binary into 2D gray-scale images and normalized the images into x dimensions to train the SVM classifier. The classifier achieved 92.52% accuracy in detecting 24 malware types. Makandar and Patrot [33] investigated using different classifiers to detect malware and discovered that ANN achieved the highest accuracy of 96.35%.

Recently deep machine learning and image processing techniques were integrated and used to detect malware. Cui *et al.* [11] and Luo and Lo [31] used CNN to extract features of malware images automatically. Both approaches achieved high accuracy in detecting malware.

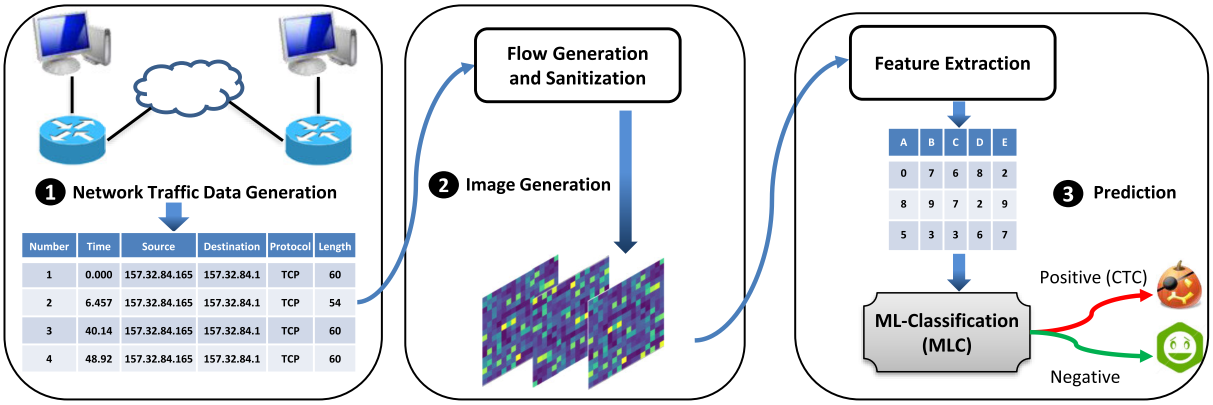
Image processing techniques also have been used to detect covert Storage Channel (CSC) threats. For instance, Kim and Reddy in [26] and [27] proposed a visualization model to detect the CSCs. In their model, the traffic data such as the IP address of sender and receiver were converted to images. Then image processing techniques have been applied to identify the existence of any data patterns in the traffic and detect CSCs. The results of the model achieved a high accuracy of such channel detection. However, the Kim and Reddy approach could not detect CTCs because CSCs utilize the data packets such as header fields or IP Identification fields for leaking the covert data while CTCs utilize the packet timing events such as inter-arrival times for leaking the covert data. Moreover, Liu *et al.* [29] presented a survey covering four network traffic visualization techniques used to monitor and detect malicious network threats: 1) visualization of bandwidth technique that used the advantage of utilizing the most bandwidth consumption in a particular location in the network to indicate attack existence and alert wherever it was detected. On the other hand, instead of using the bandwidth consumption as indicator, our model utilized the inter-arrival times of traffic that are converted to colored images, then used to detect CTCs. 2) The visualization propagation delay technique mainly focused on the idea that servers which are getting attacked will start being slower on transmitting data than the normal servers. However, the attacker in this context has no control over the inter-arrival times and does not leak any information through them. On the other hand, our contribution focuses on detecting the data leaked through inter-arrival times. 3) Visualization of the packet attributes technique was used to detect network traffic threats by visualizing and analyzing the packet attributes, including IP addresses, ports, and protocols. However, the contribution in this technique was designed for CSCs, not CTCs. Our contribution is mainly focused on CTCs. 4) Visualizing communication paths technique was used to detect network threats by finding and visualizing the weak routing nodes and abnormal network topologies. In this technique a 3D model was used where different colors were assigned to the IP addresses nodes in networks. Then, the network performance was evaluated by measuring the IP forwarding paths to identify any threats. However, there are no data leaked based on inter-arrival times in this work. Moreover, in the case of CTC detection, we are focusing on detecting leaked data using inter-arrival times. Also, such a 3D model is ineffective for detecting CTCs because the path between the covert sender and receiver is unobservable due to using overt traffic for leaking covert data.

In summary, image processing techniques have been applied successfully in various applications and domains in which big data needs to be processed and classified. Therefore, the integration of image processing techniques and machine learning algorithms is a strong candidate for detecting CTCs with high performance. To the best of our knowledge, image processing techniques and machine learning have not been applied to detect CTCs.

III. SNAPCATCH: IMAGE-BASED CTC DETECTION

This section presents our CTC detection approach and details the steps to detect CTCs using image processing and machine learning techniques.

Our approach can be compartmentalized into three main processes: 1) traffic dataset generation; 2) converting packets inter-arrival times into colored images representations; and 3) feature extraction from images and machine learning model construction for CTC detection.



Number	Time	Source	Destination	Protocol	Length
1	0.000	157.32.84.165	157.32.84.1	TCP	60
2	6.457	157.32.84.165	157.32.84.1	TCP	54
3	40.14	157.32.84.165	157.32.84.1	TCP	60
4	48.92	157.32.84.165	157.32.84.1	TCP	60

A	B	C	D	E
0	7	6	8	2
8	9	7	2	9
5	3	3	6	7

FIGURE 1. **Workflow of CTC detection using image processing and machine learning.**

Figure 1 overviews the major components of our approach. First, we design and implement an environment that contains two systems (sender and receiver) communicating over the internet from two different countries. Additionally, we design a malicious agent that injects (encodes) covert messages into the sender's traffic flows. This agent injects covert data based on three different defense evasion techniques with configurations that range from greedy data exfiltration (i.e., massive uploading of covert data) to cautious and stealthy exfiltration of the small size of data. Then, our packet capturing module utilizes Wireshark [10] to monitor and capture the traffic flows between the sender and receiver systems. The captured set of traffic flows are comprised of both overt (benign) and covert (malicious) traffic. Next, our approach extracts the packets' inter-arrival times from these flows and store them in the flow dataset. The image processing module ingests inter-arrival times of each traffic flow and converts them into colored images. Then, our feature extraction module extracts eight selected features from the colored images. Finally, our classification module takes the sets of features as input and constructs machine learning classifiers to detect the images that contain covert traffic flows. We describe the details of each module next.

A. RAW DATA GENERATION

In this step, we generate real covert timing channels between two computers (Sender and Receiver) located in different countries. These machines have two applications communicating with each other over reliable TCP connections. Then, the malicious agent creates and encodes covert messages to be sent to the receiver. This malicious agent aims to send covert data to the receiver in a way to make it look like overt traffic, as shown in Figure 2. There are two main configuration variables of the malicious agent that control its ability to evade detection: 1) choosing the time-delays of its packets, and 2) choosing the size of its covert messages. The delay of transmission times of packets plays a key role in evading detection by cyber defenses. For example, sending covert messages without considering (or imitating) benign packet

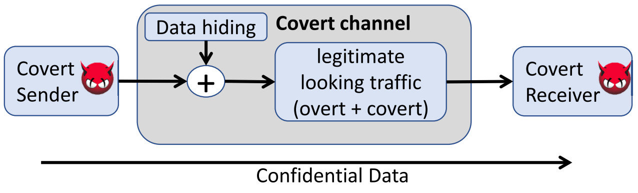


FIGURE 2. **Covert communication model.**

time-delays, like using longer time-delays than overt traffic, will result in the detection of the covert flow. Also, if the malicious agent greedy sends large covert massages, it will be easier to detect than agents that use smaller covert messages (e.g., 8 bits).

We build on the work of Al-Eidi *et al.* [1], which studied the effect of time-delays of covert traffic on the detection process. Their analysis discovered that using packet delays similar to the delays of overt traffic was a good strategy to evade detection. Therefore, our agent uses packet time-delays that are based on inter-arrival times of overt traffic. Furthermore, we implemented three different time-delay configurations to emulate three types of attacks with varying complexity: (1) greedy, (2) cautious, and (3) ultra-cautious CTC attacks.

The second setting for imitating overt traffic is choosing the size of the covert message. As mentioned earlier, malicious agents can send the stolen data in large or small covert messages. Using large covert messages cause a quick change of traffic activity, and therefore, is easier to detect than using small messages. To examine the effects of different covert message sizes on the accuracy of the CTC detection, we equip the malicious agent (the sender) with the capability to use three different sizes for covert messages: 8, 64, and 128 bits. The results of these experiments are further discussed in Section IV.

In this work, we generated two types of datasets. The first type is for the channel detection purpose, which contains 4,608,000 inter-arrival times for overt and covert traffic were collected based on using the three CTC attacks and the three sizes of covert message. The second type is for the channel localization purpose, which contains 1,534,464 inter-arrival times for covert traffic, were collected based on using three sizes of covert messages that were injected in three locations: beginning, middle, and end in the traffic flow. The details of each dataset shown in Tables 1 and 2, respectively.

TABLE 1. Detection dataset parameters.

Covert message size	8, 64, 128 bits
Delay time	2μ, 0.5μ, 0.25μ
The mean IATs of overt traffic (μ)	0.0664 seconds
Sub-flow size	256 inter-arrivals
# of dataset versions	9
# of images (all datasets)	18000
# of covert images (all datasets)	9000
# of overt images (all datasets)	9000
# of images per dataset version	2000
# of covert images per dataset version	1000
# of overt images per dataset version	1000

TABLE 2. Localization dataset parameters.

Covert message size	8, 64, 128 bits
Delay time	2μ
The mean IATs of overt traffic (μ)	0.0664 seconds
Sub-flow size	256 inter-arrivals
# of dataset versions	3
# of images (all datasets)	5994
# of images per dataset version	1998
Location of covert message	Beginning, middle,end

B. CONVERTING INTER-ARRIVAL TIMES INTO COLORED IMAGES

The previous step produces a dataset containing the packets' inter-arrival times (IATs) of the traffic flow captured from the communication between the sender and the receiver. In this step, our approach coverts these inter-arrival times into colored images based on their values. This enables our approach to utilize the popular image processing techniques to extract more robust image-based features for further processing. To do this, firstly, each sub-flow of inter-arrival times was placed into a 2D 16 × 16 matrix. Each matrix is filled with the inter-arrival times in row by row and left to right manner. Then, the inter-arrival times in each matrix are normalized into a range of 0 and 255 that represents a color image. Finally, each matrix is interpreted as a colored image by converting each of its normalized elements (16 × 16) to a color pixel. For this purpose, the matplotlib library [21] was utilized, which can create an image from a 2D array. The image will have one square for each element of the 2D array and the color of each square is defined by the value of the corresponding array element as shown in Figure 3.

C. FEATURE EXTRACTION AND IMAGE CLASSIFICATION

Once the colored images are generated, our approach extracts eight popular features from these images. These features

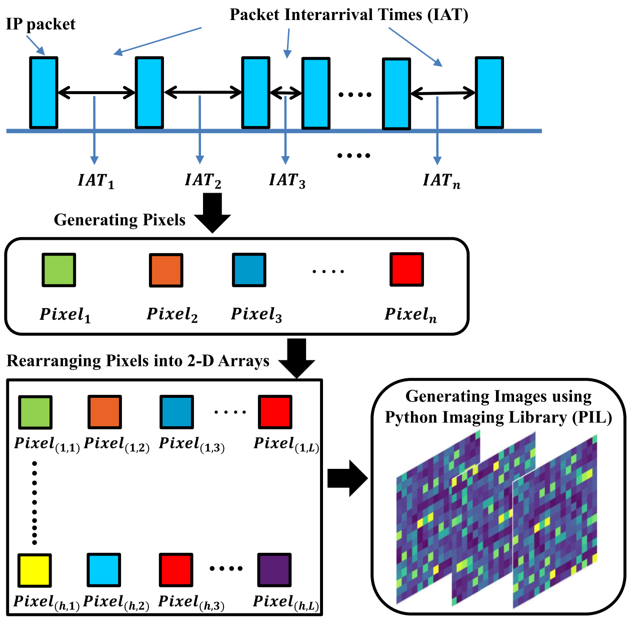


FIGURE 3. Illustration of inter-arrival time images.

are then used to train and construct a machine learning classifier for covert traffic detection. In this section, first, we describe these features and the tools used for extraction. Then, we describe the machine learning algorithms used to construct and validate the CTC detection classifier.

1) FEATURE EXTRACTION

As mentioned earlier, each colored image that is generated by the previous step represents a traffic sub-flow. Our feature extraction module takes these images as input and extracts eight popular features from each image. We enlist and explain each one of these features as follows:

1) **Mean gray value**, which calculates the average gray value within the image by taking the sum of the gray values of all the pixels in the image divided by the number of pixels.
2) **Standard deviation**, which calculates the standard deviation of the gray values used to generate the mean gray value.
3) **Mode**, which represents the most frequently occurring gray value within the image. Corresponds to the highest peak in the histogram.
4) **Center of mass**, which calculates the brightness-weighted average of the x and y coordinates of all pixels in the image. These coordinates are the first order of spatial moments.
5) **The integrated density**, which calculates the area of one pixel multiplied by the sum of pixel values in the image.
6) **Median**, which calculates the median value of the pixels in the image.
7) **Skewness**, which calculates the third-order moment about the mean where the area of the distribution falls, to the left or right of the mean.

8) **Kurtosis**, a distribution measure to describe whether a distribution is peaked or flat of the mean.

We used the ImageJ tool [41] to extract these features from the images. Then, we stored the extracted features in a CSV file to be used in the classifier training process. We explain this process next.

2) MACHINE LEARNING MODEL CONSTRUCTION

Our approach's final step is to construct a machine learning model to classify the generated images into either covert or overt. For this task, we train a set of models using the following machine learning algorithms:

- Support Vector Machine (SVM).
- Decision Tree (DT).
- Naïve Bayes (NB).
- Artificial Neural Networks (ANN).

We trained each classification model using the features extracted from each image. Then, we evaluated the models using the hold-out validation method. Based on the hold-out validation method, we split our dataset into 75% for training and 25% for testing (validation).

To investigate which machine learning algorithm performs better in this domain, we adopted four machine learning algorithms: Support Vector Machine (SVM), Decision Trees (DT), Naïve Bayes (NB), and Artificial Neural Networks (ANN). The SVM classifier works on the dataset by defining the best hyperplane that divides the best observations (features) into two parts representing the two classes (overt and covert). On the other hand, Using the DT classifier has the advantage that a human can interpret the resulting decision tree in terms of which features are the most accurate (i.e., have the lowest Gini impurity value). Moreover, the most useful (determining) features are always used at the top of the tree, and the irrelevant features are ignored (pruned). NB and ANN are very popular machine learning algorithms that continue to achieve high accuracy measures in pattern recognition due to their ability to discover underlying relationships in the data.

To construct and validate these models, we use the popular data mining tool: Waikato Environment for Knowledge Analysis (WEKA 3.8) [19]. We discuss and present our experiments and evaluation results next.

IV. EVALUATION

Our evaluation seeks to measure the performance of our approach in the following terms: (a) the effectiveness of our approach to detect covert timing channels under different defense evasion configurations of cyber attacks; (b) the ability of our approach to pinpoint the covert part (set of packets) of the traffic sub-flow; and (c) compare and contrast different machine learning classifiers based on their accuracy and interpret-ability in detecting CTCs.

A. ACCURACY MEASURES OF CTC DETECTION

In machine learning and information retrieval, there are various metrics to measure the different aspects of accuracy. For example, `recall` is an accuracy measure that targets the *completeness* of detection (i.e., the fraction of the total CTCs that were detected). On the other hand, `precision` measures the quality (or exactness) of detected CTCs.

Different factors may be considered when adopting the accuracy measure(s). Prioritizing the recall measure indicates a cautious detection approach: leaning towards the completeness of detected CTCs rather than precision (false positives). This strategy minimizes missed CTCs at the expense of falsely classifying overt traffic as covert. Some situations may require prioritizing precision rather than recall: leaning towards keeping false alarms low at the expense of missing some CTCs.

Our evaluation aims at providing a comprehensive analysis of various classifiers and accuracy measures to provide the flexibility to select the classifier whose accuracy specifications that are most relevant to users (stakeholders). Therefore, we use the most popular accuracy measures in machine learning and information retrieval domains. We list these measures and explain each one next.

- Accuracy (A): calculates the number of correctly classified instances of both classes (overt and covert) over the total number of instances using the following equation:

$$A = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

- Precision (P): calculates the number of correctly detected class members by the classifier over the total number of correctly and incorrectly detected members using the following equation:

$$P = \frac{TP}{TP + FP} \tag{2}$$

- Recall (R): calculates the number of correctly detected class members over the total number of class members using the following equation:

$$R = \frac{TP}{TP + FN} \tag{3}$$

- F_1 score: provides a score that balances both the concerns of precision and recall in one measures using the following equation:

$$F_1 = \frac{2 \times (precision \times recall)}{(precision + recall)} \tag{4}$$

where True Positive (TP) is the number of segments (images) that are correctly classified as CTCs. True Negative (TN) is the number of segments that are correctly classified as non-CTC (overt) channels. False Positive (FP) is the number of segments that are incorrectly classified as CTC channels. False Negative (FN) is the number of segments that are incorrectly classified as non-CTC channels while, in fact, they are CTCs.

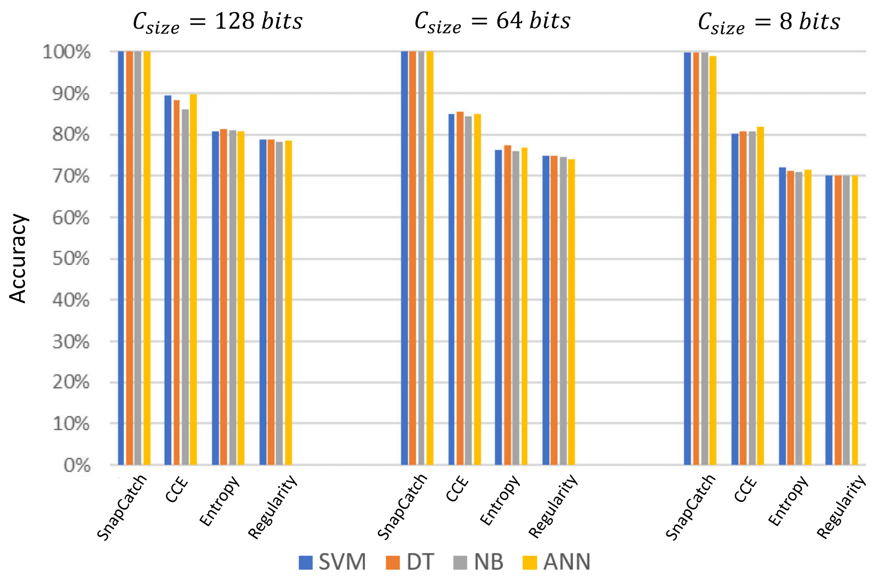


FIGURE 4. The impact of the covert message size on the detection accuracy of Greedy CTCs.

B. EXPERIMENTAL RESULTS

To validate the accuracy and efficiency of our proposed approach, we present our experimental results and compare them with three popular baseline CTC detection approaches. These detection approaches are: (1) the regularity test [8], (2) entropy test, and (3) corrected conditional entropy [17], [44].

In addition, to measure and compare our approach's effectiveness in detecting different types of covert channels, we consider different configurations of covert channels that range from simple to stealthy CTCs that utilize advanced defense evasion techniques.

For this reason, we designed the experiments using three types of CTC attacks driven from real-world scenarios: (1) Greedy Covert Timing Channels (GCTC), this type of channels does not consider the normal time-delays of overt traffic when sending packets. Therefore, it is considered the easiest type of attack to detect. (2) Cautious Covert Timing Channels (CCTC) attempts to partially imitate the delay-times of overt traffic, making this type of covert channels harder to detect. (3) Ultra-Cautious Covert Timing Channels (UCCTC) are the most advanced and hardest type of covert channels to detect. This type of channel restricts the malicious sender from exhibiting behavior that deviates from overt traffic at the expense of the time and the stolen information's quality. We provide more details about these attacks and present our approach's evaluation results for each attack next.

1) GREEDY COVERT TIMING CHANNELS (GCTC)

Our first set of evaluation results shows our approach's performance in detecting Greedy Covert Timing Channels (GCTC). GCTC is the simplest type of covert channels as it does not seek to overlap with (imitate) the time-delays of overt traffic. Therefore, it often causes abnormality in traffic when compared to overt traffic. This detectable abnormality makes this type of channel easier to detect.

To simulate the GCTC attack, the packet time-delay configuration was set to be equal to the double mean inter-arrival time of overt traffic. Also, to test our approach's effectiveness in detecting the different sizes of covert messages for this attack, we ran our experiments using three covert message sizes: 8, 64, and 128 bits. In each experiment, these covert messages were injected into traffic flow of 256 packets that were transmitted from the sender to the receiver.

As explained earlier, we utilized Wireshark to capture the generated traffic and used our image construction technique to convert the traffic flows into colored images. Then, we extracted the eight features from these images and utilized them for training four different classifiers (SVM, DT. NB, ANN) using 75% of the data, and we tested the classifiers using the other 25% of the dataset.

Figure 4 shows the detection results for the GCTC attack using the three covert message sizes: 8, 64, and 128 bits. As shown by the Figure, the highest CTC detection accuracy of the GCTC attack was achieved by the SVM classifier trained by our approach, which achieved the accuracies of 99.83%, 100%, and 100% for the covert message sizes of 8, 64, and 128 bits respectively.

The second highest CTC detection was achieved by the CCE, which reached 82%, 85.5%, and 89.83% for the covert message sizes of 8, 64, and 128 bits, respectively. The entropy-based approach achieved the third place with the accuracies of 72%, 77.33%, and 81.33%, where the regularity-based approach achieved fourth place with the accuracies of 70.2%, 74.83%, and 78.83% for the covert message sizes of 8, 64, and 128 bits respectively.

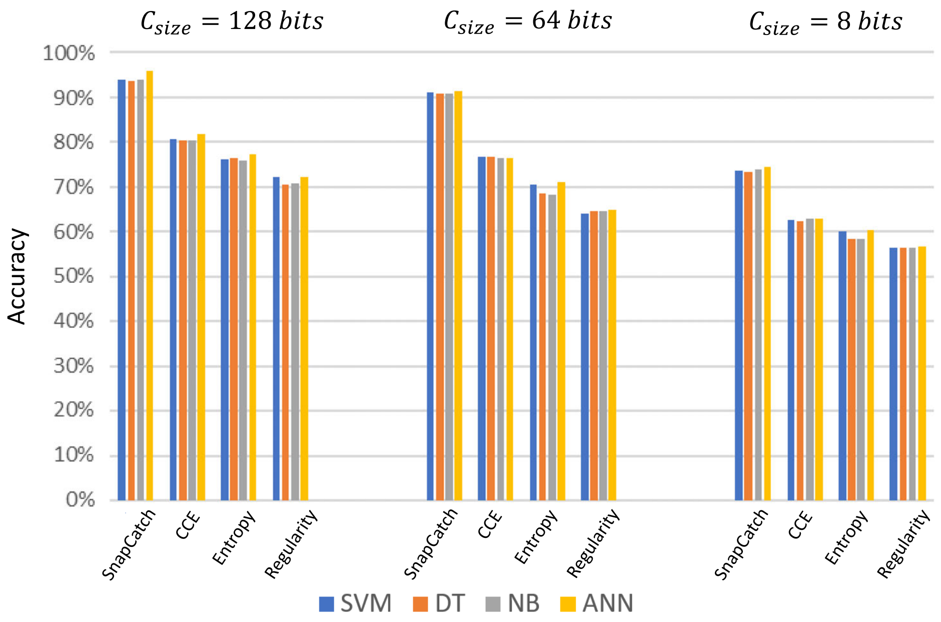


FIGURE 5. The impact of the covert message size on the detection accuracy of Cautious CTCs.

2) CAUTIOUS COVERT TIMING CHANNELS (CCTC)

Our second set of evaluation results shows our approach's performance in detecting Cautious Covert Timing Channels (CCTC). CCTC is a more advanced cyberattack that seeks to imitate the overt traffic behavior (packet inter-arrival time) to a certain degree. This type of covert channel exhibits near-overt traffic behavior using packet inter-arrival times that are partially similar to overt traffic. Because of its similarity to overt traffic, CCTC is a more challenging type of covert channel to detect.

To simulate the CCTC attack, the packet delay was set to be equal to half of the mean inter-arrival time of overt traffic. This delay, when combined with the delay enforced by the network, exhibits less abnormality in the traffic, which makes it harder to detect by the security detection measures. We ran the experiments for the CCTC attack using the three covert message sizes: 8, 64, and 128 bits. Figure 5 shows the detection results for the CCTC attack using the different covert message sizes. As shown by the Figure, our approach outperforms the three other approaches: CCE, entropy, and regularity. The ANN classifier trained by SnapCatch achieved 74.33%, 91.36%, and 95.83% for the covert message sizes of 8, 64, and 128 bits, respectively.

As expected, the accuracy of the detection methods is mostly affected by the cover message size. This is because smaller covert messages (e.g., 8 bits) cause less change to otherwise normal traffic behavior. We found that the detection accuracy of SnapCatch gradually increases when the size of covert messages increases.

The second highest detection accuracy was achieved by CCE, which reached detection accuracy of 63%, 76.83%, and 81.66% for the covert message sizes of 8, 64, and 128 bits. The entropy-based approach achieved the third place with the accuracies of 60.33%, 71.17%, and 77.33%, where the regularity-based approach achieved fourth place with detection accuracies of 56.83%, 64.83%, and 72.33% for the covert message sizes of 8, 64, and 128 bits respectively.

3) ULTRA-CAUTIOUS COVERT TIMING CHANNELS (UCCTC)

Our third set of evaluation results shows the performance of our approach for detecting Ultra-Cautious Covert Timing Channels (UCCTC). UCCTC is one of the most advanced cyber-attacks which restricts covert channels from exhibiting behaviors (packet inter-arrival times) that deviate from overt traffic. This restriction often sacrifices the stolen information's quality by force-changing the packets to occur within a reduced time frame. However, this restriction makes this type of cover channels the most difficult to detect due to the high similarity to overt traffic behavior generated by non-malicious applications.

To simulate the UCCTC attack, The packet delay was set to be equal to the quarter of the mean inter-arrival time of overt traffic. We ran the UCCTC attack experiments using the three covert message sizes: 8, 64, and 128 bits. As shown in Figure 6, the highest detection accuracy for the UCCTC attack was achieved by the DT classifier trained by our approach, which laid the detection accuracies of 61.33%, 74.5%, and 76.83% for the covert messages with sizes 8, 64, and 128 bits. The second highest CTC detection was achieved by the CCE, which achieved 58%, 63%, and 66.5%. The entropy approach (third place) achieved accuracy values of 52.83%, 60.33%, and 64.83%, and finally, the regularity approach (fourth place) achieved 52.17%, 57.5%, and 63%, respectively.

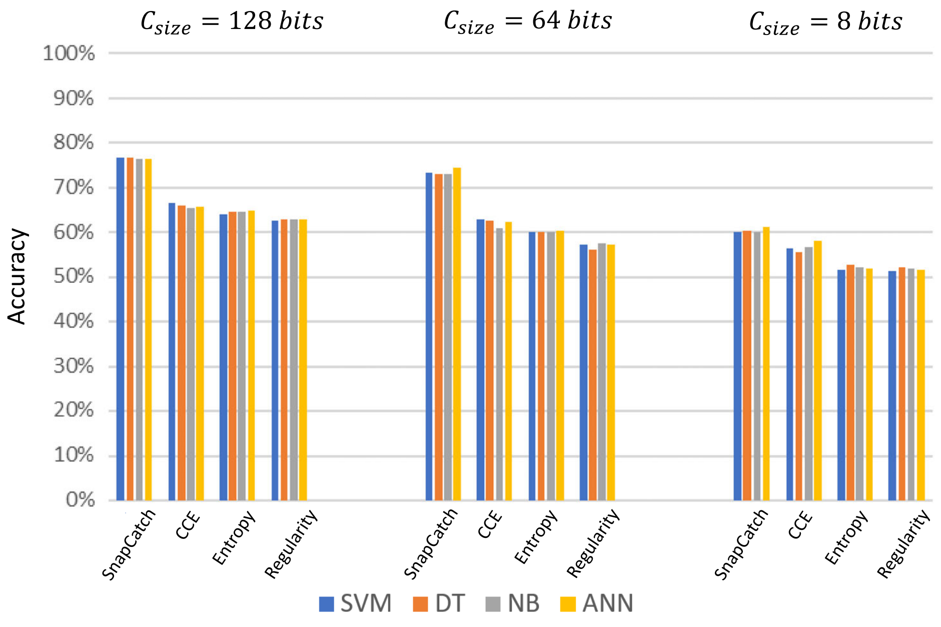


FIGURE 6. The impact of the covert message size on the detection accuracy of Ultra-Cautious CTCs.

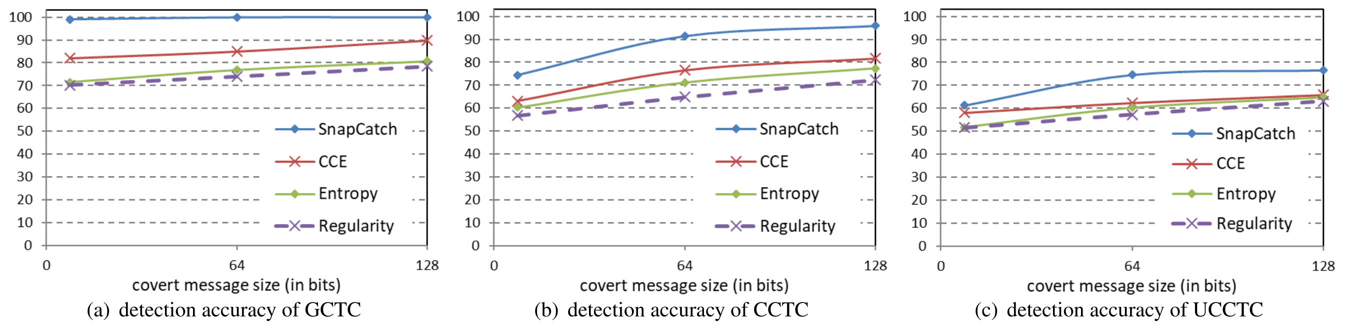


FIGURE 7. Performance of SnapCatch and the other baseline detection methods against varying sizes of covert messages.

Figure 7 shows the accuracy comparison among SnapCatch and the other approaches using the neural network classifier and varying sizes of covert messages, see the appendix for more accuracy measures (e.g, F_1 *score*).

C. PINPOINTING THE LOCATION OF COVERT MESSAGES IN TRAFFIC FLOWS

As explained earlier, detecting covert messages is very important to prevent malicious applications from stealing sensitive data from a given network. Upon successfully detecting covert messages, the traffic flow is dropped to disrupt the malicious data exfiltration process. Although dropping traffic flows that contain covert messages is an effective cyber defense against covert channels, however, it is also very disruptive to the QoS of the overt traffic of legitimate applications that might have most of the packets in the dropped flows. Discovering the segments of traffic flows (i.e., set of packets) that contain the covert message(s) is an extremely important objective as it provides the ability to drop only the malicious part of traffic flows while allowing the rest of the traffic flow to pass through. This precise detection significantly reduces the disruptions of overt traffic initiated by non-malicious applications.

In this part, we investigate and present our approach's performance in pinpointing the location (i.e., the sequence of packets) of covert messages within traffic flows. To do this, we designed an experiment in which the covert messages are injected into all traffic flows in one of three locations (segments): *beginning, middle, and end*. Then, we investigated our approach's performance to find the correct traffic flow segment (beginning, middle, or end) that contained the covert message(s). First, we trained the classifiers using new labels: (1) beginning, (2) middle, and (3) end. Each of these labels corresponds to the relevant location of the covert message in given traffic flow. These labels are generated automatically using the setting of the malicious agent. For example, if the malicious agent was set to inject the covert message

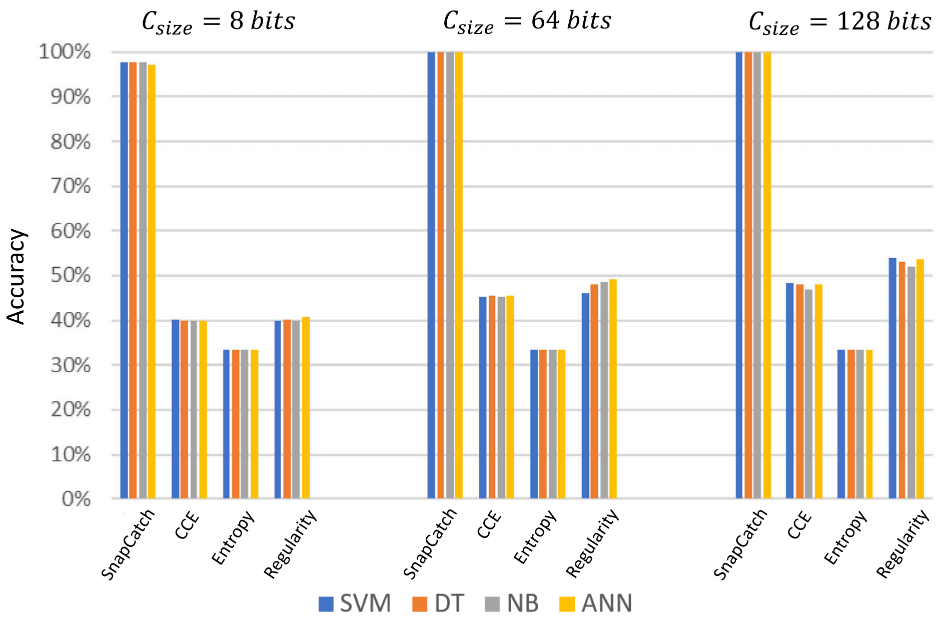


FIGURE 8. The accuracy of pinpointing the location of covert messages within a traffic flow using different message sizes.

into the beginning of traffic, it will also provide the label `beginning` to that traffic flow. Further, it does the same with the other two labels `middle` and `end`. This labeled set of covert traffic flow is then used for training and testing.

We trained the classifiers using 75% of the labeled data, and we use the other 25% for testing. Figure 8 shows the accuracies of detecting the malicious traffic flow segment (beginning, middle, and end) that contains the covert message.

As shown by Figure 8, our approach's SVM classifier successfully detected the segments within traffic flows that contained the covert message with accuracies of 97.83%, 100%, and 100% for covert message size of 8, 64, and 128 bits, respectively. To provide a baseline for comparison, we also ran the CCE, entropy, and regularity. Interestingly, the regularity achieved second place with the detection accuracy of 40.83%, 49%, and 53.83% for the covert message size of 8, 64, and 128 bits. CCE achieved third place with the accuracies 40.17%, 45.55%, and 48.17%. Entropy came in last place with the accuracies of 33.33% for all sizes of covert messages.

V. DISCUSSION

Our study shows that SnapCatch makes an important step toward accurate and automated covert channel detection. This is significant not only for the detection of covert channels but also for precise locating of covert messages within traffic flows which enables thwarting these messages without a significant loss of QoS, as demonstrated by our measurement study. With the rapid increase of cyber attacks that implement and utilize covert channels to exfiltrate sensitive information, the proposed approach can quickly detect CTCs. On the other hand, the proposed approach is still generic and needs to be tuned to adhere to the organization's security mission and constraints. In this section, we discuss the tuning required to achieve the *best* results using SnapCatch.

A. SELECTING THE BEST MACHINE LEARNING CLASSIFIER

The evaluation of SnapCatch shows that it achieves high accuracy and coverage. However, our approach still introduces some false positives (overt traffic being falsely detected) and some false negatives (malicious traffic being falsely missed/allowed). We have experimented with various machine learning classifiers and calculated their results in terms of precision and recall, as shown in Table 3. Different factors maybe are considered when selecting the most appropriate classifier based on the stakeholder defense strategy. For example, selecting the ANN classifier lays the maximum precision value (98.7%) for detecting UCCTC attack with the covert message size of 128 bits. This classifier leans towards accurately detecting only covert time channels at the cost of missing about 47% of the covert channels (false negatives). On the other hand, selecting the ANN classifier lays the maximum recall value (92.3%) for the same attack. This classifier prioritizes detecting the highest number of covert

TABLE 3. Accuracy and coverage comparison of SnapCatch classifiers for UCCTC attack (covert message size = 128 bits).

classifier	Precision	Recall
ANN	98.7%	53%
NB	90.5%	92.3%
SVM	79.7%	72%
DT	73%	37%

TABLE 4. Results of Precision, Recall, and F_1 score of image-based method: GCTC.

		Covert traffic				Overt traffic			
Size	Measure	SVM	DT	NV	ANN	SVM	DT	NV	ANN
8 bits	Precision	1.000	1.000	0.997	1.000	1.000	0.997	1.000	0.984
	Recall	1.000	0.997	1.000	0.980	1.000	1.000	0.997	1.000
	F_1 score	1.000	0.998	0.998	0.995	0.9987	0.998	0.998	0.994
64 bits	Precision	0.999	1.000	0.998	1.000	1.000	0.999	1.000	1.000
	Recall	1.000	0.999	1.000	1.000	0.998	0.999	1.000	0.998
	F_1 score	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
128 bits	Precision	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
	Recall	0.999	1.000	1.000	1.000	1.000	0.999	1.000	1.000
	F_1 score	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

TABLE 5. Results of Precision, Recall, and F_1 score of image-based method: CCTC.

		Covert traffic				Overt traffic			
Size	Measure	SVM	DT	NB	ANN	SVM	DT	NB	ANN
8 bits	Precision	0.703	0.699	0.820	0.963	0.784	0.782	0.776	0.668
	Recall	0.820	0.820	0.333	0.513	0.653	0.647	0.670	0.980
	F_1 score	0.757	0.755	0.474	0.670	0.713	0.708	0.719	0.795
64 bits	Precision	0.902	0.894	0.896	0.905	0.922	0.921	0.922	0.922
	Recall	0.923	0.923	0.925	0.926	0.901	0.891	0.894	0.904
	F_1 score	0.913	0.908	0.910	0.914	0.911	0.906	0.908	0.913
128 bits	Precision	0.946	0.937	0.937	0.972	0.934	0.937	0.94	0.924
	Recall	00.938	0.936	0.940	0.920	0.947	0.938	0.937	0.973
	F_1 score	0.950	0.935	0.938	0.945	0.940	0.939	0.938	0.948

channels at the cost of falsely detecting 9.5% of overt traffic (false positives).

B. BALANCING QoS AND SECURITY

Our study shows that SnapCatch achieves higher accuracies not only in detecting CTCs but also in segmenting traffic flows and pinpointing the correct traffic segment that contains covert messages. SnapCatch outperforms the second-best approach, regularity, by approximately 57%, 54%, and 46% for covert message sizes of 8, 64, and 128 bits. This means that SnapCatch is very effective in detecting a traffic flow segment that contains the covert message which enables precise dropping of that traffic segment rather than the entire flow. This means the loss of QoS caused by dropping entire traffic flows that contain covert messages can be significantly reduced when using our approach by up to 66%. Therefore, with the average accuracy of 99.2% to find the correct traffic flow segment, out of three possible segments containing covert message(s), we believe SnapCatch is the best candidate tool to provide a balance resolve the contention between usability (QoS) and security.

VI. CONCLUSION AND FUTURE WORK

In this article, we present *SnapCatch*, a novel technique for automate and accurate detection of covert timing channels. SnapCatch is designed to specialize image processing and machine learning techniques for covert traffic detection. First, the system converts the the inter-arrival times of traffic into colored images using an innovative mechanism that captures the concrete features of network traffic an represents them in colored images. By extracting robust and accurate features from the colored images, SnapCatch trains various machine learning classifiers to efficiently detect covert channels based on a tunable defense strategy that prioritizes (or balances) accuracy and completeness.

In addition, we propose a mechanism to pinpoint the covert messages (i.e., set of packets) within a traffic flow to allow of dropping only a segment of the traffic flow that contains the covert message rather than the entire flow. Our evaluation of SnapCatch shows that it outperforms the corrected conditional entropy, entropy, and regularity approaches. Further, our approach shows the least performance loss in detecting small (e.g., 8 bits) covert messages and the ultra-cautious covert channels (UCCTC), the most advanced type of covert cyber attacks. SnapCatch vastly outperforms the baseline approaches in detecting the segments within traffic flows that carry covert messages, which significantly reduces the loss of the quality of service caused by dropping covert traffic flows. Finally, we provide various scenarios and use cases for tuning SnapCatch to implement a defense strategy that fits the tool users' resources and security objectives.

A potential direction for further improving SnapCatch is to incorporate the deep learning algorithm, Convolutional Neural Network (CNN) is most commonly utilized to

TABLE 6. Results of Precision, Recall, and F_1 score of image-based method: UCCTC.

		Covert traffic				Overt traffic			
Size	Measure	SVM	DT	NB	ANN	SVM	DT	NB	ANN
8 bits	Precision	0.608	0.550	0.553	0.582	0.983	0.983	0.983	0.685
	Recall	0.560	0.997	0.997	0.807	0.397	0.393	0.193	0.420
	F_1 score	0.585	0.711	0.711	0.676	0.328	0.323 0	.324	0.521
64 bits	Precision	0.653	0.650	0.652	0.738	0.997	0.995	0.996	0.753
	Recall	0.997	0.998	0.996	0.760	0.465	0.458	0.462	0.730
	F_1 score	0.790	0.780	0.789	0.749	0.635	0.628	0.632	0.741
128 bits	Precision	0.797	0.730	0.905	0.987	0.745	0.819	0.922	0.680
	Recall	0.720	0.370	0.923	0.530	0.817	0.840	0.904	0.954
	F_1 score	0.757	0.491	0.914	0.693	0.779	0.831	0.913	0.810

analyzing visual images. CNN takes an input image and assigns importance to the objects (shapes) in the image, which makes it detect one from the other. The pre-processing required in a CNN is much lower compared to other classification algorithms. Moreover, we have a plan to extend the SnapCatch model into a reliable detection model that can detect CTCs over real-time using online machine learning approaches with minimal lag between the start of covert activity and the point of detection.

APPENDIX A

See Table 4–6.

REFERENCES

[1] S. Al-Eidi, O. Darwish, and Y. Chen, "Covert timing channel analysis either as cyber attacks or confidential applications," *Sensors*, vol. 20, no. 8, p. 2417, 2020.

[2] B. Anderson, C. Storlie, and T. Lane, "Improving malware classification: Bridging the static/dynamic gap," in *Proc. 5th ACM Workshop Secur. Artif. Intell.*, 2012, pp. 3–14.

[3] R. Archibald and D. Ghosal, "A comparative analysis of detection metrics for covert timing channels," *Comput. Secur.*, vol. 45, pp. 284–292, Sep. 2014.

[4] A. Askarov, D. Zhang, and A. C. Myers, "Predictive black-box mitigation of timing channels," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, 2010, pp. 297–307.

[5] V. Berk, A. Giani, G. Cybenko, and N. Hanover, "Detection of covert channel encoding in network packet delays," de lUniversité de Dartmouth, Hanover, NH, USA, Rapport Technique TR536, 2005, vol. 19.

[6] A. K. Biswas, D. Ghosal, and S. Nagaraja, "A survey of timing channels and countermeasures," *ACM Comput. Surv.*, vol. 50, no. 1, pp. 1–39, 2017.

[7] K. Borders and A. Prakash, "Web tap: Detecting covert Web traffic," in *Proc. 11th ACM Conf. Comput. Commun. Secur.*, 2004, pp. 110–120.

[8] S. Cabuk, "Network covert channels: Design, analysis, detection, and elimination," Ph.D. dissertation, Purdue Univ., West Lafayette, IN, USA, 2006.

[9] S. Cabuk, C. E. Brodley, and C. Shields, "Ip covert timing channels: Design and detection," in *Proc. 11th ACM Conf. Comput. Commun. Secur.*, 2004, pp. 178–187.

[10] L. Chappell, *Wireshark 101: Essential Skills for Network Analysis Wireshark Solution Series*. San Jose, CA, USA: Laura Chappell Univ., 2017.

[11] Z. Cui, X. Fei, X. Cai, C. Yang, G. G. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3187–3196, Jul. 2018.

[12] O. Darwish, A. Al-Fuqaha, M. Anan, and N. Nasser, "The role of hierarchical entropy analysis in the detection and time-scale determination of covert timing channels," in *Proc. Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2015, pp. 153–159.

[13] O. Darwish, A. Al-Fuqaha, G. B. Brahim, and M. A. Javed, "Using mapreduce and hierarchical entropy analysis to speed-up the detection of covert timing channels," in *Proc. 13th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2017, pp. 1102–1107.

[14] O. Darwish, A. Al-Fuqaha, G. B. Brahim, I. Jenhani, and M. Anan, "Towards a streaming approach to the mitigation of covert timing channels," in *Proc. 14th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2018, pp. 255–260.

[15] O. Darwish, A. Al-Fuqaha, G. B. Brahim, I. Jenhani, and A. Vasilakos, "Using hierarchical statistical analysis and deep neural networks to detect covert timing channels," *Appl. Soft Comput.*, vol. 82, Sep. 2019, Art. no. 105546.

[16] K. Denney, A. S. Uluagac, K. Akkaya, and S. Bhansali, "A novel storage covert channel on wearable devices using status bar notifications," in *Proc. 13th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2016, pp. 845–848.

[17] S. Gianvecchio and H. Wang, "An entropy-based approach to detecting covert timing channels," *IEEE Trans. Dependable Secure Comput.*, vol. 8, no. 6, pp. 785–797, Sep. 2010.

[18] S. Gianvecchio, H. Wang, D. Wijesekera, and S. Jajodia, "Model-based covert timing channels: Automated modeling and evasion," in *Proc. Int. Workshop Recent Adv. Intrusion Detection*. Berlin, Germany: Springer, 2008, pp. 211–230.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explor. Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.

[20] K. Han, J. H. Lim, and E. G. Im, "Malware analysis method using visualization of binary files," in *Proc. Res. Adapt. Convergent Syst.*, 2013, pp. 317–321.

[21] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, May/Jun. 2007.

[22] F. Iglesias, R. Annessi, and T. Zseby, "DAT detectors: Uncovering TCP/IP covert channels by descriptive analytics," *Secur. Commun. Netw.*, vol. 9, no. 15, pp. 3011–3029, 2016.

[23] F. Iglesias, V. Bernhardt, R. Annessi, and T. Zseby, "Decision tree rule induction for detecting covert timing channels in TCP/IP traffic," in *Proc. Int. Cross-Domain Conf. Mach. Learn. Knowl. Extraction*. Cham, Switzerland: Springer, 2017, pp. 105–122.

[24] F. Iglesias and T. Zseby, "Are network covert timing channels statistical anomalies?" in *Proc. 12th Int. Conf. Availability, Rel. Secur.*, 2017, pp. 1–9.

[25] M. H. Kang, I. S. Moskowitz, and S. Chincheck, "The pump: A decade of covert fun," in *Proc. 21st Annu. Comput. Secur. Appl. Conf. (ACSAC)*, 2005, p. 7.

[26] S. S. Kim and A. N. Reddy, "Modeling network traffic as images," in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 1, May 2005, pp. 168–172.

[27] S. S. Kim and A. L. N. Reddy, "A study of analyzing network traffic as images in real-time," in *Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, vol. 3, Mar. 2005, pp. 2056–2067.

[28] N. Kiyavash, F. Koushanfar, T. P. Coleman, and M. Rodrigues, "A timing channel spyware for the CSMA/CA protocol," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 3, pp. 477–487, Mar. 2013.

[29] X. Liu, Y. Sun, L. Fang, J. Liu, and L. Yu, "A survey of network traffic visualization in detecting network security threats," in *Proc. Int. Conf. Trustworthy Comput. Services*. Berlin, Germany: Springer, 2014, pp. 91–98.

[30] Y. Liu, D. Ghosal, F. Armknecht, A.-R. Sadeghi, S. Schulz, and S. Katzenbeisser, "Hide and seek in time—Robust covert timing channels," in *Proc. Eur. Symp. Res. Comput. Secur.* Berlin, Germany: Springer, 2009, pp. 120–135.

[31] J.-S. Luo and D. C.-T. Lo, "Binary malware image classification using machine learning with local binary pattern," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 4664–4667.

[32] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[33] A. Makandar and A. Patrot, "Malware analysis and classification using artificial neural network," in *Proc. Int. Conf. Trends Automat., Commun. Comput. Technol. (I-TACT)*, 2015, pp. 1–6.

[34] A. Makandar and A. Patrot, "Wavelet statistical feature based malware class recognition and classification using supervised learning classifier," *Oriental J. Comput. Sci. Technol.*, vol. 10, no. 2, pp. 400–406, 2017.

[35] M. Mehic, J. Slachta, and M. Voznak, "Whispering through DDoS attack," *Perspect. Sci.*, vol. 7, pp. 95–100, Mar. 2016.

[36] S. Mou, Z. Zhao, S. Jiang, Z. Wu, and J. Zhu, "Feature extraction and classification algorithm for detecting complex covert timing channel," *Comput. Secur.*, vol. 31, no. 1, pp. 70–82, 2012.

[37] H. Najadat, Y. Jaffal, O. Darwish, and N. Yasser, "A classifier to detect abnormality in ct brain images," in *Proc. Int. Multiconf. Eng. Comput. Sci.*, 2011, pp. 16–18.

[38] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath, "Malware images: Visualization and automatic classification," in *Proc. 8th Int. Symp. Vis. Cyber Secur.*, 2011, pp. 1–7.

[39] R. Paul, S. H. Hawkins, L. O. Hall, D. B. Goldgof, and R. J. Gillies, "Combining deep neural network and traditional image features to improve survival prediction accuracy for lung cancer patients from diagnostic CT," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 2570–2575.

[40] P. Peng, P. Ning, and D. S. Reeves, "On the secrecy of timing-based active watermarking trace-back techniques," in *Proc. IEEE Symp. Secur. Privacy (S&P)*, May 2006, p. 15.

[41] W. S. Rasband. (2009). ImageJ. US National Institutes of Health, Bethesda, MD, USA. [Online]. Available: http://rsb.info.nih.gov/ij/

[42] S. S. Sarikan and A. M. Ozbayoglu, "Anomaly detection in vehicle traffic with image processing and machine learning," *Procedia Comput. Sci.*, vol. 140, pp. 64–69, Jan. 2018.

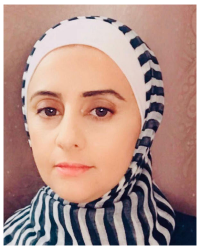
[43] P. L. Shrestha, M. Hempel, F. Rezaei, and H. Sharif, "Leveraging statistical feature points for generalized detection of covert timing channels," in *Proc. IEEE Mil. Commun. Conf.*, Oct. 2014, pp. 7–11.

[44] P. L. Shrestha, M. Hempel, F. Rezaei, and H. Sharif, "A support vector machine-based framework for detection of covert timing channels," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 2, pp. 274–283, Apr. 2015.

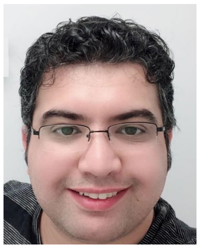
[45] T. Sohn, J. Moon, S. Lee, D. H. Lee, and J. Lim, "Covert channel detection in the icmp payload using support vector machine," in *Proc. Int. Symp. Comput. Inf. Sci.* Berlin, Germany: Springer, 2003, pp. 828–835.

[46] T. H. A. Soliman, R. Mohamed, and A. A. Sewissy, "A hybrid analytical hierarchical process and deep neural networks approach for classifying breast cancer," in *Proc. 11th Int. Conf. Comput. Eng. Syst. (ICCES)*, 2016, pp. 212–219.

[47] S. Zander, G. Armitage, and P. Branch, "Stealthier inter-packet timing covert channels," in *Proc. Int. Conf. Res. Netw.* Berlin, Germany: Springer, 2011, pp. 458–470.

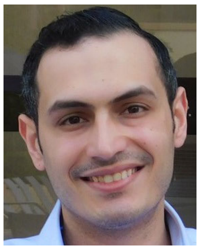


SHOROUQ AL-EIDI received the B.Sc. degree in computer science from Mutah University, Jordan, in 2006, and the M.Sc. degree in computer science from the Jordan University of Science and Technology, Jordan, in 2012. She is currently pursuing the Ph.D. degree with the Computer Science Department, Memorial University of Newfoundland. After obtaining her master's degree, she has worked as a full-time Lecturer with the Computer Science Department, Tafila Technical University, Jordan. Her current research interests include cyber security, information security, complex networks, and machine learning.



OMAR DARWISH received the M.S. degree from the Jordan University of Science and Technology, Jordan, and the Ph.D. degree in computer science from Western Michigan University, USA. He has worked as a Visiting Assistant Professor with the West Virginia University Institute of Technology, a Software Engineer with MathWorks, and a Programmer with the Nuqul Group. He is currently an Assistant Professor, a Program Coordinator of computer technology and information systems, and the Director of the IoT and Cybersecurity Laboratory, Ferrum College, a four-year college, USA. His research interests include cyber security, machine learning, networks, big data analysis, cloud computing, artificial intelligence, data mining, and information retrieval.

YUANZHU CHEN received the B.Sc. degree from Peking University, China, in 1999, and the Ph.D. degree from Simon Fraser University, Canada, in 2004. He has been a Professor of computer science since 2005 and currently serves as the Head of the Department. From 2004 to 2005, he was a Postdoctoral Researcher with Simon Fraser University. In 2005, he joined Memorial University as a tenure-track Assistant Professor. While at Memorial, he was the Deputy Head for Undergraduate Studies from 2012 to 2015 and the Deputy Head for Graduate Studies from 2016 to 2019. His research interests include complex networks, computer networking, online social networks, mobile computing, graph theory, Web information retrieval, and evolutionary computation, with funding from national agencies and various university programs and awards. He was a recipient of the President's Award for Distinguished Teaching in 2018.



GHAITH HUSARI received the Ph.D. degree from the University of North Carolina at Charlotte, in 2019. He is currently an Assistant Professor with the Department of Computer Science, East Tennessee State University. His research interests include cybersecurity and privacy, big data analytics for cyber threat intelligence, and security analytics and automation.