IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 October 2023, accepted 23 December 2023, date of publication 29 December 2023,
date of current version 10 January 2024.

Digital Object Identifier 10.1109/ACCESS.2023.3348451

RESEARCH ARTICLE

Addressing Behavioral Drift in Ransomware Early Detection Through Weighted Generative Adversarial Networks

UMARA UROOJ[1], BANDER ALI SALEH AL-RIMY[1], (Senior Member, IEEE), ANAZIDA BINTI ZAINAL[1], (Member, IEEE), FAISAL SAEED[2], ABDELZAHIR ABDELMABOUD[3], AND WAMDA NAGMELDIN[4]

[1]Department of Computer Science, Faculty of Computing, Universiti Teknologi Malaysia, Johor Bahru, Johor 81300, Malaysia
[2]DAAI Research Group, School of Computing and Digital Technology, College of Computing and Digital Technology, Birmingham City University, B4 7XG Birmingham, U.K.
[3]Department of Information Systems, King Khalid University, Muhayil 61913, Saudi Arabia
[4]Department of Information Systems, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia

Corresponding authors: Bander Ali Saleh Al-Rimy (bander@utm.my) and Umara Urooj (umaraurooj@gmail.com)

This works funded by the Deanship of Scientific Research at King Khalid University through large group Research Project under grant number (RGP.2/175/44).

ABSTRACT Crypto-ransomware attacks pose a significant cyber threat due to the irreversible effect of encryption employed to deny access to the data on the victim's device. Existing state-of-the-art solutions are developed based on two assumptions: the availability of sufficient data to perform detection during the pre-encryption phase, and that ransomware behavior is static and does not change over time. However, such assumptions do not hold as data collected during the pre-encryption phase of the ransomware attack are limited and does not contain sufficient patterns needed to identify the attack. Additionally, the evasion techniques like polymorphism and metamorphism used by ransomware lead to behavioral drift that could defeat those solutions. Therefore, this paper addresses these two issues by proposing a weighted Generative Adversarial Networks (wGANs) technique. Firstly, the proposed wGAN was used to generate synthetic data that imitate the behavior of ransomware and simulate the evolution of the attacks. Then, the mutual information was used to estimate the significance of features for different timeframes, thereby helping the detection model to handle the behavioral drift in emerging ransomware variants. Experimental evaluation demonstrates that the proposed wGAN is more robust against behavioral drift compared to the state-of-the-art solutions. The wGAN achieved higher accuracy and lower false alarm rates of 97% and 0.0088 respectively.

INDEX TERMS Adaptive, crypto-ransomware, early detection, generative adversarial networks, metamorphic, polymorphic, ransomware, ransomware prediction.

I. INTRODUCTION

The concept of cryptography was employed to develop a new type of malware named Ransomware [1], [2] which encrypts the data on the victim's device and demand ransom in return. The use of an encryption mechanism makes the ransomware attacks irreversible if the decryption key is not available [3], [4]. Ransomware attacks target both individuals and organizations on different platforms and systems [2]. In most cases, victims tend to pay the ransom due to fear of losing their reputation and potential customers. An article by BBC published on July 05, 2021, reported that 70 million ransom was demanded due to colossal ransomware attack. The attackers initially targeted US IT firm Kaseya and then spread through corporate networks which affected over one million victims from supermarkets and schools [5]. The attacker usually attacks an individual, who has little knowledge of computer usage, and then plays with his fear of data loss or disclosure, i.e., personal data, browser history, contacts, and chats [6], [7], [8], [9]. The use of crypto-currency also

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

fuels ransomware attacks as it offers a safe and non-traceable method to pay the ransom [10].

Ransomware attacks have been evolving over time and new variants will likely be generated in the future. This is because of the availability of many ready-to-use toolkits like RaaS, which makes it easy to generate ransomware mutants [11], [12]. Polymorphism and metamorphism strategies adopted by ransomware to evade detection help to increase these attacks [13], [14]. Research by Webroot showed that ''94% of malicious executables are polymorphic'' [12]. Polymorphism and metamorphism are the base of behavioral drifting of ransomware attacks. The behavioral drift is defined in terms of the development of new attack variants with changing behavior over time [15], [16]. Behavioral drift is the basis for zero-day attacks [17], by changing the attack attributes (which are also called features) [6], [14], [18].

For developing variants, changing distribution of features changes the significance of features accordingly. Polymorphic and metamorphic ransomwares try to evade detection hence should be dealt with more vigilance [19]. The polymorphic adversarial attacks are generated with the help of Generative adversarial networks. This is due to the ability of GANs to generate real-like samples from reference data [13], [20].

The problem of ransomware detection becomes more challenging when dealing with polymorphic and metamorphic attacks [21], [22]. These attacks spread rapidly and can encrypt the files in a very short time, hence early detection becomes imperative. Early detection is also important as once the ransomware encrypts the files, all the data becomes inaccessible, even if the ransomware is detected and removed [2], [21], [23]. An effective early detection solution for these evolving ransomware variants could be developed by addressing two intertwined challenges include:

The first challenge for ransomware early detection is the availability of enough data during the pre-encryption phase as the attack is still in the preparation phase and the actual attack is yet to take place. Therefore, it is important to address such data insufficiency as sufficient patterns are required for accurate early detection. Data augmentation is a popular method in machine learning-based solutions and has a potential as a solution for such data insufficiency challenge that malware and ransomware early detection solutions suffer from. To the best of our knowledge, here is no single study dealing with data augmentation during pre-encryption phase of ransomware attacks.

The polymorphic nature of ransomware introduces another challenge concerning the relevance of attack features used by existing solutions when developing and training detection models. For instance, a ransomware attack at time t_1 could display an attack pattern (features) p_1 whose relevance is greater than the same pattern displayed by another ransomware variant attack at time t_2 with attack pattern (features) p_2. This implies that the significance of features could change over time for different ransomware variants. The availability of attack patterns and their significant features are linked to a specific timeframe (t) for different ransomware variants. However, existing ransomware early detection solutions assume the significance of features is static and unchanging, leading to a behavioral drift that causes the developed solutions to become outdated quickly, thereby reducing the detection accuracy over time.

To this end, this paper is dedicated to addressing the two issues mentioned above incorporating an improved Generative Adversarial Network (GAN)-based data augmentation technique that compensates for data insufficiency during the pre-encryption phase of the attack. The proposed technique also re-weighs the features' relevancy based on the behavior progression of the new ransomware characteristics, which addresses the behavior drift of the attack. The contribution of this paper is three-fold as follows:

1) A TF-IDF and GAN-based pre-encryption data augmentation technique is proposed to address the problem of data insufficiency during the pre-encryption phase by generating adequate synthetic data.
2) A weighted GAN (wGAN) technique is proposed, which incorporates the mutual information method into the GAN structure to estimate the significance of features for different timeframes, thereby helping the detection model to handle the behavioral drift seen in emerging ransomware variants.
3) Extensive experiments were conducted to evaluate the performance of the techniques developed in (1) and (2).

The rest of the paper is organized as follows; In section II, the existing work was discussed. The methodology of the proposed work is discussed in section III. Experimental results are presented in section IV. In section V, the analysis and discussion of the experimental work and results were explained. The proposed work is concluded in section VI.

II. RELATED WORKS

The early detection of an attack helps to reduce or stop the effects of the attack in the first place. Early detection systems detect ransomware before the encryption of data starts. These detection systems have more worth than traditional detection systems as they ensure the prevention of user data from encryption attacks [24], [25].

A. EARLY DETECTION OF RANSOMWARE ATTACKS

Some of the studies are available in the literature that claimed to provide early detection for ransomware attacks. These early detection studies are briefly discussed here.

An early detection system was presented by [26], which utilized the real logs from Security Operation Centers (SOC) to perform detection. The performance of seven different machine learning classifiers was evaluated on given data. This detection system extracted patterns and performed detection before the encryption process started. A ransomware preventive and early detection system was proposed by [27]. This system was able to perform detection for known and zero-day attacks. The system performed detection by using two techniques. It compared the signature in one phase and performed the dynamic analysis in the second phase. This system showed high false alarms and hence became limited in its implementation. Another early ransomware detection system was proposed by [28], which utilized the system APIs as a base to uncover the ransomware attack. This detection system classified samples into three categories, i.e., good ware, malware, and ransomware. The work was an extension of its previous work, R-PackDroid, while focusing on the use of system API packages, classes, and methods.

An alert-based ransomware detection model was proposed by [29], which monitored the network traffic to detect ransomware. The system focused on the packet level inspection to trace the maliciousness of traffic. However, the system would generate high false alarms if an encryption operation was initiated by a benign program. A scheme was proposed by [19] which dealt with pre-encryption boundary definition and feature extraction using pre-encryption APIs. A detection model was built on the features extracted from the pre-encryption phase using the aTF-IDF technique. The study proposed using a dynamic threshold to detect ransomware in the early phase of the attack. PEDA was proposed by [30] as an early detection system for ransomware attacks. The system performed both signature-based detection and behavioral detection. The system utilized random forest for decision-making. The proposed system was limited to perform classification and was able to detect only the ransomware it trained on. This system was not fully functional to detect the whole ransomware; instead, it was developed as a supplement to the detection system.

A study in [31] presented an early detection system for crypto-ransomware. The early detection model was implemented using the machine learning classifier Random Forest and evaluated using six different metrics. The system consists of two modules, i.e., signature-based detection and behavioral detection, to perform pre-encryption detection. DeepRan, an early detection system for the bare metal server was developed in [32]. It was an anomaly-based detection system, and a profile was built by using the logs of regular activities on the host machines. The system was implemented by using deep learning models. The system was limited in working as the profile was built using the dedicated network host machines. RanStop, an early ransomware detection system, was proposed by [33] that performed detection in 2ms. This system performed detection by utilizing the hardware assistance and detected the ransomware in runtime. This system was limited in its implementation due to the requirement to access different hardware. An early detection system for ransomware was developed in [34], which utilized both machine and deep learning techniques. This detection system utilized both analysis techniques to perform the detection. Features were extracted from I/O activities and the file entropy method. Samples were executed for five minutes in the bare-metal sandbox. Robustness was maintained by extracting time series data from the samples' execution.

An early detection system CRED concept was proposed by () [35]. This study intended to use a combination of API and IRP to detect ransomware in its early phases. However, this study only proposed a concept and was not implemented. This study was also limited due to the use of a timeline to capture I/O Request Packets. An intrusion detection honeypot (IDH) acting as early warning was proposed in [36] to deal with the integration problem of intrusion detection systems and anti-viruses. This system produced early warnings for encountered ransomware by implementing the social leopard algorithm (SoLA). This IDH also utilized the SDN to monitor suspicious communication. In another work, the use of file access control to control the occurrence and spread of ransomware attacks on the windows platform was presented [37]. It was an early detection system that raised early warnings for the user to restrict the access of files to specific resources. It was a dynamic approach to perform detection by focusing on access to files. A ransomware detection method was proposed in [38], which detected the ransomware by extracting hexadecimal codes from its samples. Hexacodes were extracted by using binary decoders. It was an early detection system that utilized hexacodes of ransomware as opposed to the use of opcodes.

An early detection strategy was proposed in [39] to detect the new android ransomware families. Distinctive features of different families were identified and used as a basis to characterize ransomware samples. However, it was an empirical study due to the use of static analysis. Minerva, a ransomware detection approach was presented in [40] which observes all the operation performed on a file during a specific time. This work was limited as it can raise false alarms by focusing on the encryption of files that could be encrypted by benign programs. In [41] a decoy-based ransomware detection framework, named RTrap was developed. This work lacks performance due to the use of decoy techniques which can be detected by advanced ransomware attacks.

B. ADAPTIVE DETECTION FOR RANSOMWARE ATTACKS

An adaptive detection system is required to deal with developing variants of ransomware. Some studies tried to develop adaptive models with limitations. An adaptive ransomware detection framework was proposed by [42], which combined the supervised and unsupervised approaches to get adaptiveness. It was an anti-obfuscation model that extracted features from runtime data. This semi-supervised model was implemented by using deep learning. An adaptive ransomware detection approach was proposed in [43] which acts as a supplement to the network defense system. This approach was implemented using open-source softwares and only tested on WannaCry and Petya ransomware. This detection approach could suspend the network traffic for encryption performed by benign programs. Another adaptive pre-encryption model was proposed by [22] and considered the population drift concept for ransomware attacks. The study discussed the problem of dealing with the little amount of data available on hand before encryption. However, the implementation of the study was not discussed in the study. The work in [44] proposed a network-based ransomware detection scheme named SINN-RD. This study focused on the generation of new features from log files.

C. POPULATION DRIFT OF RANSOMWARE ATTACKS

Developing variants are another form of polymorphic and metamorphic ransomwares. These two mechanisms keep on developing zero-day ransomware variants hence introducing the concept of population drift for ransomware attacks. Some of the studies discussed the development of future ransomwares. Population drift is a concept that discussed the change in the distribution of features and characteristics of malwares and ransomwares. This concept focused on developing new variants with time [14], [18]. A study in [18] designed a framework for android malware that focused on behavioral traits to detect malware. It was an online learning detection system that dealt with the population drift concept of malware. The system provided adaptiveness and context-awareness using the graph kernel and online-learning. An analytical study was performed by [12] to predict the future types of ransomwares, i.e., poly and metamorphic ransomwares. Different future strains of ransomwares were created and run in the cuckoo to perform the analytical analysis to make predictions. The analytical analysis helped to analyze the behavior, attack process, and network-level analysis of subjected ransomwares.

D. ROLE OF GAN IN RANSOMWARE DETECTION

Generative Adversarial Networks were employed to generate as well as detect malware attacks. The studies that use GAN for attack generation and detection are briefly discussed here. Malware detection for zero-day attacks was performed by [45] using a transferred generative adversarial network (tGAN). This work proposed a method of pre-train GAN with an autoencoder to address the unstable training problem of GAN. Transfer learning was used with the GAN generator to stabilize the GAN training and control production of the nonsensical output of GAN. A study by [13] produced polymorphic adversarial DDoS attacks by using GAN to show the vulnerability of the machine learning-based defensive system. The study aimed to make AI-based IDS better and more effective by ripping off the advantages of GAN. The study was limited due to the manual updating of attack profiles. In different studies, generative adversarial networks were used for malware detection. Initially, GANs dealt with image data so some of the studies were trained on image data to detect ransomware and later some of the studies were trained on tabular data. In (MIT) Generative adversarial networks were used to detect malware and find another application to generate synthetic tabular data. A study of [46] generated two versions of GANs to be operated on the tabular data. The model worked with the generation of samples and added fairness constraints to the generated samples. This study proposed two GAN versions TabFairGAN and WassersteinGAN, to produce high quality and fair tabular data. Another android malware detection system was proposed by [47], which utilized the generative adversarial network for ransomware detection. GAN was used for training, analyzing, and making predictions about malware. This work was limited in its implementation as it performed the static analysis, and GAN was trained on the bytecodes of android APKs.

A summary of considered studies is presented in Table 1. Limitation of these studies are discussed hereafter. The mentioned solutions to detect ransomware attacks lack performance for developing variants. Some of the existing studies employed decoy techniques, hence did not ensure complete prevention from ransomware attacks due to the loss of some fraction of data compromised. File-based detection systems involved high false alarms due to misclassification between benign and ransomware encrypted files. Existing solutions did not consider the concept of continuously evolving variants of ransomware which using sophisticated executing mechanisms and evasion techniques thus result in misclassification of benign and ransomware programs and generating high false alarms. Existing solutions for prevention and detection did not fully address the mentioned challenges. All the available solutions are limited in scope as they neither considered sufficiency of pre-encryption data nor adaptiveness for developing variants depicting behavioral drift. An adaptive solution is required to combat the problem of data insufficiency and behavioral drift adopted by developing variants of ransomware attacks.

III. METHDOLOGY

This section explains the adopted methodology and experimental setup to develop the proposed technique. Brief descriptions of the used dataset, methods, and techniques are presented hereafter.

A. DATASET

This work utilized the pre-encryption dataset generated by study [19]. The dataset was generated by using both benign and ransomware samples. The ransomware samples were downloaded from VirusShare and consisted of 39378 samples, while benign samples were downloaded from www.informer.com and consisted of 16057 samples. The pre-encryption dataset was generated from the runtime data of dynamic analysis performed in the Cuckoo Sandbox on a collection of ransomware and benign samples. Pre-encryption features were extracted from the separate log files of ransomware and benign samples.

B. EXPERIMENTAL SETUP

To detect the polymorphic and metamorphic ransomwares, the behavior of the discovered samples should be monitored in a controlled environment to avoid obfuscation.

TABLE 1. State-of-the-art approaches for ransomware detection.

Studies	Population Drifting	Early Detection	Use of synthetic Features
[26]		✓	
[27]		✓	
[45]			✓
[28]		✓	
[29]		✓	
[42]	✓		
[19]		✓	
[30]		✓	
[31]		✓	
[13]	✓		✓
[32]		✓	
[43]	✓		
[18]	✓		
[33]		✓	
[34]		✓	
[46]			✓
[35]		✓	
[36]		✓	
[22]	✓	✓	
[37]		✓	
[38]		✓	
[39]		✓	
[12]	✓		✓
[40]		✓	
[41]		✓	
[44]	✓		✓
[47]			✓

Analysis of polymorphic and metamorphic ransomwares is better performed with dynamic analysis [14]. The dataset utilized here was generated by the experimental environment built-in [19]. A Cuckoo sandbox was used to carry out the dynamic analysis. Cuckoo provides a realistic yet isolated environment to monitor malicious codes, i.e., malware and ransomware. It is a widely used open-source automated malware analysis platform to get detailed reports of the runtime behavior of the subjected files [48] so that we can generate synthetic data. Most of the studies utilized the Cuckoo to analyze the crypto-ransomware behavior [49]. The cuckoo was installed in VMware to create realistic and virtualized architecture to get the actual behavior of ransomware samples [50]. Separate trace files were generated for each ransomware and benign sample as both samples were executed separately. The trace files contained all the APIs called during the underwent analysis in Cuckoo. The extracted APIs were used to build the synthetic data in the proposed work. An overview of the experimental flow to give an idea about the dataset generation is visualized in fig 1.

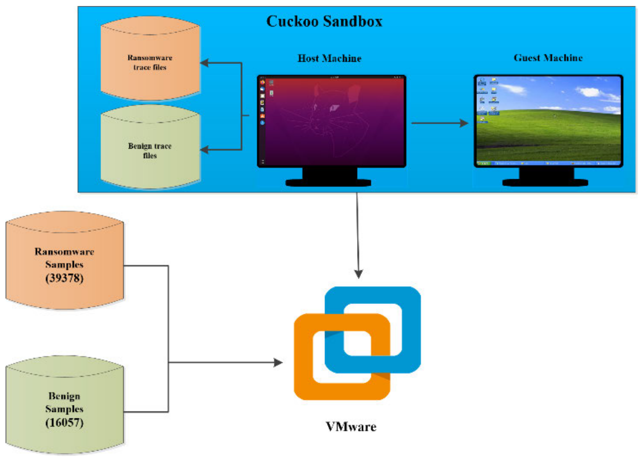

FIGURE 1. The Experimental flow for the behavioral data logs generation.

1) PROCESSING TOOLS

To implement the proposed work, all the analysis and results generation was performed using the Python 3.7 libraries Scikit-learn, Pandas, and Numpy. These libraries have built-in high-level mathematical functions, data preparation and extraction, and data analysis tools [51]. Keras and TensorFlow frameworks were used to implement the deep learning models, i.e., GAN. Some heavy computational tasks were also performed with the help of GPUs.

C. GENERATION OF THE PRE-ENCRYPTION DATASET

This section explains the whole process adopted in work [19] by which the pre-encryption dataset was generated. The dynamic analysis was performed on all the downloaded samples from VirusShare and informer. All the samples were run inside the controlled environment depicted in fig 1. The runtime data generated by each sample in the sandbox was captured in a separate trace file, which included all the API calls and their parameters and unnecessary data associated with that specific sample. The guest machine hooked each process with the help of a sandbox agent. After each sample execution, the guest machine was reverted, so each next sample generated its own set of API calls and parameters without any infection from the previous sample execution. All the unnecessary data in the trace files was discarded, and the final pre-encryption dataset contains the cryptographic API calls along with their parameters. The extracted dataset contained pre-encryption APIs, filtered from the encryption process by implementation of aTF-IDF technique. The dataset contained the data from the first cryptography API encountered until the start of the attack execution. The dataset has a limited amount of data as a fraction of the data was available during the pre-encryption phase.

D. MODULE 1: PRE-ENCRYPTION DATA INSUFFICIENCY

The problem of pre-encryption data insufficiency is addressed by this module. This module is further splits up into two parts discussed hereafter.

1) TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY (TF-IDF) FOR PRE-ENCRYPTION FEATURES EXTRACTION

The dataset used in this work was built in [19] by the execution of ransomware and benign samples in the cuckoo sandbox. The JSON files extracted from the dynamic analysis have textual data. The TF-IDF helps to find out the frequency of APIs and extract features from textual data. In this work, TF-IDF is applied to determine the frequency of each API called during pre-encryption phase [4]. The textual format is converted into discrete data to be fed to the GAN. The general formula to calculate the TF-IDF value of the subjected APIs is given in (1).

$$w(api_k^j) = tf(api_k^j). \log \frac{N}{idf(api_k)} \tag{1}$$

In this equation, $tf(api_k^j)$ represents the frequency of an API api_k called by the instance in a subset, while $idf(api_k)$ represents the number of instances in a subset. The total number of instances is represented with N.

2) PRE-ENCRYPTION DATA AUGMENTATION

The pre-encryption phase of ransomware attacks generates a little amount of data. In this section, the problem of data limitation during the pre-encryption phase is considered and addressed. Production of real-like data is a complementary nature of GAN. The ability of GAN finds application in malware generation by generating synthetic data similar in distribution as fed samples.

The generative adversarial networks (GAN) is a data synthesizer used to generate real-like synthetic data. It consists of two neural networks $G(x)$ and $D(x)$ that play adversarial games to generate real-like fake samples by fooling each other. These networks produce the optimal solution by playing a min-max game. GAN produced the same joint probability distribution as its input has [20]. A discriminator gets input from two different sources i.e., one from real data and the other from a generator then it performs discrimination between real and fake samples. The real data is referred to as positive samples and generator's generated data is referred to as negative samples to the discriminator.

In the discriminator training process, a discriminator is connected to two loss functions. The discriminator network calculates the loss function at the end of classification. This loss function is then backpropagated to the generator and discriminator. The discriminator loss is used in the training process of the discriminator. A discriminator loss penalizes the discriminator if a real sample is classified as fake or vice versa. At the time of training, both the generator and discriminator trained one after another while at the time of backpropagation both the generator and discriminator network worked together. The generator is responsible for generating the fake samples while the discriminator produced labels. The loss function of both networks is different and differs in sign. The discriminator network classifies the data by using the binary classification and sigmoid function and produces the output ranges from zero to one.

The generator network is intended to increase the probability of the discriminator network so that it misclassifies the fake data. The generator is responsible for generating data that looks real as the fed samples and cannot be detected by the discriminator. It simulates by incorporating the periodic feedback from the discriminator. The generator takes input from the random noise which is a fixed-length random vector and feedback from the discriminator to generate the data instances i.e., fake samples. The generator uses noise to produce a wide variety of meaningful output, i.e., fake samples. These fake samples are then trained on the discriminator. A separate database is set up to store the real and fake samples. The backpropagation method helps to generate the gradient which helps the generator to generate the weights. This method also calculates the impact of weights on the output and is responsible for the weight's adjustment. Weights were adjusted by both networks i.e., generator and feedback from discriminator.

The discriminator tries to maximize the discriminator network $D(x)$ by approaching the value 1 while generator tries to minimize $D(x)$ by approaching 0. The discriminator identifies the real samples from the training set as real if $D(x)=1$ else it identifies the generated samples as fake when $D(x)=0$. The generator tries to generate samples that are classified as real and produced 1 for the discriminator [13], [52]. Discriminator tries to maximize, and Generator tries to minimize the following function.

$$V(D,G)=E_{x\sim Pdata^{(x)}}[\log D(x)]+E_{z\sim P_{Z}^{(z)}}[\log(1-D(G(z)))] \tag{2}$$

In the equation above G, D, and x represent Generator, Discriminator, and data respectively. E_x represents the expected value over all the real samples. $D(x)$ represents the Probability Estimation that the real sample is real. The expected value over all random inputs to the generator or expected value over all the generated fake samples is represented by E_z. $G(z)$ is the Generator network, Generator output is generated over Noise (z) input while $D(x)$ is the Discriminator network, Probability Estimation that fake sample is fake. $D(G(z))$ is discriminator estimation that fake sample is real.

$$[\log D(x)] \tag{3}$$
$$[\log(1-D(G(z)))] \tag{4}$$
$$[\log D(x)+\log(1-D(G(z)))] \tag{5}$$

Discriminator recognizes the real data and fake data by using the functions of (3) and (4) respectively. The generator has no access to logD(x) hence cannot affect it directly instead it affects the term log(1-D(G(z))) by minimizing it or maximizing logD(G(z)). Discriminator assigns probability P(x) = 0 to real samples and P(x) = 1 to fake samples. Equation (5) combines both (3) and (4) and represents the working of generator and discriminator. Generator tried to minimize the function of (4) while discriminator tries to maximize the function of (5).

By using the complementary nature of GAN, the first module added more pre-encrypted synthetic data to overcome the problem of data limitation in the early phases of ransomware attacks. The potential pre-encryption synthetic data was generated by ripping off the advantages of using GAN. After applying TF-IDF on pre-encryption data, it was fed into the GAN. It generated real-like pre-encrypted patterns to deal with the data limitation problem of the pre-encryption phase. This module acts as a supplement to the pre-encryption data.

E. MODULE 2: SIGNIFICANCE OF FEATURES

The problem of significance of features to detect the polymorphic and metamorphic ransomware is addressed by this module. This module is further divided into two parts discussed hereafter.

1) THE SIGNIFICANCE ESTIMATOR

To make a model adaptive to the behavioral drift of polymorphic and metamorphic ransomwares, it should be trained on changing features. The behavioral drift concept is tied to deal with significant features for different timeframes. The significance of a feature can have a lower value at a given time t_1, but it can be significant at a different timeframe t_2 in the future. The significance of the features keeps on changing for the evolving variants of ransomware. To deal with the behavioral drift of different variants of ransomware developing in different timeframes (t_1, t_2, t_3, …) we must consider the set of all significant features at a time (t). A mechanism is required to determine the significance of the features while considering all the features at a time. Hence, to estimate the significance of a feature it should be assigned weight accordingly. The significance estimator emphasizes the significance of each feature and helps to reduce the uncertainty. The significance of each feature x_1, x_2, x_3, x_4, etc. with target y_1, y_2, y_3, y_4 etc. is calculated according to (6). It is a mutual relationship of a feature with its target of being ransomware and benign. It describes the dependency between two random variables i.e., feature and target.

$$\begin{aligned} u = I(X;Y) &= H(X) - H(X|Y) \\ &= \sum_{y\in Y}\sum_{x\in X} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \end{aligned} \tag{6}$$

The u is the mutual information of each feature with target. The terms $p(x)$ and $p(y)$ represent the marginal distributions, while *p(x,y)* represents the joint distribution. X and Y will be independent if they produce a mutual information value of zero. This relationship is expressed in expression (7).

$$I(X;Y)=0 \tag{7}$$

The term $I(X;Y)$ is a non-negative number and represents the values about relationship of two random variables. The entropy H(X) is calculated using (8).

$$H(X) = -\sum_{x_i\in X} p(x_i)\log(p(x_i)) \tag{8}$$

The conditional entropy H(X|Y) is calculated using (9).

$$H(X|Y) = -\sum_{y_j\in Y} p(y_j)\sum_{x_i\in X} p(x_i|y_j)\log(p(x_i|y_j)) \tag{9}$$

The general formula for the linear combinations of Shannon information terms is represented by [53], [54] and expressed in (10).

$$J(X_k) = I(X_k;Y) - \beta\sum_{X_j\in S} I(X_j;X_k) + \gamma\sum_{X_j\in S} I(X_j;X_k|Y) \tag{10}$$

This equation is composed of a relevancy term and a redundancy term, represented by expressions (11) and (12), respectively. The redundancy term represented in (12), is sum of marginal redundancy shown in expression (13) and conditional redundancy shown in expression (14), both being weighed by parameters β and γ with values between 0 and 1.

$$I(X_k;Y) \tag{11}$$
$$\beta\sum_{X_j\in S} I(X_j;X_k) + \gamma\sum_{X_j\in S} I(X_j;X_k|Y) \tag{12}$$
$$\beta\sum_{X_j\in S} I(X_j;X_k) \tag{13}$$

$$\gamma \sum_{X_j \in S} I(X_j; X_k|Y) \tag{14}$$

The mutual information between the candidate feature X_k and the class label Y is expressed by $I(X_k; Y)$, while the conditional mutual information between X_k and other features X_j in the selected set S given the class label Y is expressed by $I(X_j; X_k|Y)$. The marginal redundancy is expressed by mutual information between feature of selected set S and feature of candidate set as $I(X_j; X_k)$.

2) weightedGAN (wGAN)

In this study, the proposed work consists of two modules addressing both challenges. The first module discussed earlier calculated the TF-IDF value from the pre-encryption data available in the form of JSON files. The pre-encryption phase has a small amount of data as data is extracted in little time before the encryption happens. To overcome the problem of data insufficiency, GAN was used. GAN was applied on TF-IDF of ransomware and benign samples to generate synthetic data with a similar distribution. A new dataset was generated consisting of original and synthetic features. However, dealing with patterns limitation problem does not justify which feature is more significant for the detection of ransomware attacks. The second module addressed the problem of data insignificance to perform detection of ransomware attacks. The significance of features was calculated in the significance estimator unit. This module calculated the significance of the features that are most useful for the development of behavioral drift-based ransomware detection technique. It assigned the weights (w) to the more significant features. The architecture of the proposed model weightedGAN (wGAN) is described in fig 2.

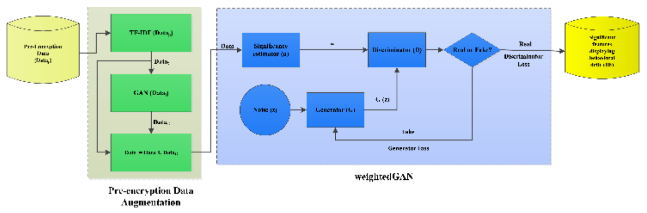
Pre-Encryption Data (Data$_p$)
TF-IDF (Data$_p$)
Data$_t$
GAN (Data$_t$)
Data$_s$
Data = Data$_t$ ∪ Data$_s$
Pre-encryption Data Augmentation
Data
Significance estimator (u)
w
Discriminator (D)
Real or Fake?
Real
Discriminator Loss
Noise (z)
Generator (G)
G (z)
Fake
Generator Loss
weightedGAN
Significant features displaying behavioral drift (Dh)

FIGURE 2. The architecture of weightedGAN technique.

The synthetic features of the pre-encryption data were generated using the GAN that helped to imitate the polymorphic and metamorphic ransomwares for the development of an adaptive detection model. The first module acts as a pre-training module to generate data for training. In the second module, the significance of each feature is estimated by calculating the weight of each feature by using (15).

$$w_{ij} = X_{ij}.u_j \tag{15}$$

Here w_{ij} represents a significant feature containing non-null values. Features are represented by X_{ij}, which gains worth due to the significance estimator u_j. Here i and j belong to the set of whole numbers 0, 1, 2, 3, ... and represent an entry in the weighted set to be fed to the GAN. The weights are used to estimate the significance of the features. The mathematical form of weightedGAN is presented in (16).

$$\begin{aligned} &wV(D, G) \\ &= wE_{x\sim Pdata(x)}[\log D(x)] + wE_{z\sim P_Z(z)}[\log(1 - D(G(z)))] \end{aligned} \tag{16}$$

A flowchart of the process of generation of significant features imitating polymorphic and metamorphic ransomwares, displaying behavioral drift by using the proposed weightedGAN technique is explained in fig 3.

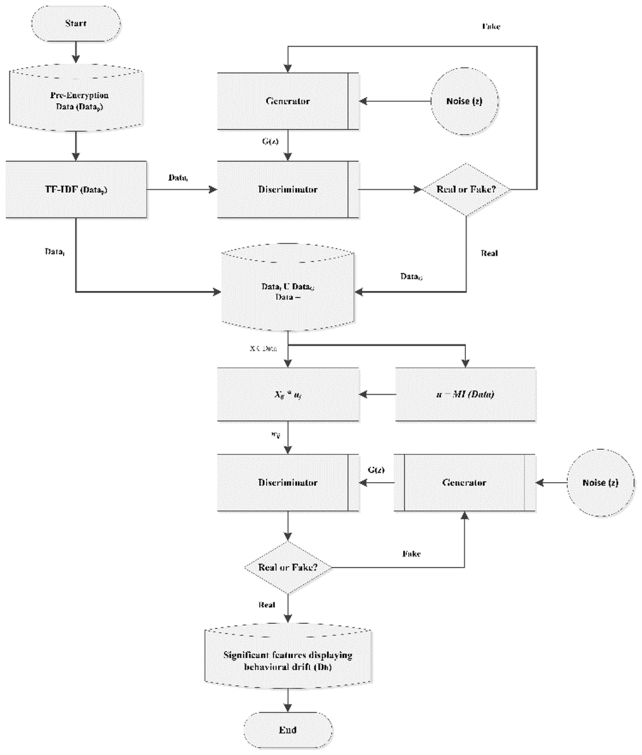

FIGURE 3. The flowchart of generation of significant features displaying behavioral drift ransomwares using weightedGAN.

The pre-encryption dataset Data$_p$ contains both benign and ransomware samples. TF-IDF values are calculated for both types of samples i.e., benign and ransomware. The samples of both ransomware and benign are fed in the GAN to generate synthetic features of pre-encryption patterns. A combination of synthetic and pre-encryption patterns was fed to weightedGAN. The weightedGAN generated significant features carrying patterns that display behavioral drift.

F. EXPERIMENTAL CONSIDERATIONS

The detection of threats and attacks becomes effortless and time efficient by ripping off the advantages of the latest technologies, i.e., deep learning, due to their high tendency of pattern understanding. Therefore, this work also utilized deep-learning algorithms to perform ransomware classification for emerging attacks depicting behavioral drift [55], [56].

Pseudo-code: weightedGAN (wGAN) technique for the generation of significant features displaying behavioral drift.

Input: Pre-Encryption Dataset ($Data_p$)
Output: Significant features displaying behavioral drift (D_b)

```
1: $Data_t$ ← Set of Pre-encryption features extracted from APIs
2: Data ← Set of Pre-encryption features carrying potential attack patterns
3: C = Class Label (0,1) ; w ← Ø
4: G ← Generative Neural Network
5: D ← Discriminator Neural Network
6: $D_b$ ← Set of significant features displaying behavioral drift

7: for each API ε $Data_p$ do
8:     $Data_t$ ← TF-IDF(API)
9:     end for
10: for each $y_{ij}$ ε $Data_t$ do
11:     do until convergence of loss values
12:         Generate mini-batches $B_i$ of values $y_{ij}$ from $Data_t$
13:         Calculate random vector $Z_{1i}$ for $G_1$
14:         for ∀ $y_{ij}$ ε $Data_t$ train $G_1$ by min $[\log(1 - D(G(z)))]$
15                      for ∀ $y_{ij}$ ε $Data_t$ train $D_1$ by max $[\log D(y) + \log(1 - D(G(z)))]$
16:     end for
17:     $Data_G$ ← Return $E_y[\log D(y)] + E_z[\log(1 - D(G(z)))]$
18: Data ← $Data_t$ U $Data_G$

19: for each $X_{ij}$ ε Data do
20:     $u_j = MI(X_{ij} ; C)$
21:     $w_{ij} = X_{ij} * u_j$
22:     end for
23: for each $a_{ij}$ ε $w_{ij}$ do
24:     do until convergence of loss values
25:     Generate mini-batches $B_i$ of values $a_{ij}$ from $w_{ij}$
26:     Calculate random vector $Z_{2i}$ for $G_2$
27:     for ∀ $a_{ij}$ ε $w_{ij}$ train $G_2$ and $D_2$ by
            $wE_a[\log D(a)] + wE_z[\log(1 - D(G(z)))]$
28:     end for
29:     $D_b$ ← $wE_a[\log D(a)] + wE_z[\log(1 - D(G(z)))]$
```

FIGURE 4. The pseudo-code of the proposed technique weightedGAN (wGAN).

The results are generated and analyzed by using deep learning models. The deep learning models can learn the underlying patterns quickly. All the models were trained on the same considered datasets to compare the resulting efficiency for the considered models. Deep learning models included in this work are Convolutional Neural Networks (CNN), Deep Neural Networks (DNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Multilayer Perceptron (MLP).

The model's validity is expressed by the predictions made by the model it is supposed to predict i.e., representing the behavior of a model [57]. There are different validation techniques, i.e., re-substitution, hold-out, bootstrapping etc. [58]. We used hold-out validation, a widely used validation technique, to randomly split the dataset into training and testing subsets. This data division is performed to avoid the model overfitting issues. If the data splitting is not performed, the model is evaluated on the same data used in the training set which ultimately causes model overfitting [59]. In each experiment, the dataset was divided into training and testing data in a ratio of 80-20 respectively. The training dataset was further split up into training and validation datasets in a ratio of 60-20 respectively during the model fitting process. The testing dataset was used to measure the performance of the trained model.

G. EVALUATION PARAMETERS

The detection models are evaluated by considering a set of metrics. Different machine and deep learning studies in literature considered a different set of performance indicators for the model evaluation. In this work, the obtained results are evaluated by using the performance metrics for classification problems, including accuracy, precision, recall, and F1-score [60]. These evaluation metrics are calculated using the TP, TN, FP, and FN values. Other performance indicators include FPR and DR. A brief description of each term and their relationship with each other is shown in Table 2.

IV. EXPERIMENTAL RESULTS

In this work, a ransomware detection technique was introduced that can adapt according to the developing variants of ransomware. This technique performed detection before the encryption process starts for the ransomware variants displaying behavioral drift concept. The technique consists of two modules each of which addressed a different problem. The first module addressed the problem of data limitation in the pre-encryption phase. This problem is solved by using synthetic data, similar in distribution to pre-encryption data. The second module addressed the significance problem for developing polymorphic and metamorphic ransomwares to deal with the behavioral drift concept.

This section presents the performance of the proposed behavioral model against the available relevant studies [19], [42]. The results validation is performed by comparing the results of the proposed work with related work. The datasets of related works are generated by implementation of the study. To check the efficacy of the proposed work, a comparison is drawn using the performance depicted by considered datasets trained on five deep learning models CNN, DNN, RNN, MLP, and LSTM. The performance of each model is different when trained on different datasets. Evaluation metrics involved in the comparison included accuracy, precision, recall, and F1-score. False Positive Rate (FPR) and Detection Rate (DR) are also used to elaborate on the potency of the proposed work. To comprehend the proposed work, result analysis and discussion are also presented in the next section.

TABLE 2. Evaluation parameters.

Terms	Quantity		Conversion from Gaussian and CGS EMU to SI [a]
True Positive	The TP value represents the ransomware samples predicted correctly as ransomware.	TP	[44, 59, 61]
True Negative	The TN value represents the benign samples predicted correctly as benign.	TN	[44, 59, 61]
False Positive	The FP value represents the predicted ransomware samples as benign.	FP	[44, 59, 61]
False Negative	The FN value represents predicted benign samples as ransomware.	FN	[44, 59, 61]
False Positive Rate (Fall-Out)	A ratio of ransomware samples predicted as benign with sum of FP and TN. It represents the measurement of accuracy.	$FPR = \frac{FP}{FP+TN}$ (17)	[62-64]
Detection Rate	A ratio between the correctly detected ransomware samples and the sum of total ransomware and benign samples.	$DR=\frac{TP}{TP+TN+FP+FN}$ (18)	[65]
Accuracy	Accuracy represents overall correctly identified samples. It is a ratio of correct predictions over total predictions.	$Accuracy=\frac{TP+TN}{TP+TN+FP+FN}$ (19)	[44, 59, 61]
Precision (Positive Value)	Precision describes correctly predicted samples. It is a ratio of True Positive with all the positive predictions; hence, visualize the reliability. It is a measure of quality i.e., predictions made.	$Precision=\frac{TP}{TP+FP}$ (20)	[44, 59, 61, 65]
Recall (Sensitivity or True Positive)	It represents the number of times a specific sample is detected by the model. It is a measure of quantity i.e., detection made.	$Recall=\frac{TP}{TP+FN}$ (21)	[44, 59, 61, 65]
F1-score	F1-score represents the weighted average. It is a harmonic mean of precision and recall.	$F1-score=2.(\frac{Precision.Recall}{Precision+Recall})$ (22)	[44, 59, 61, 65]

Here the data from the proposed work is referred to as dataset proposed (DSPr) contains 2125 records, Sharmeen [42] is referred to as dataset 1 (DS1) contains 1063 records, and Al-rimy [19] is referred to as dataset 2 (DS2) contains 1063 records. The dataset DSPr has almost doubled in records as it is a combination of the original pre-encryption data and synthetic data of the pre-encryption values to compensate the problem of data limitation in pre-encryption phase.

The performance of the detection model is presented by using performance graphs drawn against evaluation metrics. A performance graph visualizes the performance of a model against different considered aspects. The performance of each model represents its suitability for the proposed work. The comparison of the proposed work and related datasets is visualized in figures 5, 6, 7, and 8 reflecting accuracy, precision, recall, and F1-score metrics respectively. In all the performance graphs, the measurement metrics value for the proposed work and related works is plotted on the y-axis while adopted deep learning models were represented on the x-axis.

Accuracy represents the correctly identified samples. It is a ratio of correct predictions over total predictions. Accuracy is calculated according to (19). The bigger the numerator produced the higher accuracy. It is usually calculated in points that range between 0 to 1 and it is presented in the percentage. However, relying only on accuracy is not good practice. That's why other measures are also used to explain the effectiveness of the proposed work. The proposed work outperformed the classification performance in terms of accuracy. The accuracy of the proposed work stands highest against considered related studies for all the deep learning models. The highest accuracy of the proposed work was 0.9765 achieved by the RNN model and the lowest accuracy of 0.9247 was achieved by the CNN model.

Precision is the positive predictive value and represents the correctly predicted samples. It is a ratio of True Positive with all the positive predictions; hence, visualize the reliability. It is a measure of quality i.e., predictions made. Precision is calculated according to (20). The precision results for the deep learning models range between 0 to 1. The higher precision value represents the more relevant results hence, good performance of the model. Precision value is affected by FP value i.e., lower or zero, FP value will generate higher precision results. The precision values of the proposed work are highest only for the LSTM model. The precision value of the proposed technique varies among different trained

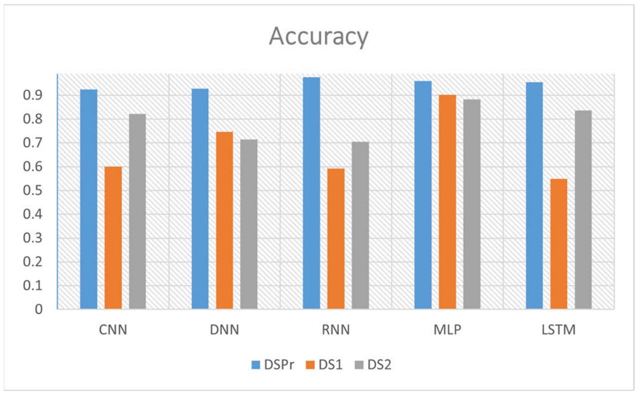

FIGURE 5. Classification Accuracy of the proposed technique with available state-of-art techniques.

DL models. On average, the precision of DSPr outperformed the other two comparison studies. The precision results stand highest at 0.9891 for the LSTM model and lowest at 0.8615 for the CNN model.

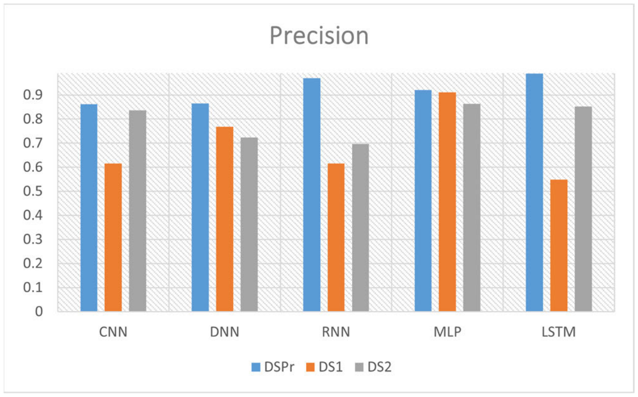

FIGURE 6. Classification Precision of the proposed technique with available state-of-art techniques.

The recall represents specific samples hence also called Sensitivity or True Positive Rate. It represents the number of times a specific sample is detected. It is a measure of quantity i.e., detection made. The recall is calculated according to (21). The recall results of the deep learning models range between 0 to 1. A higher recall value represents a higher detection value of the model. The proposed work outperformed the classification performance in terms of recall. Recall of the proposed work is highest against considered studies for the model CNN, DNN, and MLP by achieving the value of 1. In the case of RNN and LSTM models the proposed work still performed better than the considered work.

The F1-score represents the unbiasedness in the detection results of a model. F1-score considers both measures i.e., precision and recall which describe relevance. F1-score is 1 only when both precision and recall are 1. F1-score is calculated according to (22). F1-score represents the weighted average. It is a harmonic mean of precision and recall and a better

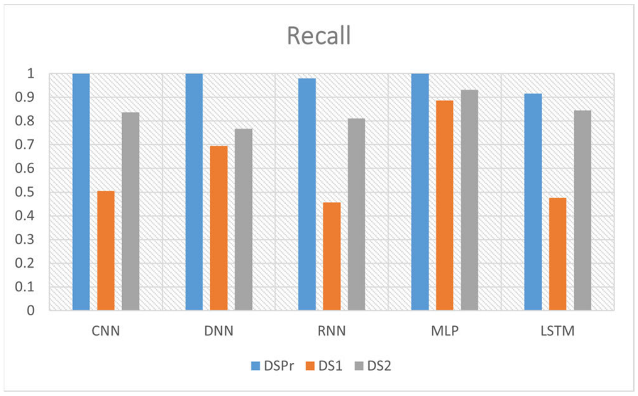

FIGURE 7. Classification Recall of the proposed technique with available state-of-art techniques.

measure than accuracy. The F1-score results for the deep learning models range between 0 and 1. High precision and recall produce a high value of F1-score and represent good results. The proposed work outperformed classification performance in terms of the F1-score. F1-score ranges between the lowest value of 0.9256 for CNN and the highest value of 0.975 for the RNN model.

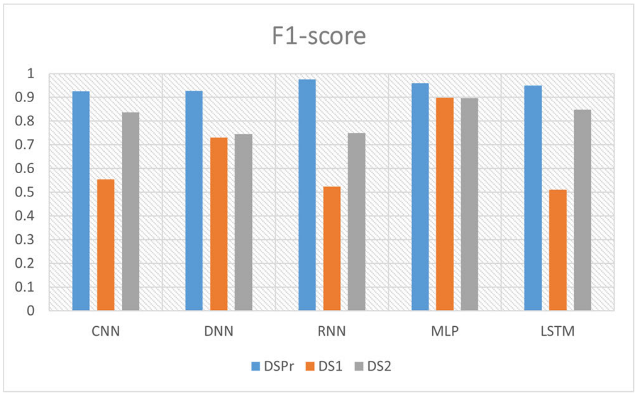

FIGURE 8. Classification F1-score of the proposed technique with available state-of-art techniques.

The summary of achieved results of the proposed work with different deep learning models and measurement metrics is summarized in Table 3.

TABLE 3. Comparison results of the proposed work with different deep learning models.

	Accuracy	Precision	Recall	F1-SCORE
CNN	0.9247	0.8615	1	0.9256
DNN	0.9271	0.8652	1	0.9277
RNN	0.9765	0.9702	0.9799	0.975
MLP	0.96	0.9213	1	0.959
LSTM	0.9553	0.9891	0.9146	0.9504

In addition to the considered metrics, a comparison of the False Positive Rate and Detection rate for the proposed work is also used to represent the efficacy of the proposed technique. A False Positive Rate (FPR) also known as Fall-Out is the measurement of accuracy and calculated with (17). False Positive Rate is defined as a ratio of false positive with a sum of false positive and true negative. A ratio of benign samples predicted as ransomware with the total number of ground truth negatives. In Table 4, the FPR of each model trained on different datasets is given. The proposed work depicted the lowest FPR for LSTM with a value of 0.0088 and the highest FPR for CNN with a value of 0.1415. Hence, displayed better classification performance for LSTM and the lowest performance for CNN. The proposed work also displayed the lowest False Positive Rate against comparison studies. FPR of the proposed work along with considered work is also drawn in fig 9.

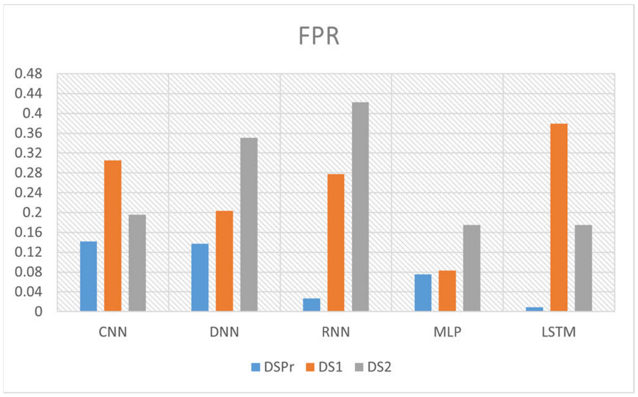

FIGURE 9. False positive Rate (FPR) of the proposed technique with available state-of-art techniques.

A detection rate is a ratio between the correctly detected ransomware samples and the sum of total ransomware and benign samples [65]. It is calculated according to (18). In Table 5, a comparison of the related studies and the proposed work is presented based on the detection rate. The detection rate of the proposed technique is highest for all the DL models as compared to related techniques. The highest detection rate of 0.9765 was accomplished for RNN and the lowest value of 0.9247 was for the CNN model. A pictorial representation of the Detection rate is presented in fig 10. It shows that the proposed technique presented the highest detection rate against all the considered comparison studies.

Table 6 and 7 show the significance results of the p-test and t-test for the proposed technique against considered studies of datasets DS1 and DS2 by using previously mentioned performance measures include accuracy, precision, recall, F1-score, False Positive Rate, and Detection rate. The tables show that the proposed technique is significant against both comparison work.

V. DISCUSSION

Ransomware detection solutions in the literature assume that sufficient data is available during pre-encryption phase and field of ransomware is stationary and does not evolve over time. In this work we performed early detection for developing variants of ransomware attacks in the presence of behavioral drift concept. The proposed technique presented an adaptive solution to deal with developing polymorphic and metamorphic crypto-ransomware attacks while considering the behavioral drift concept in different timespans. This work also coped with data limitations problem during pre-encryption phase.

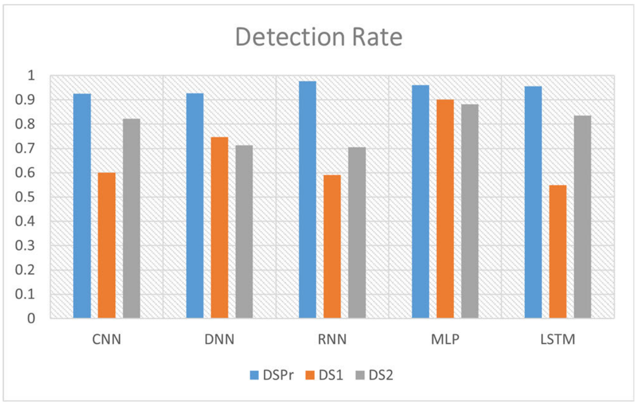

FIGURE 10. Detection Rate (DR) of the proposed technique with available state-of-art techniques.

The proposed work has the highest performance metrics as the problem of data limitation is properly addressed and the significance of features is adequately estimated. Discovered ransomware variants act like historical data i.e., data on hand for model training. However, a model performance could be degraded if it is only trained on historical data which might not generalize the drift in the behavior of developing variants over different timeline. Existing solutions performed detection by looking into historical data, which might not help to develop the behavioral drift detection model. To develop an adaptive detection model, we tried to predict the significance of potential attacks that can be developed in the future. Future behavior will be displayed by discovering the potential patterns of future variants. A model should be trained on historical and potential patterns to deal with emerging attacks. The problem of behavior changing was addressed by training the model with the potential patterns of existing and future ransomwares. In addition to addressing the behavioral drift problem, the problem of data limitation during pre-encryption phase is also solved.

According to figures 5 and 6 representing accuracy and precision respectively, the wGAN technique correctly identified and detected the ransomware samples. The original samples and generated synthetic data represented the true nature of ransomware attacks. The significance of features for discovered and developing variants is appropriately estimated which helped to achieve the highest performance metrics. For the proposed technique, the highest accuracy represents reliance, the highest precision represents reliability, and the highest recall represents the quantity of correctly predicted ransomwares. The correctly predicted ransomware samples

TABLE 4. Comparison of FPR of the proposed work with other state-of-the-art approaches.

	Studies	Year	TECHNIQUE NAME	CNN	DNN	RNN	MLP	LSTM
DS1	[42]	2020	Adaptive Approach	0.1959	0.3505	0.4226	0.1753	0.1753
DS2	[19]	2020	DPBD-FE	0.3055	0.2037	0.2777	0.0833	0.3796
DSPr		2023	wGAN	0.1415	0.1371	0.0265	0.0752	0.0088

TABLE 5. Comparison of Detection rate (DR) of the proposed work with other state-of-the-art approaches.

	Studies	Year	TECHNIQUE NAME	CNN	DNN	RNN	MLP	LSTM
DS1	[42]	2020	Adaptive Approach	0.6009	0.7464	0.5915	0.9014	0.5493
DS2	[19]	2020	DPBD-FE	0.8216	0.7136	0.7042	0.8826	0.8356
DSPr		2023	wGAN	0.9247	0.9271	0.9765	0.96	0.9553

TABLE 6. Significance test (t-test) of the proposed work with considered state-of-art study (DS1).

	p-value	*t-value*	Significance
Accuracy	0.014857	4.099652	Statistically Significant
Precision	0.044698	2.886985	Statistically Significant
Recall	0.007567	4.985461	Statistically Significant
F1-score	0.016035	4.007046	Statistically Significant
FPR	0.041894	2.658864	Statistically Significant
DR	0.014857	4.099652	Statistically Significant

TABLE 7. Significance test (t-test) of the proposed work with considered state-of-art study (DS2).

	p-value	*t-value*	Significance
Accuracy	0.012971	4.267926	Statistically Significant
Precision	0.041995	2.949383	Statistically Significant
Recall	0.011188	4.456752	Statistically Significant
F1-score	0.012564	4.308169	Statistically Significant
FPR	0.034636	3.146330	Statistically Significant
DR	0.012959	4.269060	Statistically Significant

produced the highest value of F1-score. The proposed technique displayed the highest detection rate and lowest FPR as the model is already trained on the developing variants of ransomware. Recall, f1-score, DR, and FPR are represented in figures 7, 8, 9, and 10 respectively. This pre-training on potential features not only increased the detection rate, but it also increased the response time which ultimately safeguards the timely detection before occurrence of ransomware attack. Each feature is considered to deal with the complementary nature of all instances to make this work more reliable. The early training on all the features of developing variants will minimize the occurrence of ransomware attacks variants and make it suitable to be a part of malicious software detection. The pre-encryption and early training of the model will make it suitable for ransomware detection in the real world with maximum efficiency. The proposed work outperformed for all the considered performance metrics. These better results are also supported by performing significance test. Table 6 and 7 presented the significance of the obtained results after the application of the proposed work. It shows for both comparison studies, the p-value is less than 0.05 for all the cases. This implied that the performance of the proposed work is statistically significant. The results also emphasize that wGAN generated cryptographic APIs i.e., significant features are relevant and substantially weighted to determine the ransomware attacks. The generated polymorphic and metamorphic ransomwares reflected the same behavior as ransomwares do.

The proposed work can detect the ransomware's new variants by analyzing data from the detected and developing ransomwares due to polymorphism and metamorphism of ransomware signatures. The most of existing solution might lack in performance as they only learn from discovered ransomware attacks. By assessing the developing variant of ransomware, the proposed detection architecture presented time efficient and robust solution to behavioral drifting. If behavioral drifting is not considered properly, a system will misclassify future ransomware variants as benign program. The prediction model will result in sudden spikes of false negative, whenever a new variant emerges. The proposed technique is time efficient and detect ransomware in real-time with less time complexity as the future variants are already predicted and the model is trained before the attacks occur. Results after wGAN implementation emphasized that the generated data from the features of synthetic data are closely related to the original pre-encryption data. The improvement observed after applying the proposed work is summarized in table 3. This reflects that the wGAN technique was able to better represent the detection of the behavioral drift concept. The comparison results indicated that the *weightedGAN* technique can detect the crypto-ransomware before the encryption process starts and the adaptive model can detect the new variants of ransomware that incorporate the behavioral drift over time. The availability of pre-encryption behavioral information before the attacks occur makes this technique adaptive. Our approach presented lowest false alarms as compared to previously published methods for developing variants depicting future behavior. The higher results of performance metrics represent the suitability of the proposed technique for the development of an adaptive model.

VI. CONCLUSION

Ransomware attacks, fueled by the evolution of cyber threats, pose a significant issue due to their irreversible encryption mechanisms. This paper introduced an innovative approach to these evolving challenges, proposing a weighted Generative Adversarial Network (wGAN) to augment data during the pre-encryption phase of an attack and re-evaluate the significance of features as per the changing behavior of new ransomware variants. The wGAN technique demonstrated robust performance against the behavioral drift of ransomware attacks, effectively predicting new variants by generating synthetic data mimicking existing and potential ransomware patterns. Extensive experimentation revealed the superior accuracy, precision, and detection rate of the wGAN, along with a low false positive rate. The results were statistically significant, underscoring the wGAN's potential as an adaptive, reliable, and efficient model for the early detection of ransomware attacks and indicating a promising direction for future research in the field.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through large group Research Project under grant number (RGP.2/175/44).

REFERENCES

[1] A. Young and M. Yung, "Cryptovirology: Extortion-based security threats and countermeasures," in *Proc. IEEE Symp. Secur. Privacy*, May 1996, pp. 129–140.

[2] U. Urooj, B. A. S. Al-Rimy, A. Zainal, F. A. Ghaleb, and M. A. Rassam, "Ransomware detection using the dynamic analysis and machine learning: A survey and research directions," *Appl. Sci.*, vol. 12, no. 1, p. 172, Dec. 2021.

[3] G. Hull, H. John, and B. Arief, "Ransomware deployment methods and analysis: Views from a predictive model and human responses," *Crime Sci.*, vol. 8, no. 1, pp. 1–22, Dec. 2019.

[4] A. Alqahtani and F. T. Sheldon, "A survey of crypto ransomware attack detection methodologies: An evolving outlook," *Sensors*, vol. 22, no. 5, p. 1837, Feb. 2022.

[5] *Gang Behind Huge Cyber-Attack Demands $70m in Bitcoin*, BBC News, London, U.K., 2021.

[6] Monika, P. Zavarsky, and D. Lindskog, "Experimental analysis of ransomware on windows and Android platforms: Evolution and characterization," *Proc. Comput. Sci.*, vol. 94, pp. 465–472, Jan. 2016.

[7] L. F. Maimó, A. H. Celdrán, Á. P. Gómez, F. G. Clemente, J. Weimer, and I. Lee, "Intelligent and dynamic ransomware spread detection and mitigation in integrated clinical environments," *Sensors*, vol. 19, no. 5, p. 1114, Mar. 2019.

[8] U. J. Butt, M. Abbod, A. Lors, H. Jahankhani, A. Jamal, and A. Kumar, "Ransomware threat and its impact on SCADA," in *Proc. IEEE 12th Int. Conf. Global Secur., Saf. Sustainability (ICGS)*, Jan. 2019, pp. 205–212.

[9] N. Scaife, H. Carter, P. Traynor, and K. R. B. Butler, "CryptoLock (and drop It): Stopping ransomware attacks on user data," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2016, pp. 303–312.

[10] N. Kshetri and J. Voas, "Do crypto-currencies fuel ransomware?" *IT Prof.*, vol. 19, no. 5, pp. 11–15, 2017.

[11] T. Micro, "A deep dive into the evolution of ransomware part 1," TREND, Tech. Rep., 2023.

[12] N. K. Popli and A. Girdhar, "Behavioural analysis of recent ransomwares and prediction of future attacks by polymorphic and metamorphic ransomware," in *Computational Intelligence: Theories, Applications and Future Directions*, vol. 2, Springer, 2019, pp. 65–80.

[13] R. Chauhan and S. Shah Heydari, "Polymorphic adversarial DDoS attack on IDS using GAN," in *Proc. Int. Symp. Netw., Comput. Commun. (ISNCC)*, Oct. 2020, pp. 1–6.

[14] S. P. Choudhary and M. D. Vidyarthi, "A simple method for detection of metamorphic malware using dynamic analysis and text mining," *Proc. Comput. Sci.*, vol. 54, pp. 265–270, Jan. 2015.

[15] D. E. García, N. DeCastro-García, and A. L. M. Castañeda, "An effectiveness analysis of transfer learning for the concept drift problem in malware detection," *Expert Syst. Appl.*, vol. 212, Feb. 2023, Art. no. 118724.

[16] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining Knowl. Discovery*, vol. 8, no. 4, p. e1249, Jul. 2018.

[17] Y. Chen, Z. Ding, and D. Wagner, "Continuous learning for Android malware detection," 2023, *arXiv:2302.04332*.

[18] A. Narayanan, M. Chandramohan, L. Chen, and Y. Liu, "Context-aware, adaptive, and scalable Android malware detection through online learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 1, no. 3, pp. 157–175, Jun. 2017.

[19] B. A. S. Al-Rimy, M. A. Maarof, M. Alazab, F. Alsolami, S. Z. M. Shaid, F. A. Ghaleb, T. Al-Hadhrami, and A. M. Ali, "A pseudo feedback-based annotated TF-IDF technique for dynamic crypto-ransomware pre-encryption boundary delineation and features extraction," *IEEE Access*, vol. 8, pp. 140586–140598, 2020.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 2672–2680.

[21] B. A. S. Al-Rimy, M. A. Maarof, and S. Z. M. Shaid, "Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions," *Comput. Secur.*, vol. 74, pp. 144–166, May 2018.

[22] U. Urooj, M. A. B. Maarof, and B. A. S. Al-Rimy, "A proposed adaptive pre-encryption crypto-ransomware early detection model," in *Proc. 3rd Int. Cyber Resilience Conf. (CRC)*, Jan. 2021, pp. 1–6.

[23] F. De Gaspari, D. Hitaj, G. Pagnotta, L. De Carli, and L. V. Mancini, "Evading behavioral classifiers: A comprehensive analysis on evading ransomware detection techniques," *Neural Comput. Appl.*, vol. 34, no. 14, pp. 12077–12096, Jul. 2022.

[24] S. Mehnaz, A. Mudgerikar, and E. Bertino, "RWGUard: A real-time detection system against cryptographic ransomware," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*, Springer, 2018, pp. 114–136.

[25] D. Morato, E. Berrueta, E. Magaña, and M. Izal, "Ransomware early detection by the analysis of file sharing traffic," *J. Netw. Comput. Appl.*, vol. 124, pp. 14–32, Dec. 2018.

[26] Q. Chen, S. R. Islam, H. Haswell, and R. A. Bridges, "Automated ransomware behavior analysis: Pattern extraction and early detection," in *Proc. Int. Conf. Sci. Cyber Secur.*, Springer, 2019, pp. 199–214.

[27] S. Kok, A. Abdullah, N. Jhanjhi, and M. Supramaniam, "Prevention of crypto-ransomware using a pre-encryption detection algorithm," *Computers*, vol. 8, no. 4, p. 79, Nov. 2019.

[28] M. Scalas, D. Maiorca, F. Mercaldo, C. A. Visaggio, F. Martinelli, and G. Giacinto, "On the effectiveness of system API-related information for Android ransomware detection," *Comput. Secur.*, vol. 86, pp. 168–182, Sep. 2019.

[29] R. Moussaileb, N. Cuppens, J.-L. Lanet, and H. L. Bouder, "Ransomware network traffic analysis for pre-encryption alert," in *Proc. Int. Symp. Found. Pract. Secur.*, Springer, 2019, pp. 20–38.

[30] S. H. Kok, A. Abdullah, and N. Jhanjhi, "Early detection of crypto-ransomware using pre-encryption detection algorithm," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 5, pp. 1984–1999, May 2022.
[31] S. H. Kok, A. Azween, and N. Jhanjhi, "Evaluation metric for crypto-ransomware detection using machine learning," *J. Inf. Secur. Appl.*, vol. 55, Dec. 2020, Art. no. 102646.
[32] K. C. Roy and Q. Chen, "DeepRan: Attention-based BiLSTM and CRF for ransomware early detection and classification," *Inf. Syst. Frontiers*, vol. 23, pp. 1–17, Jun. 2020.
[33] N. Pundir, M. Tehranipoor, and F. Rahman, "RanStop: A hardware-assisted runtime crypto-ransomware detection technique," 2020, *arXiv:2011.12248*.
[34] C.-Y. Yang and R. Sahita, "Towards a resilient machine learning classifier—A case study of ransomware detection," 2020, *arXiv:2003.06428*.
[35] A. Alqahtani, M. Gazzan, and F. T. Sheldon, "A proposed crypto-ransomware early detection (CRED) model using an integrated deep learning and vector space model approach," in *Proc. 10th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2020, pp. 275–279.
[36] S. S. Chakkaravarthy, D. Sangeetha, M. V. Cruz, V. Vaidehi, and B. Raman, "Design of intrusion detection honeypot using social leopard algorithm to detect IoT ransomware attacks," *IEEE Access*, vol. 8, pp. 169944–169956, 2020.
[37] T. McIntosh, A. S. M. Kayes, Y.-P.-P. Chen, A. Ng, and P. Watters, "Dynamic user-centric access control for detection of ransomware attacks," *Comput. Secur.*, vol. 111, Dec. 2021, Art. no. 102461.
[38] B. V. Reddy, G. J. Krishna, V. Ravi, and D. Dasgupta, "Machine learning and feature selection based ransomware detection using hexacodes," in *Evolution in Computational Intelligence*, Springer, 2021, pp. 583–597.
[39] M. Scalas, K. Rieck, and G. Giacinto, "Explanation-driven characterization of Android ransomware," in *Proc. Int. Conf. Pattern Recognit.*, Springer, 2021, pp. 228–242.
[40] D. Hitaj, G. Pagnotta, F. De Gaspari, L. De Carli, and L. V. Mancini, "Minerva: A file-based ransomware detector," 2023, *arXiv:2301.11050*.
[41] G. O. Ganfure, C.-F. Wu, Y.-H. Chang, and W.-K. Shih, "RTrap: Trapping and containing ransomware with machine learning," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 1433–1448, 2023.
[42] S. Sharmeen, Y. A. Ahmed, S. Huda, B. S. Koçer, and M. M. Hassan, "Avoiding future digital extortion through robust protection against ransomware threats using deep learning based adaptive approaches," *IEEE Access*, vol. 8, pp. 24522–24534, 2020.
[43] P. B. Caliaberah, S. Armoogum, and X. Li, "An adaptive security architecture for detecting ransomware attack using open source software," in *Proc. Future Inf. Commun. Conf. (FICC)*, vol. 1, Springer, 2020, pp. 618–633.
[44] J. Singh, K. Sharma, M. Wazid, and A. K. Das, "SINN-RD: Spline interpolation-envisioned neural network-based ransomware detection scheme," *Comput. Electr. Eng.*, vol. 106, Mar. 2023, Art. no. 108601.
[45] J.-Y. Kim, S.-J. Bu, and S.-B. Cho, "Malware detection using deep transferred generative adversarial networks," in *Proc. Int. Conf. Neural Inf. Process.*, Springer, 2017, pp. 556–564.
[46] A. Rajabi and O. O. Garibay, "TabFairGAN: Fair tabular data generation with generative adversarial networks," *Mach. Learn. Knowl. Extraction*, vol. 4, no. 2, pp. 488–501, May 2022.
[47] M. Amin, B. Shah, A. Sharif, T. Ali, K.-I. Kim, and S. Anwar, "Android malware detection through generative adversarial networks," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 2, p. e3675, Feb. 2022.
[48] C. Guarnieri, "Cuckoo," Tech. Rep., 2022.
[49] Y. A. Ahmed, B. Koçer, S. Huda, B. A. S. Al-Rimy, and M. M. Hassan, "A system call refinement-based enhanced minimum redundancy maximum relevance method for ransomware early detection," *J. Netw. Comput. Appl.*, vol. 167, Oct. 2020, Art. no. 102753.
[50] *VMware*, VMware, Palo Alto, CA, USA, 2022.
[51] R. Roy, "Best Python libraries for machine learning," GeeksforGeeks, Tech. Rep., 2022.
[52] *The Generator*, G. Developers, Mach. Learn., 2022.
[53] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *J. Mach. Learn. Res.*, vol. 13, pp. 27–66, Jun. 2012.
[54] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, pp. 1–45, Jan. 2018.
[55] S. I. Bae, G. B. Lee, and E. G. Im, "Ransomware detection using machine learning algorithms," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 18, p. e5422, Sep. 2020.
[56] B. Geluvaraj, P. Satwik, and T. A. Kumar, "The future of cybersecurity: Major role of artificial intelligence, machine learning, and deep learning in cyberspace," in *Proc. Int. Conf. Comput. Netw. Commun. Technol.*, Springer, 2019, pp. 739–747.
[57] S. Riezler and M. Hagmann, "Validity, reliability, and significance: Empirical methods for NLP and data science," *Synth. Lect. Hum. Lang. Technol.*, vol. 14, no. 6, pp. 1–165, 2021.
[58] A. Kumar, "Machine learning—Validation techniques (interview questions)," Data Anal. Data, Data Sci., Mach. Learn., AI, Tech. Rep., 2018.
[59] O. Shalev, "Recall, precision, F1, ROC, AUC, and everything," The Startup, Tech. Rep., 2019.
[60] E. Zvornicanin. (2022). *How to Use K-Fold Cross-Validation in a Neural Network?* Accessed: Oct. 15, 2022. [Online]. Available: https://www.baeldung.com/cs/k-fold-cross-validation
[61] *Confusion Matrix, Accuracy, Precision, Recall, F1 Score*, H. N. B, Analytics Vidhya, 2019.
[62] *False Positive Rate*, Wikipedia, 2022.
[63] T. Sarkar, "False positives/negatives and Bayes rule for COVID-19 testing," Towards Data Sci., Tech. Rep., 2020.
[64] Ö. Aslan, M. Ozkan-Okay, and D. Gupta, "Intelligent behavior-based malware detection system on cloud computing environment," *IEEE Access*, vol. 9, pp. 83252–83271, 2021.
[65] S. Prabhakaran, "Top 15 evaluation metrics for classification models," *Mach. Learn.*, 2017.

UMARA UROOJ received the B.S. degree (Hons.) in information technology and the M.S. degree in computer science from the Government College University Faisalabad, Pakistan, in 2012 and 2014, respectively. She is currently pursuing the Ph.D. degree in ransomware attacks detection with Universiti Teknologi Malaysia (UTM), Malaysia. Her research interests include malware, ransomware, and the IoT.

BANDER ALI SALEH AL-RIMY (Senior Member, IEEE) received the B.Sc. degree in computer engineering from the Faculty of Engineering, Sana'a University, Yemen, in 2003, the M.Sc. degree in information technology from Open University Malaysia (OUM), Malaysia, in 2013, and the Ph.D. degree in computer science (information security) from the Faculty of Computing, Universiti Teknologi Malaysia (UTM), Malaysia, in 2019. He is currently a Senior Lecturer with UTM. His research interests include malware, IDS, network security, and routing technologies. He was a recipient of several academic awards and recognitions, such as the UTM Alumni Award, the UTM Best Postgraduate Student Award, the UTM Merit Award, the UTM Excellence Award, the OUM Distinction Award, and the Best Research Paper Award.

ANAZIDA BINTI ZAINAL (Member, IEEE) received the B.Sc. degree in computer science from Rutgers University, New Brunswick, NJ, USA, in 1990, and the M.Sc. degree in computer science and the Ph.D. degree in computer science and network security from Universiti Teknologi Malaysia (UTM), Malaysia, in 2000 and 2011, respectively. She is currently an Associate Professor with the Faculty of Computing and leading the Information Assurance and Security Research Group (IASRG), UTM. Her research interests include cyber threat intelligence, security analytics, network security, and anomaly detection.

FAISAL SAEED received the B.Sc. degree in computers (information technology) from Cairo University, Egypt, and the M.Sc. degree in information technology management and the Ph.D. degree in computer science from Universiti Teknologi Malaysia (UTM), Malaysia. He is currently a Senior Lecturer with the Computing and Data Science Department, School of Computing and Digital Technology, Birmingham City University (BCU), U.K., where he is also leading the Smart Health Laboratory, Data Analytics and AI Research Group. Previously, he was an Assistant/Associate Professor with Taibah University, Saudi Arabia, from 2017 to 2021, and a Senior Lecturer with the Department of Information Systems, Faculty of Computing, UTM, from 2014 to 2017. He has published several papers in indexed journals and international conferences. His research interests include data mining, artificial intelligence, machine learning, information retrieval, and health informatics. He served as the general chair for several international conferences and the guest editor for several indexed journals.

ABDELZAHIR ABDELMABOUD received the M.Sc. degree in computer science and information from Gezira University, Sudan, and the Ph.D. degree in software engineering from Universiti Teknologi Malaysia (UTM), Malaysia. He is currently an Associate Professor with the Department of Information Systems, College of Science and Arts at Mohayil, King Khalid University, Saudi Arabia. Previously, he has been the IT Manager, the Quality Manager, and a Database Administrator. He is also a member of the Software Engineering Research Group (SERG), UTM. His research interests include cyber security and blockchain technology based on the Internet of Things and cloud computing.

WAMDA NAGMELDIN received the B.Sc. degree in computer science from the Faculty of Computer Studies, International University of Africa, Sudan, in 2004, the M.Sc. degree in computer science from the Faculty of Mathematical Sciences, University of Khartoum, Sudan, in 2007, and the Ph.D. degree in computer science from the University of Technology Malaysia, Malaysia, in 2020. She is currently a Lecturer with the Department of Information Systems, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Saudi Arabia. Her research interests include cryptography, cyber security, machine learning and networks, the IoT, and cloud computing.