Egyptian Informatics Journal 23 (2022) 45–55

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

A novel approach for detecting advanced persistent threats

Jaafer Al-Saraireh *, Ala' Masarweh

King Hussein School of Computing Sciences, Princess Sumaya University for Technology, PSUT, Jordan

ARTICLE INFO

Article history:
Received 21 March 2022
Revised 29 May 2022
Accepted 22 June 2022
Available online 14 July 2022

Keywords:
Advanced persistent threat
Machine learning
eXtreme gradient boosting
Analysis of variance

ABSTRACT

Cyber security has been drawing massive attention in recent years due to human reliance on new technology, and systems. Therefore, securing these systems against cyber-attacks has become an essential task nowadays. The advanced persistent threat is one of the most sophisticated cyber-attacks in which malicious actors gain unauthorized access to a network and remain undiscovered for a lengthy period of time. The number of recorded advanced persistent threat attacks and threats to organizations intensifies. One method used in detecting advanced persistent threat attacks is machine learning. However, this method has not been covered in many previous kinds of research due to the lack of datasets covering an entire advanced persistent threat attack lifecycle. Thus, this paper aims to build a new dataset that covers the complete life cycle of an advanced persistent threat attack to detect them in different stages such as normal, reconnaissance, initial compromise, lateral movement, and data exfiltration activities. The newly collected dataset is based on advanced persistent threat attacks using tactics, techniques, procedures, and indicators of compromise. Then, it is applied to a proposed machine learning model employing eXtreme gradient boosting and analysis of variance feature selection method. The model was compared to other traditional classifiers: random forest, decision tree, and K-nearest neighbor. Based on the accuracy score of the proposed model 99.89% using only 12 features, it proved to be more powerful and efficient than the other classifiers in detecting advanced persistent threat attacks. The dataset used in this research is newly built based on advanced persistent threat attack behaviors, which will aid organizations in detecting advanced persistent threat attack activity efficiently. The experimental evaluation proved that the proposed method effectively detects advanced persistent threat attacks at different stages.

1. Introduction

Protecting data has become an issue that has attracted everyone's attention worldwide [1]. Advanced Persistent Threat (APT) attacks are executed by threat actors sponsored by states, with the primary goal of hacking and infiltrating different systems globally. These attacks usually exploit unknown vulnerabilities called zero-day. Mitigating such attacks and detecting them can be an endless and bothersome task due to the lack of data on their whole lifecycle. Security entities have sought to prevent APT threats by identifying them in their early stages. Previous literature research did not provide a reliable dataset on the entire lifecycle of APT attacks, which could enhance the efforts of detecting and mitigating them. Furthermore, due to the limitations of the current approaches to cover this topic, APT attacks are still a problem facing security entities today where the custom Penetration Test (PT) and current detection methods are not capable of mitigating such attacks [1].

Much research has been conducted to detect and mitigate APT attacks using Artificial Intelligence (AI). The success of these detection methods relies significantly on the datasets used. On the other hand, APTs have been using AI to intelligently create new and more sophisticated attack methods making it even more challenging to detect them [2]. Therefore, should be a new approach implemented to address these issues.

Cyber Threat Intelligence (CTI) is crucial for collecting a comprehensive database on APT behavior. CTI assists in managing indicators of compromise, techniques, tactics, and protocols for various APT groups. Security analysts can use this information to under-

* Corresponding author.
E-mail address: j.saraireh@psut.edu.jo (J. Al-Saraireh).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.eij.2022.06.005
1110-8665/

stand the APT attack's behavior better. It also aids in determining which APT groups to pursue, as the information provided by CTI aids in closing the system's required gaps and vulnerabilities. CTI can come from within a security organization, from CTI feed providers, or cooperation with other security organizations around the world [1].

The problem statement addressed in this research paper is that the PT approach covers only the present vulnerabilities in a system without minding any new potential vulnerabilities exploited by APT groups that could harm such systems. In addition, previous literature research did not provide a reliable dataset on a complete lifecycle of APT attacks, which could enhance the efforts of detecting and mitigating them.

The motivation for this article comes from enhancing and proposing an effective detection method for APT attacks using various Machine Learning (ML) classifiers based on known patterns used by APTs in their attacks. To understand their intentions and possible maneuvers, there is a need to analyze different APT lifecycles comprehensively. This could help mitigate vulnerabilities exploited by such attacks and detect an attack early on before succeeding.

The primary purpose of this study is to produce a dataset utilizing CTI to aid in the development of the PT method and detection model. In addition, propose a model to detect APT attacks using ML and implement a comparative analysis to evaluate the approaches in this research work to verify the results achieved. The main contribution of this research work is to propose a model to detect APT attacks based on behavior using ML utilizing a new dataset created in this study.

The article is organized as follows: Section 2 provides background and reviews the related works; Section 3 details an outline of the proposed model framework and methodology; and Section 4 presents results and discussion. Section 5 discusses the conclusion and future research directions.

2. Background and related works

This section aims to provide background material, put this research into context, and set the scene for this research paper's contribution to knowledge.

2.1. Advanced persistent threat (APT)

The concept of APT came after an attack on the US-Force. They were targeted by different types of attacks from various hackers. As they requested the aid of civilian experts to help deter such attacks, they did not want to discuss the origin of the attack with these civilians. But at the same time, they wanted the civilian experts to know that these attacks were severe and devastating.

The National Institute of Standards and Technology (NIST) defines an APT as follows (NIST, 2012):

"An adversary with sophisticated levels of expertise and significant resources, allowing it through the use of multiple different attack vectors (e.g., cyber, physical, and deception) to generate opportunities to achieve its objectives, which are typically to establish and extend its presence within the information technology infrastructure of organizations for purposes of continually exfiltration information and to undermine or impede critical aspects of a mission, program, or organization, or place itself in a position to do so in the future; moreover, the APT pursues its objectives repeatedly over an extended period, adapting to a defender's efforts to resist it, and with determination to maintain the level of interaction needed to execute its objectives."

APT attacks are carefully planned and require several steps. APT groups could have distinct features, but the phases of their attacks are similar and the only difference could be in the tactics and techniques in each phase. The phases of an APT attack are [1,3]:

- **Phase 1 Reconnaissance:** APT threat groups do not attack at random. They frequently select targets based on their employer's needs. Some information is gathered via social engineering or Open Source Intelligence (OSINT).
- **Phase 2 Delivery:** it delivers the exploit to the target. The exploit is supplied directly via a phishing or spear-phishing email attachment. Another way to provide the exploit is to find a misconfiguration vulnerability on the target.
- **Phase 3 Installation & Exploitation:** the attacker must find a way to install the provided exploit on the target computer. This is frequently done using social engineering, especially if sent the attachment by email. He will try to convince the user to download the attachment or click the link. An exploit or macro will be executed on the victim's system. Once the exploit is successfully executed, the attacker attempts to connect to a Command and Control (C&C) server using secure ports such as port 443.
- **Phase 4 Lateral Movement:** the attacker will start moving around stealthily in the network. The attacker does not know where the sensitive information is located, so he will keep moving around until he finds it.
- **Phase 5 Data Exfiltration:** this could be a tedious task as if the data is exfiltrated in large amounts, the network administrators could discover it. The attackers could use legitimate tools on the target network to exfiltrate the data; this is called the living off the land technique. APTs use this technique to move around and execute undetected commands on a compromised network using the same tools and software used by the users on the compromised network.
- **Phase 6 Clearing Tracks and Erasing Evidence:** the attacker will attempt to clear his tracks and erase any evidence on the target system.

2.2. Machine learning

The main goal of ML is to enable computers to learn on their own, without the need for human intervention, and to adjust their actions and judgments accordingly. Supervised learning and unsupervised learning algorithms are the most commonly used.[4].

The process of ML consists of two phases: training and classification. In the training phase, the goal is to construct a model capable of predicting or detecting hidden features and characteristics of the data. The second phase, classification, is the phase in which the model controlled during the training is used on new data samples to perform a specified task (e.g., clustering) [5].

This research paper, deals with a multiclass classification problem; the most popular ML classifiers used for such a problem using the supervised approach such as Random Forest (RF), Decision Tree (DT), eXtreme Gradient Boosting (XGB), K-Nearest Neighbor (KNN), Naïve Bayes and Support Vector Machine (SVM).

2.3. Related works

Many kinds of research have been implemented in recent years to use ML algorithms in detecting APT attacks. An intrusion detection [6] was used by applying a hyper graph-based genetic algorithm for parameters and feature selection using SVM on the NSL-KDD dataset. The suggested method achieved a 96.7% accuracy. It was better than other SVM classifiers, Bayes net, and RF. Their research evaluation was based on the NSL-KDD dataset, which has some issues in it. This dataset used only solved some problems in the older version of KDD CUP 99, meaning that some problems still exist that would affect the performance and evaluation of the classifiers. Also, the results achieved were from the

SVM, which is known for not giving very accurate readings when dealing with high volumes of data.

A system for intrusion detection by deploying deep neural networks was implemented [7]. KDD CUP 99 was used to assess the model's effectiveness. The testing process was 10% to 90% of normal data against 90% to 10% of the attack data. They recorded an accuracy of approximately 99%, a high detection rate of more than 99%, and a false alarm of 0.08%. The authors used an open-source dataset that has some issues in it. This dataset has many replicated data in it, which could affect the results significantly.

Anomaly detection using graph databases and ML was proposed by [8]. Their approach was to implement a practical solution to rapidly analyze real-time data and detect any breach or malicious activity. The ML classifier that used anomaly detection was SVM. They achieved an accuracy from 95% to 98%. The authors stated that in a real-world scenario, there would be much more log data to filter through; thus, the SVM used in this research is known for not working very well with high volumes of data due to the performance. Also, it does not work very well when the data has a lot of noise, which in this case, when collecting real-time data, there will be plenty of noise that will hinder the detection process.

A novel system based on ML was presented by [9]. They claimed that the system could predict and detect APT incidents reliably and quickly. The three primary phases of the system are alert correlations, detecting threats, and predicting attacks. The prediction accuracy achieved was 84.8%. The ML algorithms used in this system were SVM, DT, KNN and Ensemble. The accuracy achieved in this research was not remarkable as they are dealing with APTs; there is still a 16% chance that an attack could be successful without detection, which could be very destructive knowing the nature of APT attacks and their consequences. Moreover, the highest result given was by the SVM, which does not yield good results when dealing with high volumes of data.

The combination of recognition patterns and ML to resist and mitigate any attack was used by [10]. A dataset was used with over 1 million PCAP cases to analyze and evaluate their approach. The highest accuracy was achieved by the RF classifier. The limitation of this research is that it collected the data over a week, which could not be very conclusive when facing an APT attack. APT attacks are very persistent and take a long time to execute the attack, so one week of data collected would not serve the whole purpose of detecting these types of attacks.

An effective intrusion detection system using XGBoost classifier on the NSL-KDD dataset was presented by [11]. It achieves the best results based on several metrics: accuracy, precision, and the confusion matrix. It used all 41 features of the dataset in its model. It compared the proposed model with several ML classifiers, where it reached the highest accuracy of 98.7%. Their research was implemented on a dataset which does not cover any APT attacks.

Early warnings for an APT attack were presented by [12]. They used the NSL-KDD dataset. They used the principal component analysis approach to reduce the size of the dataset and improve the detection accuracy. The highest accuracy reached was 97.2% by the SVM. By relying on the open-source NSL-KDD data set, the authors could have affected the performance of their classifiers. This dataset is known for having some issues concerning the final results. There are also 41 features in this dataset, some of which have no values. Instead, the authors stated they used 122 features in their testing process. They could have added their features, potentially affecting the results.

A novel deep learning stack model to detect APT attacks was presented by [13]. They stated that they must use the raw network data in the detection process to capture such attacks. It proposed that one must identify the most significant factor in the attack to aid in detecting an attack, which is communications when it comes to APT. Moreover, must log the network stream to identify the attack. Their research doesn't have any results yet since it is still an ongoing process. The limitation here is when dealing with raw data, there will be time complexity problems. The system takes a significant amount of time to analyze raw data, which a security analyst wouldn't have on his side when dealing with APT attacks.

ML algorithms, namely Naive Bayes, SVM, and J48, have been used to detect and classify intrusion using the KDD-99 [14]. They selected their features based on information gained. Using the AdaBoost as a combo with the J48, it achieved an accuracy of 97%. The authors evaluated their approach to the KDD99 dataset. This dataset has many issues that could highly affect the performance of the detection process[15](Tavallaee et al. 2009).

An intrusion detection system to detect and predict APT attacks was proposed by [16]. Their system had two phases. In the first phase, the attack is reconstructed and correlates with the alerts belonging to a specific APT campaign. On the other hand, the second phase is interpreting the attack. Their system achieved an accuracy of 91.8% for two observations in estimating the order of the attack and 100% for a sequence of more than two observations. Their research relied heavily on predictions. The ultimate goal was to predict the next step of an APT attack or campaign. However, APT attacks have a life cycle they follow, but the phases of this life are not always the same and could differ, or there could be new methods that APT attack follows. The results achieved in this research of 100% is from studying and testing 361 APT alerts and 2300 uncorrelated alerts. When defending against an APT in the fourth stage of the campaign, the attack is already successful, and the detection process could be a bit too late to yield any benefit.

In [17] presented a cyber-kill chain approach for detecting APT; they created a data-driven approach for detecting APT stages utilizing the cyber kill chain using a public available APT dataset. The Naive Bayes classifier produced the best performance result of 91.1 %.

In [18] highlighted ML classifiers for traffic monitoring to identify the malicious activity as part of a network intrusion detection in software-defined networks. Three distinct traditional and modern tree-based ML classifiers, RF, DT, and XGB, were used to show the detection mechanism. Regarding training and testing, the NSL-KDD dataset is employed; this dataset is regarded as a benchmark for numerous state-of-the-art intrusion detection approaches. They achieved an accuracy of 95.9%. However, they implemented their research on a dataset that does not cover any APT attacks.

A multi-layer approach for advanced persistent threat detection using ML-based on network traffic was introduced by [19]. Their study employed three primary layers to detect APT attacks:

- Detecting APT attacks based on abnormal connection analysis.
- Detecting APT attacks based on studying and evaluating Suricata log analysis.
- Detecting APT attacks based on analyzing behavior profiles derived from layers.

They claimed that the research findings showed that the APT attack detection method based on analyzing and evaluating behavior profiles outperformed individual detection approaches.

APT attacks are among the most destructive on any network. Several researchers used free source datasets or ones they developed in a short time. APT assaults can go undetected for 169 days on a network and take 69 days to recover. Hence, to construct an effective APT detection methodology using ML, the dataset must be collected over a long time. This correlation will focus on TTP and IOCs used by known APTs to collect valuable logs that might integrate into an ML algorithm to improve the detection process as effectively as feasible with a low mistake rate. Researchers' cur-

rent open-source datasets don't cover the entire APT lifecycle, making detection less reliable.

In conclusion, the previous studies did not introduce a full APT attack lifecycle dataset to detect and mitigate APT attacks. Therefore, to implement a successful and an effective detection methodology using ML, the dataset used must be collected over a long period of time to correlate as much behavior as possible for an APT. This correlation will focus on TTPs and IOCs used by known APTs around the world to collect valuable logs that could be introduced to a ML algorithm to make the detection process very effective with low error rate as possible. The open source datasets used by the researchers in recent years do not cover a full APT lifecycle, making the detection process not as dependable as it should be.

3. Conceptual framework and methodology

This section describes the proposed APT detection model architecture methodology, the approach used to accomplish prospective objectives, and its main phases.

The proposed detection approach has several stages. The first stage is data collection and pre-processing; the second stage is dataset classification. Finally, an ML detection model is used to test the dataset. The APT detection mechanism and main pathway are as follows:

- Collect the data.
- Implement data pre-processing.
- Extract features and implement feature selection.
- Split data into training and testing portions.
- Build a model, and evaluate it.

The ML techniques focus on creating an explicit or implied model that allows for the classification of variations in original data. Maybe use ML techniques for independent, hybrid, or ensemble classifiers. Can be classified classification model in use into three types of operation: supervised, unsupervised, and semi-supervised. In general, the supervised method surpasses the other techniques. ANN, KNN, Naive Bayes, Genetic Algorithm, SVM, Logistic Regression, and DT are ML algorithms used in intrusion detection systems. The ML model development process consists of four steps: data acquisition/collection, data pre-processing, model selection and training, and model assessment. Among the many tasks that can perform in the pre-processing data, stage are the two fundamental principles of this job, feature selection, and normalization[20]. Fig. 1 depicts the APT detection proposed model design.

The goal of the proposed approach will be a multiclass classification problem where the model will receive pure data labeled as normal, reconnaissance, initial compromise, lateral movement, and data exfiltration. The objective is to classify normal traffic as normal, and the other labels will be considered attacks. The attacks represent the stage the attacker has achieved in his attack. For example, if the log is deemed a lateral movement, the attacker is in the 3rd stage of the attack.

3.1. Data source and collection techniques

The data collection technique implemented in this research work is based on packet capture. APT attacks are conducted using their TTPs and IOCs to build such a dataset. Exercising APT group techniques will also help researchers analyze and comprehend them. This research spent four months collecting data on real-world cyber network architecture. And provided a base of users with user/admin credentials early in the collecting phase to simulate regular/normal traffic. During this time, the company ran as usual. For example, a network admin could rearrange data, directories, and users or update one of the website's data. There was no malicious traffic on the network for the normal traffic to have a baseline.

Now that the network traffic baseline is known, certain attacks are launched to gather it. Separate APT group attacks were launched on the network. Each attack was carried out separately over time. The purpose of each attack varies. Aims of each attack included reconnaissance, compromise, lateral movement, and data exfiltration. Fig. 2 depicts the systems utilized to acquire the data for this study.

In the above figure, the network has several data collection systems. The attacker (tester) conducts APT attacks on the network, and each system logs each attack or maneuver. The API network aggregates the network's logs via Logstash. It also connects incompatible applications to make them work together.

Several resources were used to acquire data. First, CTI free and commercial sources were used. CTI provides excellent insight into recent APT attacks, IOCs, and TTPs. Second, SIEM solutions were used to correlate APT attack data and behaviors. This stage utilizes ArcSight Security Information and Event Management (SIEM) and Moloch (Arkime).

The SIEM and Moloch will collect data from actual APT attacks. They have executed some attacks during the threat-led APT PT approach phases [1], which APT groups track. The attacks used several ways to obtain data on APT threats. Techniques used to collect attack traffic: CTI, ArcSight, Moloch (Arkime), Wireshark, TCP dump, and network miner. The collection process focused on normal traffic to establish a baseline for the network and understand what normal traffic looks like. Then launched, different APT attacks correlate the network traffic generated. Table 1 shows the APT attack vectors employed in this study. Each attack will represent the attacker's mission stage and the techniques used in each stage:

Various traffic logs from attacks and normal operations were obtained. In total, 42 thousand traffic logs were acquired and labeled based on their normality or attack scenario.

3.2. Data pre-processing

First, the entire dataset will be normalized before adding data mining methods to the dataset to achieve better outcome. Data pre-processing involves removing duplicates, missing values, and normalizing data. This research used the complete case analysis method to handle missing values.

In a dataset with a wide range of numeric values, normalization reduces the range of the values to a common scale. Normalizing data balances all attributes. For classification techniques employing neural networks or nearest-neighbor, normalization speeds up the model training stage. Some popular normalization techniques include minimum–maximum and decimal scaling. To normalize the data, we utilize Min-Max normalization normalize all the numeric features within a range of 0 to 1 using the a below equation [20].

$$xscaled = (x - \min(x))/(\max(x) - \min(x))$$

3.3. Feature extraction

Feature extraction refers to approaches that choose or integrate elements to generate features, hence minimizing the quantity of data that should be handled while properly and thoroughly characterizing the actual dataset.

Labeling the data should be done upon extracting the features and generating the CSV file. Label the data for every flow by referring to the attack scenario timeline, the IP addresses, ports of the source and destination, and the protocols used. The dataset col-

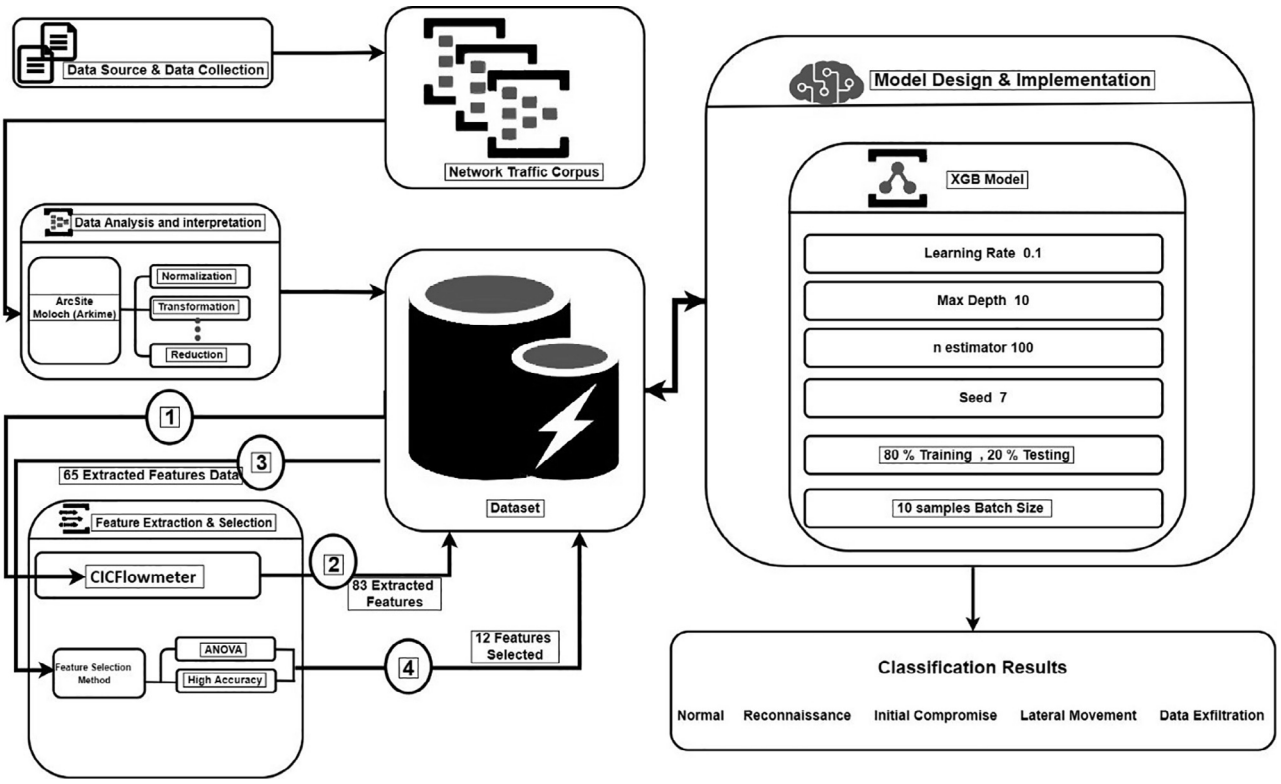

Fig. 1. APT Attack Detection Proposed Model Design.

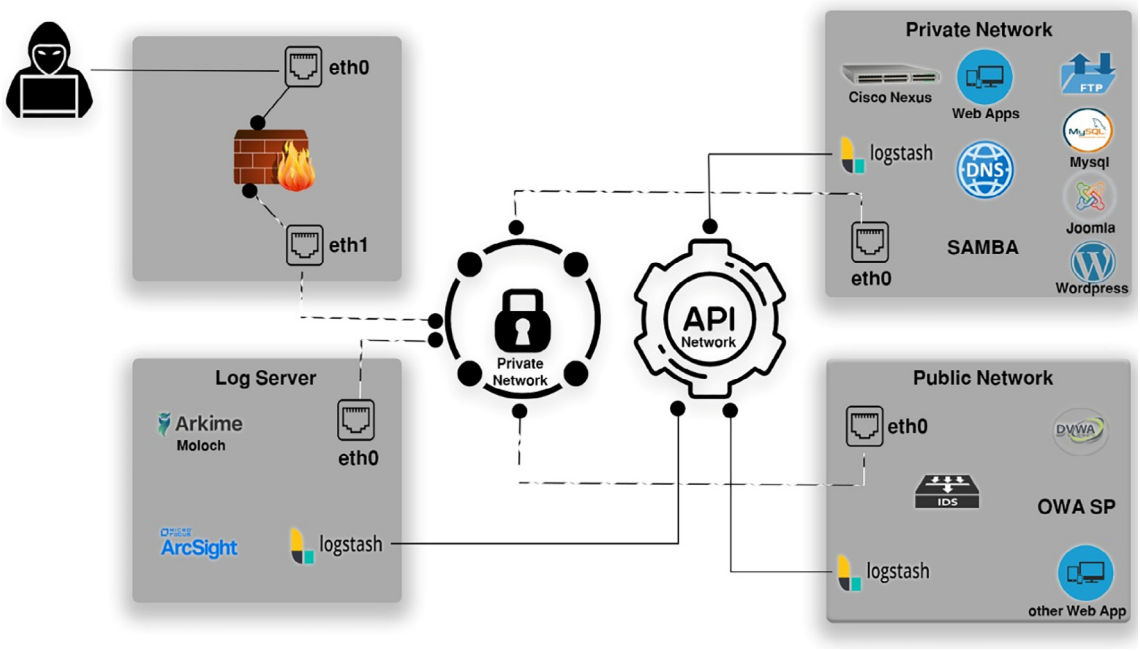

Fig. 2. Systems for Data Collection.

lected has 42,000 records, as depicted in Fig. 3. The dataset comprises multi-stages/classes, namely, Normal, Reconnaissance, Initial Compromise, Lateral Movement, and Data Exfiltration.

The features extracted were 83 network traffic features. The name and a description of each feature are presented in Table 2. A features heat map is implemented before the feature selection process to become familiar and get a summary of the main features through visual methods. A further step was taken to better understand and interpret the heat map by plotting the overall feature importance. Feature importance describes a set of strategies for giving ratings to given features in a classification model, indicating the comparative importance of each attribute when producing a

Table 1
APT Attack Vectors.

Attack Vector	Description	Techniques Used	Stage
Normal	Establish a baseline for normal traffic.	Ping, Get, Post, browse, downloads, curl, dig, uploading	**Stage 0**
Reconnaissance	Reconnaissance on the public network, directory structures, vulnerability identification, poor authentication/authorization	Nmap, Nessus, Burp Suite, Web Scarab, ipsweep, port sweep, sqlmap, Nikto, dirbuster	**Stage 1**
Initial Compromise	Establishing a foothold into the network via different techniques, Directory Brute force, SQL Injection and Downloading Malware	Key Loggers, Poison Ivy, Guess-Password, Send mail, Scanbox, Ftp-Write	**Stage 2**
Lateral Movement	Discovery of the internal network through compromised systems and taking control of critical devices	Key Loggers, privilege escalation, Buffer Overflow, Metasploit, Nmap	**Stage 3**
Data Exfiltration	Moving data from local machines on the network to remote servers, locations, users.	ftp, email, SSH, SFTP, google drives	**Stage 4**

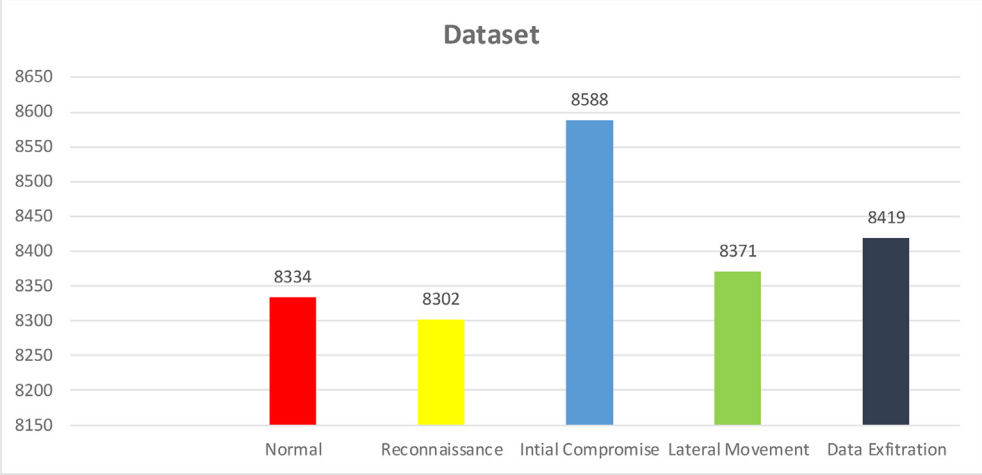

Fig. 3. Collected Dataset.

Table 2
Extracted Features.

#	Feature Name	Description
F1-F2	Fl.ID, Fl-dur,	Flow ID, Flow Duration
F3-F6	(Src,Dst) IP, (Src, Dst)port	IP address (Source/destination), port (source/destination)
F7-F8	Timestamps, Protocol	Duration of Flow, Time the event was recorded, Protocol used
F9-F10	Tot-Fwd-Pkts, Tot-Bwd-Pkts	The total packets forward/backward direction
F11-F12	TotLen-FwdPkts, TotLen-Bwd-Pkts	The total size of packets going in forward/backward direction
F13-F20	Fwd-Pkt-Len-Max, Fwd-Pkt-Len-MinFwd-Pkt-Len-Mean, Fwd-Pkt-Len-StdBwd-Pkt-Len-Max, Bwd-Pkt-Len-MinBwd-Pkt-Len-Mean, Bwd-Pkt-Len-Std	The (Max/min/avg./standard deviation/mean) Packet size in (Forward/backward) direction
F21-F22	Flow-Byts/sec, Flow-Pkts/sec	Number of packets transmitted per second (flow byte/packets rate)
F23-26	Flow-IAT-avg, Flow-IAT-StdFlow-IAT-Max, Flow-IAT-Min	Average/standard deviation/max-min time between two flows
F27-F36	Fwd-IAT-Tot, Fwd-IAT-avgFwd-IAT-Std, Fwd-IAT-MaxFwd-IAT-Min, Bwd-IAT-TotBwd-IAT-avg, Bwd-IAT-StdBwd-IAT-Max, Bwd-IAT-Min	The total/average/standard deviation/maximum/minimum/time between two packets that are sent in (Forward/Backward direction)
F37-F40	Fwd-PSH-Flags, Bwd-PSH-FlagsFwd-URG-Flags, Bwd-URG-Flags	Number of times (URG/Push) flags are set to 1 (0 for UDP) in (forward/backward) direction
F41-F44	Fwd-Header-Len, Bwd-Header-LenFwd-Pkts/sec, Bwd-Pkts/sec	Total bytes used for headers forward/backward direction.Number of forward/backward packets each second.
F45-F49	Pkt-Len-Min, Pkt-Len-MaxPkt-Len-Mean, Pkt-Len-StdPkt-Len-Var	(Minimum, Maximum, Mean, Standard deviation) length of a flow. Minimum inter-arrival time of packet
F50-F57	FIN-Flag-Cnt, SYN-Flag-CntRST-Flag-Cnt, PSH-Flag-CntACK-Flag-Cnt, URG-Flag-CntCWE-Flag-Count, ECE-Flag-Cnt	Number ofpacketswith (FIN/SYN/RST/PUSH/ACK/Urg/CWE/ECE) flags.
F58- F59	Down-Up-Ratio, Pkt-Size-Avg	Download and upload ratio, Average size of packet.
F60-F61	Fwd-Seg-Size-Avg, Bwd-Seg-Size-Avg	Average size observed forward/backward direction.
F-62-F67	Fwd-Byts-blk-Avg, Fwd-Pkts-blk-AvgFwd-Blk-Rate-Avg, Bwd-Byts-blk-AvgBwd-Pkts-blk-Avg, Bwd-Blk-Rate-Avg	Average number of (bytes/packets/bulk rate going forward/backward direction.
F68-F71	Subflw-Fwd-Pkts, Subflw-Fwd-BytsSubflw-Bwd-Pkts, Subflw-Bwd-Byts	The average number of (packets/bytes) in a sub flow forward/backward direction.
F72- F73	Init-Fwd-Win-Byts, Init-Bwd-Win-Byts	Number of bytes sent in initial window forward/backward direction.
F74	Fwd-Act-Data-Pkts	Number of packets with at least 1 byte of TCP data payload forward direction
F75	Fwd-Seg-Size-Min	Minimum segment size observed forward direction
F76- F79	Active-Mean-avg, Active-StdActive-Max, Active-Min	(Mean, Standard deviation, Maximum, Minimum) time a flow was active before becoming idle.
F80- F83	Idle-Mean-avg, Idle-Std Idle-Max, Idle-Min	(Mean, Standard deviation, Maximum, Minimum) time a flow was idle before becoming active.

prognosis. The ratings are valuable and could be used to interpret the data further and lower the number of input features in a prediction model issue in various contexts.

3.4. Feature selection

Feature selection is considered to be one of the essential phases in ML, as it could significantly affect the efficiency and performance of an ML model [20]. However, before implementing any feature selection techniques, one further step of data cleaning is performed. Four features were removed: flow ID, source IP, destination IP, and timestamp. These features could affect the classification process by making the classifier biased towards them and data cardinality. In some previous research, the destination port is also removed from the data. But since this research deals with APT attacks and data exfiltration, this feature must be included.

Additionally, according to [21], the destination port could be helpful in the classifier used. Data exfiltration techniques could use specific port numbers that need to be monitored and included in the detection model. In addition to the four features that were eliminated, the following fourteen features depicted in Table 3, which had a value of zero per instance, were removed as well:

Once filtered out all unwanted features, reduced the newly formed features to 65 features in total. The inclusion of all features in a dataset may cause dimension problems. Large feature sets lead to overfitting and complexity. To reduce the dimensionality in this research, six features selection methods were chosen considering their popularity, effectiveness, and complexity reported in previous interrelated research [22,23]. The methods used included ANOVA, chi-square, Forward Feature Selection (FFS), Recursive Feature Selection (RFS), XGB, and Lasso. In addition to using the whole subset of features in one of the experiments.

3.5. Classification methods

Four machine-learning classifiers were investigated in this research: RF, DT, XGB, and KNN. All the feature selection and classification methods were implemented using scikit-learn package in python. A cross-combination approach was utilized to evaluate feature selection and classification efficiency. Each classification algorithm was combined with all six feature selection methods. The final result was a total of 28 features selection and classification technique combinations were implemented.

Table 3
Constant Features.

1. Fwd-PSH-Flags	8. Bwd-Pkts-blk-Avg
2. Fwd-Pkts-blk-Avg	9. CWE-Flag-Count
3. Fwd-URG-Flags	10. Bwd-Blk-Rate-Avg
4. Fwd-Blk-Rate-Avg	11. ECE-Flag-Cnt
5. Bwd-URG-Flags	12, Init-Fwd-Win-Byts
6. Bwd-Byts-blk-Avg	13. Fwd-Byts-blk-Avg
7. URG-Flag-Cnt	14. Fwd-Seg-Size-Min

3.6. Comparison of feature selection and classification methods

An 80/20 percentage split was used to measure the detection effectiveness of feature selection and classification algorithms. We tested 28 features selection and classification methods. Table 4 depicts the accuracy results and the number of features used from each approach. This study's purpose was to improve accuracy while reducing features. The XGB classifier with the ANOVA feature selection approach scored the most excellent accuracy utilizing only 12 features out of 65 in the dataset.

The ANOVA feature selection method is known for selecting the best feature subset from a dataset. It could be a suitable criterion for selecting distinctive features in network traffic data.

As shown in the proposed model design, the first step was to gather data through various sources to build a corpus of network traffic. Then data analysis and interpretation were implemented using different techniques, namely ArcSight, Moloch (Arkime), and CICFlowmeter, to implement data preprocessing techniques. Once built the dataset, feature extraction was the next step, where the CICflowmeter utility was used to extract 83 features. Then, after further data cleaning, some features were dropped to have 65 total features left ultimately. These features were given to the ANOVA feature selection method to get the best subset of 12 features in the end. Table 5 displays the ANOVA-selected features.

The next step was to load the data into the XGB classifier-based APT detection model. For the XGB classifier, either adjust parameters before applying the dataset or use the default parameters. [11] (Dhaliwal, Nahid, and Abbas 2018) In this research work, the following parameters were applied [24]:

- The learning rate, often known as the eta parameter, was set as 0.1. It is used to prevent overfitting. It shrinks the step size, as well as the weights for the additional features, which may be easily extracted.
- The max depth option, which was set to 10, is utilized to specify the depth of a tree: the greater the value, the more complicated the model is.
- The number of cycles or trees employed in the model is specified by the n estimators' parameter, which was set to 100.
- The seed parameter was set to 7. It is a learning parameter also known as a random state.
- The n-splits or batch size parameter was set as 10. It is employed to split the dataset into several k parts.

The percentage split is a method of splitting data and segregating × percent of it for training from the remainder for testing. To run a more significant distinction and minimize bias in the dataset, an 80–20% ratio for training and testing techniques was applied. The training data consisted of (33611) events, and the testing consisted of (8403) events.

Table 4
Feature Selection and Classification Methods Comparison.

Feature Selection Method		RF		DT		XGB		KNN	
		Acc.	Number of Features	Acc.	Number of Features	Acc.	Number of Features	Acc.	Number of Features
–	No Selection	99.6	65	99.6	65	99.8	65	99	65
Filter	ANOVA	99.8	18	99.7	20	**99.8**	**12**	98.8	27
	Chi Square	99.8	31	99.6	31	99.6	27	98.9	30
Wrapper	FFS	99.8	18	99.8	21	99.7	18	99.2	28
	RFS	99.8	19	99.7	21	–	–	–	–
Embedded	XGB	99.2	28	89.6	31	99.7	28	91	30
	Lasso	99.8	47	99.7	47	99.8	47	98.6	47

Table 5
Selected Features By ANOVA.

1. Src-port	7. Flow-IAT-Max
2. SYN-Flag-Cnt	8. Fwd-Act-Data-Pkts
3. Dst-port	9. Fwd-IAT-Tot
4. ACK-Flag-Cnt	10. Idle-Max
5. Fl-dur	11. Fwd-IAT-Max
6. Init-Bwd-Win-Byts	12. Idle-Min

4. Results and discussion

In this work, an APT detection approach using ML is proposed utilizing a newly created dataset. To evaluate the effectiveness of the proposed model and the new dataset, a comparison with several machine learning algorithms was conducted. In addition, the built dataset was applied to a previously used approach to test its validity.

Firstly, the proposed model was compared to three well-known classifiers, namely RF, DT, and KNN. For a fair comparison, the number of features applied for each classifier will be the same. The XGB was able to achieve the best results using the least number of features are12, when applying the ANOVA features selection method. Therefore, for this evaluation 12 features for each of the other classifiers for comparison will be used. The measurements are evaluated by applying the confusion matrix, Accuracy, Precision, Recall, and F-Measure.

Secondly, apply the created dataset to a model presented in previous research [11]. They built a model by employing an XGB classifier on the NSL-KDD dataset. The parameters used in their model were learning rate set to 0.1, max depth was set to 3, n-estimators was set to 100, seed or random state was set to 7, and batch size to 10. The entire dataset with all 65 features was applied to their model. The results of this evaluation were accuracy 98.7%, precision 98.5%, recall 98.3%, and F1 measure 98.5%. It is clear from these results that the created dataset is valid and can be used for APT attack detection.

To get an intuitive view of the main issues of the suggested model at hand the significant evaluation criteria are obtained; for comparing the performance of the proposed model, accuracy, precision, recall, and F1 measure are used [25]. Accuracy indicates the model's accuracy rate or the percentage of total instances classified correctly by the classifiers. Precision is the correctly classified instances by a classifier. The recall is the percentage of relevant instances obtained compared to the total number of relevant instances. The F-measure is a metric for determining how accurate a test is. It's the sum of recall and precision in a harmonic form.

All metrics, measurements, and evaluation criteria were obtained from the PyCM version 3.1. PyCM is written in Python language and is a multiclass confusion matrix library [26]. The measurements are evaluated by applying the confusion matrix giving four possible results True Positive (TP), False Positive (FP), True Negative (TN), and False Negatives (FN). Fig. 4 is a sample of a multiclass classification confusion matrix.

ML, a confusion matrix is a breakdown of prediction outcomes for a multiclass classification problem. It represents the value of the output of a classifier on any dataset. It is one of the most straightforward and obvious measures for determining a model's accuracy and efficiency. It's used to solve classification problems where the outputs can be divided into two or multiple classes [20]. The diagonal components reflect correct or valid classifications, while the other components reflect the classification model has incorrectly classified ones. As a result, the greater the value of the confusion matrix's diagonal elements, the more accurate and better the classification model grows.

For the proposed model, a comparison was performed to evaluate the efficiency of all three algorithms. The same data set has been used with 12 features. The confusion matrix of the proposed model is depicted in Fig. 5.

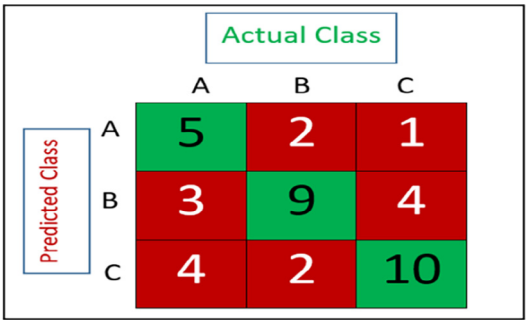

Fig. 4. Multiclass Classification Confusion Matrix Sample.

Concerning the confusion matrix, it is clear that the proposed model performed well in classifying each instance to the appropriate stage, with only a few instances being miss classified. Table 6 summarizes the multiple classification algorithms analysis results obtained.

TP prediction is one that detects a condition when it already exists. Whenever a TN test outcome is obtained, the condition was not detected when the condition was not present. FP prediction occurs whenever a condition is detected, even when it's not present. When a model fails to detect a condition, it is called a FN. These metrics can calculate the True Positive Rate (TPR), False Positive Rate (FPR), True Negative Rate (TNR), and False Negative Rate (FNR).

Since this model predicts multiclass, Table 7 presents the results for each class/stage. For a good classifier, the TP and TNR should be closer to 100, whereas for the FP and FNR, its better when they are closer to 0.

From the above table, it can be observed how accurate the proposed model is in predicting each class/stage. The detection accuracy rate per stage for the proposed model is depicted in Fig. 6.

The overall metrics achieved will be analyzed to evaluate the proposed model's metrics. Because the number of classified instances varies, accuracy alone may not necessarily give the whole picture. Thus, the proposed model's evaluation metrics results will be charted. Fig. 7 compares classification algorithms' accuracy, recall, precision, and F-Measure. The proposed model achieved the highest in each criterion, followed by RF, then DT, while the KNN achieved the lowest.

In the case of accuracy, for the proposed model, the accuracy with the best rate is 99.89%. In comparison, the KNN could oust the least percentage of accuracy, 96.59%. RF scored the second-highest, 98.75%. Finally, 97.65% was obtained by DT, 98.7 was achieved by Dhaliwal, Nahod and Abbas approach (2018), and 91.1 was achieved by Ahmed, Asyhari and Rahman approach (2021).

The highest results were obtained for recall, precision, and f-measure results by the proposed model recall 99.89%, precision 99.89%, and f-measure 99.89%. Followed by RF recorded a 98.7% precision ratio, and for F-measure and recall, 98.75%.

Given that recall, precision, and f-measure are all related, a detailed analysis of the results is required to demonstrate the model's performance. The focus was on the relations between the results and their significance. Finally, what are the main factors leading to them?

Accuracy indicates the ratio of actually correct predictions. Nevertheless, it does not give detailed information regarding its test. Achieving a high accuracy may seem significant at first sight, but another classification model may achieve the same or better accu-

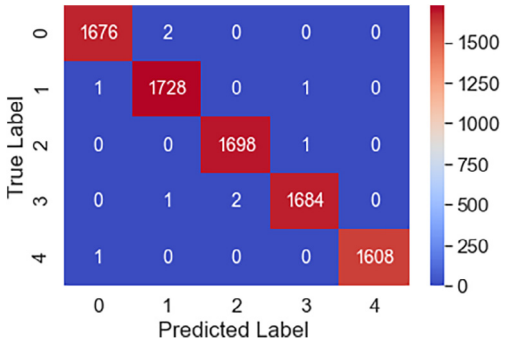

Fig. 5. Proposed Model Confusion Matrix.

racy. In other words, the proposed model is not better than one with a lower predictive ability to classify APT attack traffic from normal traffic. To give a better answer, one needs to know about the recall, precision, and f-measure. However, accuracy, in general, informs us instantly whether a model is being properly trained and how it generally performs. This ratio indicates the compatibility and appropriateness of the components of the model.

Regarding recall and precisions. Recall tells out of actual positive data how many times you predicted correctly. It also helps when the cost of false negatives is high. Precision informs that When the model predicts positive, how often it's correct. Precision aids when the totals of false positives are high.

F-Measure is a cumulative calculation of the prediction accuracy that incorporates recall and precision. A strong F-Measure result shows that you get lower FP and lower FN, so you are accurately detecting actual threats and not distracted by any false alarms. The F-Measure is deemed flawless once it is 1, whereas the design is a failure whenever it's 0. F-measure is a requirement when a balance between Precision and Recall is sought.

Based on the previous discussions and results regarding the proposed model. From the 99.89% accuracy ratio obtained, it is concluded that it has a high rate of correct predictions to classify APT attacks from normal traffic. Additionally, the model was properly trained, and its arguments and components were tuned and selected appropriately. Concerning 99.89% and 99.89% precision and recall ratio, respectively, it is presumed that the proposed model has a low FPR and FNR. In addition, the proposed model is designed and tuned with best-fit criteria. Finally, the results obtained by the proposed model are verified. This helps us decide

Table 6
Results of Proposed Model and Multiple Classification Algorithm.

	Accuracy	Precision	Recall	F-Measure	Error Rate
Proposed Model	**99.89**	**99.89**	**99.89**	**99.89**	**0.11**
RF	98.75	98.70	98.75	98.75	1.25
DT	97.65	97.64	97.65	97.65	2.35
KNN	96.59	96.59	96.55	96.55	3.41
Dhaliwal, Nahid and Abbas, 2018 Approach	98.7	98.5	98.3	98.5	1.3

Table 7
TPR, FPR, TNR, FNR.

Stage	TP	FP	TN	FN	TPR	FPR	TNR	FNR
Normal	1676	2	6723	2	99.88	0.0002	99.97	0.001
Reconnaissance	1728	3	6670	2	99.88	0.0004	99.95	0.001
Initial Compromise	1698	2	6702	1	99.94	0.0002	99.97	0.0005
Lateral Movement	1684	2	6714	3	99.82	0.0002	99.97	0.001
Data Exfiltartion	1608	0	6794	1	99.93	0	100	0.0006

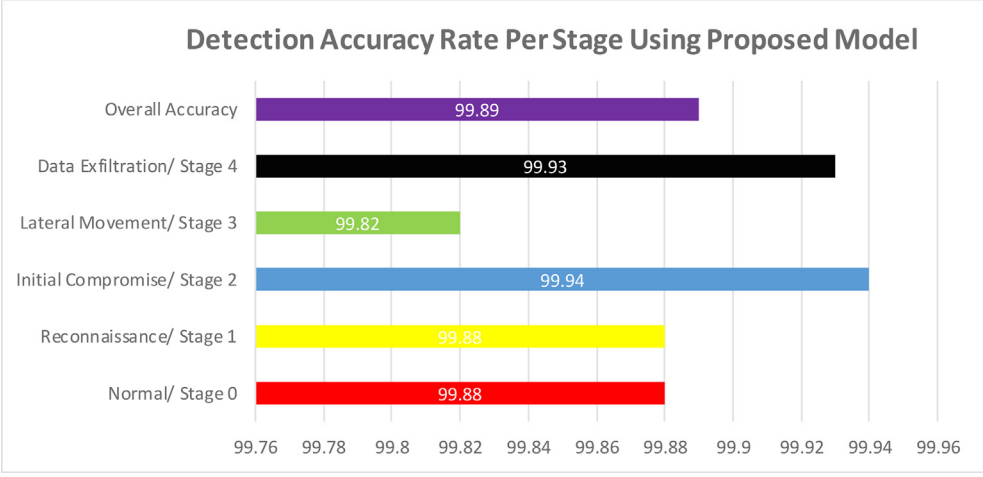

Fig. 6. Detection Accuracy Rate Per Stage for Proposed Model.

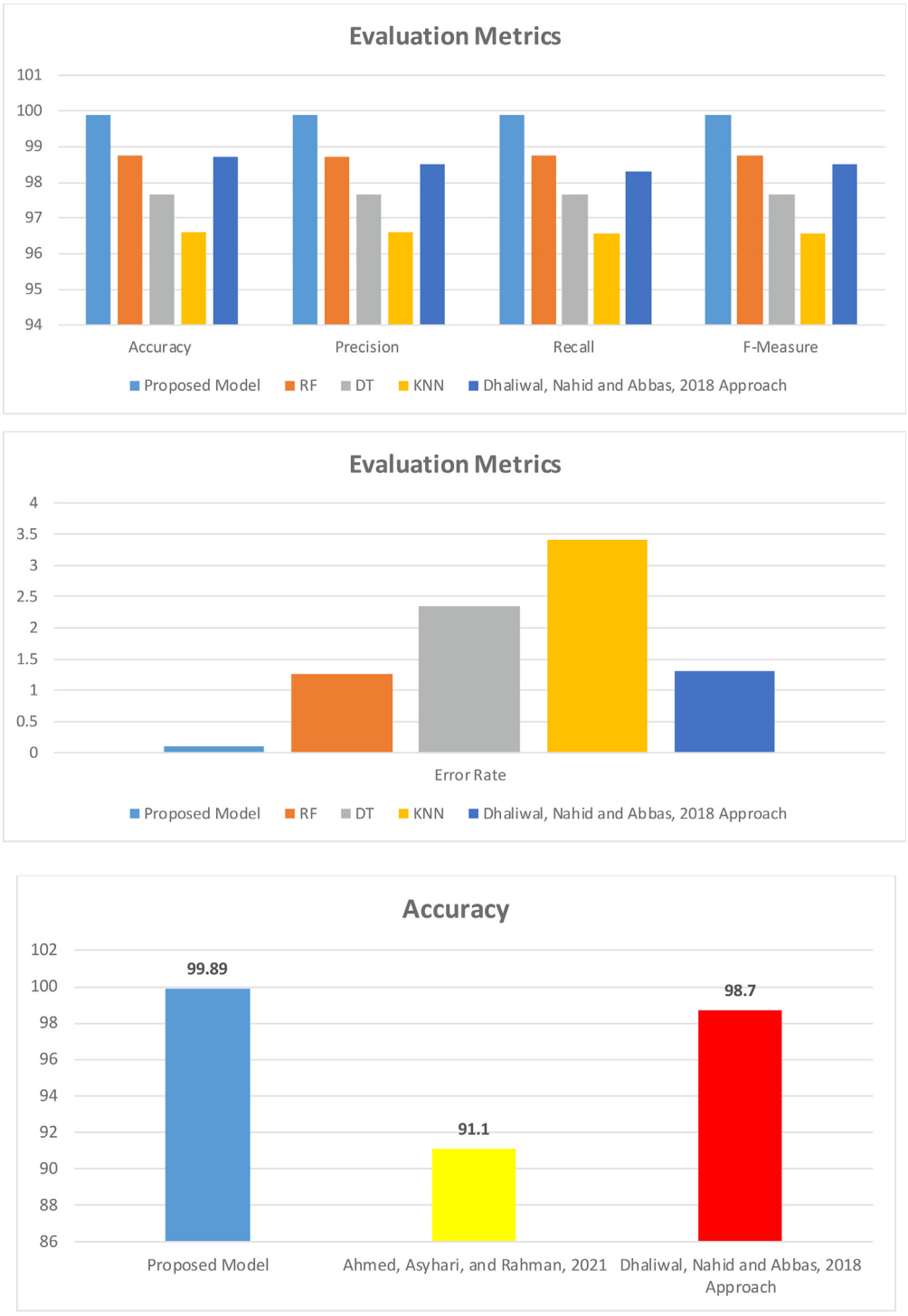

Fig. 7. Evaluation Metrics.

the suitability and ability of this model to operate efficiently as an APT attack detection model.

Regarding the Dhaliwal approach, the accuracy of their model was less than the proposed model at 98.7% when applying the same dataset, which was due to the reason of changing the hyper-parameters of the XGB, which led to lower results. As per the error rates for the proposed model and the other methods, the proposed model achieved the lowest rate at 0.11%, RF at 1.25%, DT at 2.35%, KNN had 3.41%, and finally, the Dhaliwal approach achieved 1.3%.

As mentioned before, the proposed model is based on the XGB classifier was able to record the highest results using the least number of features. XGB was primarily developed for performance and efficiency by employing gradient boosting to decision trees. It represents the process for applying boosting to machines, which was introduced by [27] and has since been adopted by many developers [11]. For tree boosting methods, XGB improves in maximizing memory and computing resources. It has the advantages of improving the algorithm and modifying the model, and it can also

be used in a computing environment. Three types of boosting can be done by the XGB regularized, Gradient, and Stochastic boosting. It is very efficient in lowering computation time and making the best use of memory resources. It can handle missing and null values, performs concurrent tree build, and has the distinct ability to execute boosting on data that has already been included in a classifier [11,18].

The infrastructure in which an attack detection system is installed is perhaps the most significant factor. The sort of data entry in the detection system is closely tied to the environment in which it is used. That's when XGB excels because of its adaptability in deployment. XGB may accept a variety of data sources as inputs and therefore can operate on multiple operating systems at the same time. Furthermore, XGB effectively handles data overfitting issues, and it provides the option to use Decision Tree Algorithms [21].

Researchers have long strived to identify the best classification method by comparing different algorithms. Compared to traditional Boosted models, XGB enhances performance by a factor of ten, making the classification model stronger because speed is vital in a network attack detection model. In past years, researchers have explored XGB and other classification methods on datasets to discover the best classification model. XGB has the edge over the different categorization algorithms based on the results [11,21].

5. Conclusion and future work

In this research work, an approach to detect APT attacks was presented using a newly built APT attack dataset. The dataset was deployed into a proposed ML model to detect APT attacks based on different attack types. Five types of data were collected, namely, normal, reconnaissance, initial compromise, lateral movement, and data exfiltration. Each data type represents a stage that the attacker might have gotten to. The APT attack ML detection proposed model was built over the XGB classifier with the ANOVA feature selection method. An outstanding performance was achieved with a detection accuracy of 99.89% using only 12 features out of the 65 features in the dataset. The proposed model was compared with other known ML classifiers, which proved superior to them based on the results gained.

As per the future work, Further studies will be conducted to improve this step by enhancing the model to use multi-label classification. Therefore, enhancing the proposed APT detection approach to use a multi-label technique could give more insight into the attacks that could be carried out against a network.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Masarweh A, ALSaraireh J. Threat Led Advanced Persistent Threat Penetration Test. Int. Journal Secur. Networks 2021;16(4):239–57.

[2] K. Xing, A. Li, R. Jiang, and Y. Jia, "A review of APT attack detection methods and defense strategies," *Proc. – 2020 IEEE 5th Int. Conf. Data Sci. Cyberspace, DSC 2020*, pp. 67–70, Jul. 2020, doi: 10.1109/DSC50466.2020.00018.

[3] Steffens T. Attribution of Advanced Persistent Threats. 2020.

[4] Al-Tarawneh A, Al-Saraireh J. Efficient detection of hacker community based on twitter data using complex networks and machine learning algorithm. J. Intell. Fuzzy Syst. Jan. 2021;40(6):12321–37. doi: https://doi.org/10.3233/JIFS-210458.

[5] I. Goodfellow, Y. Bengio, and A. Courville, "deep learning English version," p. 800, 2017.

[6] Gauthama Raman MR, Somu N, Kirthivasan K, Liscano R, Shankar Sriram VS. An efficient intrusion detection system based on hypergraph – Genetic algorithm for parameter optimization and feature selection in support vector machine. Knowl-Based Syst 2017;134:1–12.

[7] Kim J, Shin N, Jo SY, Kim SH. "Method of intrusion detection using deep neural network", *2017 IEEE Int.* Conf. Big Data Smart Comput. BigComp Mar. 2017;2017:313–6. doi: https://doi.org/10.1109/BIGCOMP.2017.7881684.

[8] T. Schindler, "Anomaly Detection in Log Data using Graph Databases and Machine Learning to Defend Advanced Persistent Threats," *Lect. Notes Informatics (LNI), Proc. – Ser. Gesellschaft fur Inform.*, vol. 275, pp. 2371–2378, Feb. 2018, doi: 10.18420/in2017_241.

[9] Ghafir I, Hammoudeh M, Prenosil V, Han L, Hegarty R, Rabie K, et al. Detection of advanced persistent threat using machine-learning correlation analysis. Futur. Gener. Comput. Syst. 2018;89:349–59.

[10] Adelaiye O, Ajibola A, Faki S. Evaluating Advanced Persistent Threats Mitigation Effects : A Review. Int. J. Inf. Secur. Sci. 2018;7(4):159–71.

[11] S. S. Dhaliwal, A. Al Nahid, and R. Abbas, "Effective Intrusion Detection System Using XGBoost," *Information*, vol. 9, pp. 1–24, Jun. 2018, doi: 10.3390/INFO9070149.

[12] Chu WL, Lin CJ, Chang KN. Detection and Classification of Advanced Persistent Threats and Attacks Using the Support Vector Machine. Appl. Sci. Oct. 2019;9 (21):1–16. doi: https://doi.org/10.3390/APP9214579.

[13] T. Bodström and T. Hämäläinen, "A Novel Deep Learning Stack for APT Detection," *Appl. Sci. 2019, Vol. 9, Page 1055*, vol. 9, no. 6, p. 1055, Mar. 2019, doi: 10.3390/APP9061055.

[14] Mazraeh S, Ghanavati M, Neysi SHN. Intrusion detection system with decision tree and combine method algorithm. Int. Acad. J. Sci. Eng. Jun. 2019;06 (01):167–77. doi: https://doi.org/10.9756/IAJSE/V6I1/1910016.

[15] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set in Computational Intelligence for Security and Defense Applications," *Comput. Intell. Secur. Def. Appl.*, no. Cisda, pp. 1–6, 2009.

[16] Ghafir I, Kyriakopoulos KG, Lambotharan S, Aparicio-Navarro FJ, Assadhan B, Binsalleeh H, et al. Hidden markov models and alert correlations for the prediction of advanced persistent threats. IEEE Access 2019;7:99508–20.

[17] Ahmed Y, Asyhari AT, Rahman MA. A Cyber Kill Chain Approach for Detecting Advanced Persistent Threats. Comput. Mater. Contin. 2021;67(2):2497–513. doi: https://doi.org/10.32604/cmc.2021.014223.

[18] Alzahrani AO, Alenazi MJF. Designing a network intrusion detection system based on machine learning for software defined networks. Futur. Internet 2021;13(5):1–18. doi: https://doi.org/10.3390/fi13050111.

[19] Xuan CD, Duong D, Dau HX. A multi-layer approach for advanced persistent threat detection using machine learning based on network traffic. IFS 2021;40 (6):11311–29.

[20] M. A. Umar and C. Zhanfang, "Effects of Feature Selection and Normalization on Network Intrusion Detection," no. June, pp. 1–25, 2020, doi: 10.36227/techrxiv.12480425.

[21] Leevy JL, Hancock J, Zuech R, Khoshgoftaar TM. Detecting cybersecurity attacks across different network features and learners. J. Big Data 2021;8(1):1–29. doi: https://doi.org/10.1186/s40537-021-00426-w.

[22] Liu Z, Liu Q. Balanced feature selection method for Internet traffic classification. IET Networks 2012;1(2):74–83. doi: https://doi.org/10.1049/iet-net.2011.0049.

[23] Oluranti J, Omoregbe N, Misra S. Effect of Feature Selection on Performance of Internet Traffic Classification on NIMS Multi-Class dataset. J. Phys. Conf. Ser. 2019;1299(1):1–11. doi: https://doi.org/10.1088/1742-6596/1299/1/012035.

[24] Ma X, Sha J, Wang D, Yu Y, Yang Q, Niu X. Study on a prediction of P2P network loan default based on the machine learning LightGBM and XGboost algorithms according to different high dimensional data cleaning. Electron. Commer. Res. Appl. 2018;31:24–39. doi: https://doi.org/10.1016/j.elerap.2018.08.002.

[25] Sree Kala T, Christy A. HFFPNN classifier: a hybrid approach for intrusion detection based OPSO and hybridization of feed forward neural network (FFNN) and probabilistic neural network (PNN). Multimed. Tools Appl. 2021;80(4):6457–78. doi: https://doi.org/10.1007/s11042-020-09804-7.

[26] Haghighi S, Jasemi M, Hessabi S, Zolanvari A. PyCM: Multiclass confusion matrix library in Python. J. Open Source Softw. May 2018;3(25):1–2. doi: https://doi.org/10.21105/JOSS.00729.

[27] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794, doi: 10.1145/2939672.2939785.