Advanced Persistent Threats Detection based on Deep Learning Approach

1st Hope Nkiruka Eke
School of Computing
Robert Gordon University, Aberdeen, UK
000-0001-5049-8212

2nd Andrei Petrovski
School of Computing
National Subsea Centre, Aberdeen, UK
0000-0002-0987-2791

2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS) | 979-8-3503-1125-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICPS58381.2023.10128062

***Abstract*—Advanced Persistent Threats (APTs) have been a major challenge in securing both Information Technology (IT) and Operational Technology (OT) systems. APT is a sophisticated attack that masquerade their actions to navigates around defenses, breach networks, often, over multiple network hosts and evades detection. It also uses "low-and-slow" approach over a long period of time. Resource availability, integrity, and confidentiality of the operational cyber-physical systems (CPS) state and control is highly impacted by the safety and security measures in place. A framework multi-stage detection approach termed *"APT_{DASAC}"* to detect different tactics, techniques, and procedures (TTPs) used during various APT steps is proposed. Implementation was carried out in three stages: (i) Data input and probing layer - this involves data gathering and pre-processing, (ii) Data analysis layer; applies the core process of *"APT_{DASAC}"* to learn the behaviour of attack steps from the sequence data, correlate and link the related output and, (iii) Decision layer; the ensemble probability approach is utilized to integrate the output and make attack prediction. The framework was validated with three different datasets and three case studies. The proposed approach achieved a significant attacks detection capability of 86.36% with loss as 0.32%, demonstrating that attack detection techniques applied that performed well in one domain may not yield the same good result in another domain. This suggests that robustness and resilience of operational systems state to withstand attack and maintain system performance are regulated by the safety and security measures in place, which is specific to the system in question.**

***Index Terms*—Advanced Persistent Threats, Cyber-Physical Systems, Critical Infrastructures, Deep Learning, Industrial Control Systems, Supervisory Control and Data Acquisition.**

I. Introduction

CYBER attack such APT has continuously evolving, posing a devastating security risk for governments and organizations alike. The intrusion detection and prevention systems (IDPSs) mechanisms that are available are not entirely able to address this type of attack [1]. Understanding APT constituent and its mode of operation will help towards continuous development of approaches that will protect systems or/ minimize APT attack impact.

A. APTs Traits

APTs and the actors behind them constitute a serious global threats. This type of attacks differs from commodity threats that seek to gain immediate advantage and are broad in their targeting and process [2]. APT on the other hand is very;

- ☐ *resourceful*
- ☐ *with well defined objectives and purpose*
- ☐ *uses sophisticated methods and technology*
- ☐ *substantially funded.*

The APT threat process follows a staged approach to find, penetrate and exploit their target. Understanding the ***advanced***, sophisticated tactics and ***persistent*** nature of APT is unavoidable in defending against an APT ***threat***.

- ■ ***Advanced* -** The advanced nature of APT provide the attackers with the capability of maintaining prolonged existence through stealthy approach inside an organization once they successfully breach security controls. Attackers uses sophisticated tools and techniques such as malware, if the malware is detected and removed, they change their tactics to secondary attack strategies as necessary [3].
- ■ ***Persistent* -** The "persistent" is referred to the acts of the attackers persistently launching spear-phishing attacks against their targets. The attackers obtain information and monitor network activity of the victim's networks by navigating from system to system, and adapt to be resilient against new security measures while maintaining a stealthy approach to reach the targets [4].
- ■ ***Threat* -** The threat actors also have the capability of gaining access to electronically stored sensitive information [5]. Other than the purpose of collecting of national secrets or political espionage, based on the functions discovered, it is believed that this threat can also apply to the cases in business or industrial espionage, spying acts or even un-ethical detective investigations [6]–[8].

B. An Overview of APT Lifecycle

APT attacks are generally utilises zero day exploit of unpublished vulnerabilities in computer programs or operating systems together with social engineering techniques to maximise the effectiveness of the exploits [5]. Launching an APT campaign involves numerous hacking tools, a sophisticated pattern, high level knowledge, varieties of resources and processes as an APT attack is not a single step attack unlike other attacks [9]. APT proved extremely effective at infiltrating their targets, going undetected for an extended periods, increasing their appeal to hackers who target businesses as mentioned in several large-scale security breaches [10]–[12].

Each APT attack is customised with respect to attacker's target and aim at each stage. The patterns of APT attacks are similar in most cases but differ in the techniques used at each stage to deliver. The description of six basic APT attack phases as shown in Figure [1], based on literature review in combination with the "Intrusion Kill Chain (IKC)" model described in [3], [13], [14] are as follows.

- ***Reconnaissance and Weaponisation:*** involves information gathering about the target. This could be, but not limited to, about organizational environment, employees' personal details, the type of network and defence measure in use. This information gathering can be carried out through social engineering techniques, port scanning and open source intelligence (OSINT) tools.
- ***Delivery:*** attackers utilise the information gathered from reconnaissance stage to execute their exploits either directly or indirectly to the targets. In direct delivery, the attackers use social engineering such as spear phishing by sending phishing email to the target. While in indirect delivery, attacker will first compromise a trusted third party, which could be a vendor or frequently visited website by target, and uses these to deliver an exploit.
- ***Initial Intrusion and Exploitation:*** attacker utilises the credentials obtained through social engineering to gain access to target's network. The delivered malware code is downloaded, installed and activated backdoor malware, creating a command and control (C&C) connection that links the target and the remote attacker's system. Once an attacker has secured connection to the target system, while attacker continuously gather more security related information such as security configuration and user names, while maintaining a stealthy behaviour in preparation for the next attack.
- ***Lateral Movement and Operation:*** At this stage, once the attacker has established communication between the target's compromised systems and servers, the attacker moves horizontally within the target network, identifying the servers storing the sensitive information of users with high access privileges. This is to elevate their privileges to access the sensitive data, making their activities undetectable or even untraceable due to the level of access they have.
- ***Data Collection:*** This stage involves utilizing the privileged users' credentials captured at the previous stage to gain access to the targeted sensitive data. With the attackers having a privileged access, they create redundant copies of C&C channels using sophisticated tools should there be any change in security configuration. Once the target information has been accessed, redundant copies are created at several staging points, where the gathered information is packaged and encrypted before exfiltration.
- ***Exfiltration:*** An attacker once gained full control of target systems, they proceed with the theft of intellectual property or other confidential data. Stolen information from a staging server is transferred to attackers external multiple servers, either in the form of encrypted packages, password protected zip files or even through clear web mails, as drop points. This idea of obfuscation strategy is to stop any investigation from discovering the final destination of the stolen data.

II. Related Works

The ability of an intrusion detection system to detect every possibility of an active attack on a system is a security issue. There have been a number of successful breaches of CI. Few examples of APTs attack are: - ***Stuxnet*** which is an APT attack aimed against uranium enrichment facilities in Iran [15], [16], ***Deep panda*** [17], [18], ***Epic Turla*** [19], [20] and the ***Oldsmar water treatment plant***, which exposed over 15,000 residents of Florida's west coast to potential poisoning [21].

Different techniques and frameworks have been proposed and successfully implemented to address these security issues [22]. These proposed approaches have led to a significant pool of solutions geared towards addressing security and systems resilience. One of this detection model is intrusion kill chain (IKC) model, created by Lockheed Martin analysts in 2011 to support a better detection and responding to attacker's intrusions. This model can serve as a great building foundation and a concept to start with [23].

However, it is very improbable to stop cyber breaches through prevention approach. Discovering an unusual behaviour within a system that may represent security breach in real time will provide an opportunity for the target organization to respond to such situation. According to [24], there exist different approaches to deal with specific threat, however, none of this approach is good enough to safeguard a system as there is always new evolving threats to deal with at any point in time. Hence, the author highlighted the important of developing " a multi-faceted, joined up approach" that posses breach detection capabilities, since security breach can only be managed effectively if discovered on time.

Deep learning (DL) techniques has been applied in cybersecurity [2] for attack detection. It can detect cyber threats by learning the complex underlying structure of security data. The authors [25], proposed RNN-based model against classical support vector machine classifier (SVM) for cybersecurity in Android malware classification. RNN emerged as a powerful approach for DL architecture generally applicable for time-series data modelling. Despite the RNN and its variant networks remarkable performance in long standing AI sequence data modelling tasks such as anomaly detection [26] and bone age assessment from X-Ray images [27]. Applying the same in cyber security task is in early stage of development [22].

A framework for ranking internal hosts as to identify and rank suspicious hosts that could be involved in data exfiltration activities related to APTs. By deploying a prototype of this framework, the author realized ranked list of suspicious hosts that could be involved in data exfiltrations and other APT related activities [28].

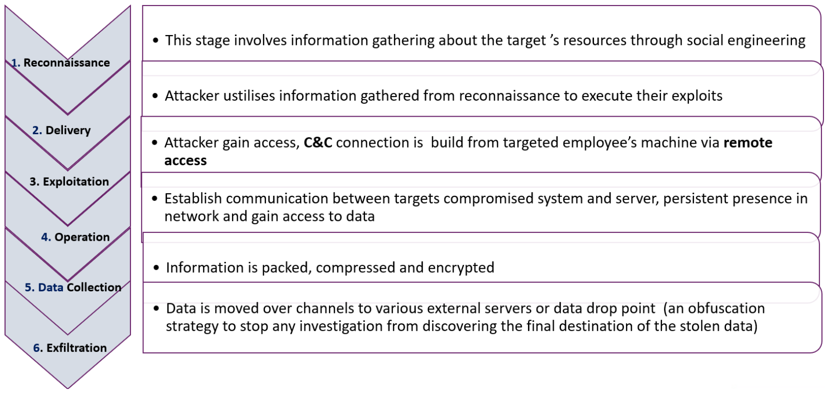
1. Reconnaissance
2. Delivery
3. Exploitation
4. Operation
5. Data Collection
6. Exfiltration
• This stage involves information gathering about the target 's resources through social engineering
• Attacker ustilises information gathered from reconnaissance to execute their exploits
• Attacker gain access, C&C connection is build from targeted employee's machine via remote access
• Establish communication between targets compromised system and server, persistent presence in network and gain access to data
• Information is packed, compressed and encrypted
• Data is moved over channels to various external servers or data drop point (an obfuscation strategy to stop any investigation from discovering the final destination of the stolen data)

Fig. 1. APT Lifecycle

Authors in [29] proposed DTB-IDS: an IDS based on decision tree using behavior analysis that can minimized APT attacks damage by executing quick detection of APT attacks. The authors evaluated their system using data collected from API features of malicious code behaviour and achieved accuracy detection rate of 84.7%.

Considering the impact of APT data availability for active decision and limited generalization for new threats constrain that many available solution relies on, [30] introduced domain adaptation APT malware log samples to target domain by fusing and leveraging various data sources to combat APTs using APT driven Bayes Net technique. This approach enables the use of system-based features that seek an indication of compromise within the APT-EXE[1] data from anomaly perspectives.

An assumption that if any stage of APT lifecycle fails, the entire plot to execute an APT attack will also fail, thus, [31], proposed an approach based on monitoring all accesses to unknown domains for detecting APT attacks. This approach was motivated by the hypothesis that user are been lured into downloading malware by redirecting user to fake domains with the intent for malware to spread once download is made, hence detecting such unknown domains comes with high accuracy.

APT attack follows an organised planned steps or stages as discussed in Sub-section [I-B]. Each individual stages can be detected through several probabilities. Authors in [32]–[34] suggested that system security can be improved by educating and creating awareness among system users & system administrator as well as web-based communities[2] users on different attack vectors thereby equipping them with necessary information and knowledge, thus contribute towards system protection.

III. APT Detection Framework Based on Deep APT Step Analysis and Correlation

The description and the rational behind APT_{DASAC} framework based on DL approach, the architectural deigned and implementation are presented.

A. Rational of Design

The rational behind the design are influenced by few factors; firstly, considering that APT attack is a multi-step attack of which each attack step is delivered through the use of a particular attack techniques. The detection of any single step of this attack techniques does not infer the detection of an APT scenario [22]. Thus, to detect and mitigate an APT attack, the detection system should be able to go through the detection of each of the techniques utilized at each step of the APT lifecycle, as highlighted in Sub-section [I-B]. Hence, the detection model should be designed and developed to have the capability to detect the most commonly used tactics at the earliest stage. That is, the model should be developed to be able to correlate and link the result of each detection modules as to determine any existence of APT attack scenario.

To explore implementation of a model DL-based approach to effectively detect attacks steps at it's earliest stage of an APT attack, thereby minimise the impact and preventing the execution of full APT scenario, thus pre-empt attackers from executing attack on their target system. To be specific, implementing stacked-ensemble RNN variants algorithm with generic settings which can be deployed in different domain and achieve the same purpose of safeguarding a system against any malicious intrusion. Different research has been proposed based on DL approach to analyse, detect and classify complex network traffic events. RNN emerged as a useful approach for DL architecture generally applicable for time-series data modelling. These provides motivation for work towards developing an approach for APT detection.

B. APT_{DASAC} Architectural Design

The architectural design of the proposed model for APT intrusion detection system (IDS) is built to run through three stages. This involves implementing a multi-step security detection approach based on DL, that takes into consideration the distributed and multi-level nature of the ICS architecture and reflect on the APT lifecycle for the four main SCADA cyber-attacks as suggested in [35]. Thus, APT detection system

[1]APT-EXE: https://github.com/aptresearch/datasets

[2]Web-based Communities: is an important place where people seek information and share expertise. It has attracted millions of users, many of whom have integrated these activities into their daily practices [33].

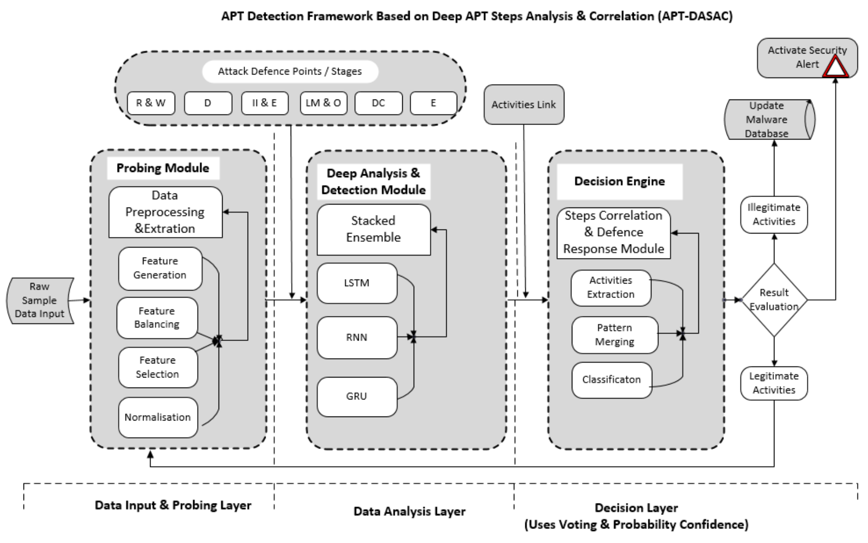

Fig. 2. APT detection framework based on deep APT step analysis & correlation (APT_{DASAC})

should be able to detect every single possible step applied by an APT attacker during the attack campaign.

The implementation stages of this model design as shown in Figure [2] are as follows:

Proposition 1: Implementation Stages of APT_{DASAC}

Stage 1 Data input and probing layer
Stage 2 Data analysis Layer
Stage 2 Decision Layer

C. Data Input and Probing Layer

Data input and probing Layer consists of two modules; (i) Data input and (ii) Probing module. Data gathering and pre-processing is an important fundamental process in data analysis to gather and convert raw data into a usable dataset. There are a good number of different pre-processing approaches, these includes but not limited to noise reduction, data cleaning, outlier removal and removing/replacing missing data. None of these approaches are simple task to accomplish, though very useful. The presence of noise may lead to model overfitting as a result of increase in the number of parameters due to the depth of Model. Most often, DNN are prone to this type of issue, where also a model that performed well on training dataset suddenly performs very poorly on test dataset.

- ***Data Input*** - involves data gathering, raw sample/simulated synthetic data been introduced into the system and transfer the collected data to probing module.
- ***Probing Module*** – involves data pre-processing and feature transformation which runs through four stages. Here all the data that has been collected and introduced into the module are encoded into numerical vector by the pre-processor ready to go through the neural network.

- ☐ ***Feature Transformation***: UNSW-NB15[3] dataset consists of 42 features with three of these features been categorical *(proto, service and state)* data. These features need to be encoded into numeric feature vectors as it goes into the neural network for analysis, classification, detection and prediction.
- ☐ ***Balancing Training & Testing Data Features***: This function is only invoked based on the nature of the features contained in both training and testing data of the domain network traffic information in use as the case of UNSW-NB15 dataset.
- ☐ ***Normalization***: At this stage, the *ZScore* method of standardisation is used to normalize all numerical features to preserve the data range, introduce the dispersion of the series, and to improve model convergence speed during training.

Proposition 2: Data Pre-processing

The ***pre-processing data*** stage takes raw network traffic data as an input from a specific problem domain, processes and transforms the data into a meaningful data format that the algorithm requires by converting any symbolic attributes into usable features, and deals with null values. The output from this stage is a ***new transformed data*** containing valuable information that the analyses stage will utilize.

D. Analysis Layer

At this layer, the *"Pseudocode for Sequence Data Training, Validation and Testing"* The pre-processed data or the network traffic is scanned and processed to detect possible attack tactics

[3]UNSWNB:https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/

used within an APT lifecycle. Two difficulty were encountered; (i) class distribution problem and (ii) classification of rare attacks.

The rate of attack detection is affected by the parameters used as these parameters have direct impact on attack detection. Based on this, several experiments with different network configuration were implemented to find the best optimal values for parameters such as number of epochs, batch_sizes, dropout, learning rate, network structure etc.

Again, considering these identified issues, to achieve a good detection rate for rare attack steps whilst maintaining overall good model performance, these two issues need to be carefully considered - the rare attack class distribution and the difficulty of correctly classifying the rare class. Firstly, when considering the class distribution, more emphasis should be placed on the classes with fewer examples. Secondly, attention should also be given to examples that are difficult to be correctly classified.

Taken these identified issues with class distribution and classification of rare attacks into account, the processed data are used to build the model that analyses and distinguishes attack(s) from normal network events. The result of this layer is passed to *"Decision Engine layer"*.

E. Decision Layer

This Layer operates in three phases; firstly, it receives information from the analysis layer and extract the attack step present. Secondly, it processes this information and links it to any previously identified attack steps that are related. Then, lastly, it uses voting and probability confidence to establish if the attack is a potential chain of attack campaign is found, and if it is consistent with other attack campaigns.

At this point, the attack impact is determined at this stage through the decision engine by correlating the output from the analysis layer using probability confidence to check for any presence of security risks. If an attack or security risk is present, it requests the defence response module to raise a security alert. This is checked with the previously detected step to see if this could be related to the newly discovered security risk alert. This is to reconstruct APT attack campaign steps, and hence highlights an APT campaign scenario if any, so that an appropriate action can be taken.

The impact of an attack can be considered as low depending on the attack activity stage. However, if this stage can be linked with other attacks step to show that it is part of that attack campaign, forming a full APT step cycle, then the impact at this stage can be considered as high. With this information in mind an appropriate response can be taken.

IV. Implementation of APT_{DASAC}

This section, presents the description of the platform and the approach taken to implement the APT_{DASAC} approach.

A. Implementation Setup

The implementation was carried to examine the performance of utilizing deep ensemble stacked RNN variants approaches in implementing APT_{DASAC} framework for APT attack steps detection.

The network topologies and payload information values of the NGP[4] dataset containing 214,580 Modbus network packets with 60,048 packets associated with cyber attacks are used. These attacks contains 7 different attack categories with 35 different specific attack types as explained in [36]. These attack categories align with APT lifecycle. The number of records in each of these categories and the main four types of attacks as contained in the NGP data. The batch size of 64 & 124, epoch between 300 to 500 are run with a learning rate set in the range of 0.01-0.5. All the 17 features of NGP data and 49 features of UNSW-NB15 data are used as input vector.

Furthermore, traditional ML classification algorithms such as *Decision Tree (DT)* was also explored. Result from this was compared to the result achieved from implementing *"stacked Deep ensemble RNN-LSTM variants"* in order to further evaluate the APT steps detection capability of implementing the proposed framework.

B. Hyperparameters Settings Used

"The rate of attack detection is affected by the parameters used as these parameters have direct impact on attack detection. Based on this, several experiments with different network configuration were implemented to find the best optimal values for parameters such as number of epochs, batch_sizes, dropout, learning rate, network structure etc".

Proposition 3: Hyperparameters Settings Used

- ☐ Batch_sizes: 64, 128 & 256
- ☐ Train_Test data_size: 67%_33%
- ☐ Learning rate: 0.01 to 0.5 with polynomial decay over all the epochs.
- ☐ Epochs: 100, 300 & 500 epochs.
- ☐ Neural Network Layer: Four layers was used (RNN-LSTM)
- ☐ Activation Function:
  - $*$ Each of the hidden layers has a ***sigmoid or ReLU*** activation function applied to produce non-linearity. This transforms the input into values usable by the output layer.
  - $*$ The ***softmax*** function is applied to the output layer to get probabilities of categories. This also helps in learning with ***cross-entropy loss*** function.
- ☐ Loss Function: ***categorical cross-entropy*** is applied as *loss function* to calculate the error rate.
- ☐ Optimizer: Adaptive Moment Estimation ***(Adam)*** optimizer is used for the back propagation to minimise the loss of *categorical-*

[4]NGP: https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets

cross entropy.

- ☐ Regularization: The ***dropout*** is used to alleviate the over-fitting (used as regularization technique to prevent over-fitting in neural networks. This randomly removes the units along with connections.

Considering the impact of a model hyperparameter setting, the experimental study focused on two task; the first task was to derive hyperparameter values for best model performance. The second task applied the derived hyperparameter values to measure the APT_{DASAC} performance.

The implementation framework is carried out in three stages. This proposed framework took advantage of supervised machine learning approach for classification and detection.

C. Preparing the Used Dataset

Deployed processes for preparing dataset can have a great impact on classification, detection and the quality of the developed model detection performance. However, it is important to understand the data type of the domain of application.

This study, developed a dedicated module for data processing called *process_data*. This *process_data* module applies different phases based on data input type to obtain a transformed data ready to be utilized to train model for classification, detection and prediction. These phases are:

- ☐ data collection
- ☐ data cleaning
- ☐ data scaling & normalization

The *first phase* involves information gathering from the designated source. In the *second phase - data cleaning*; majority of ML classifiers work well with numeric values, during the data transformation of the cleaning phase, any columns which are not numeric are converted into numerical value format for the machine learning dataset. Any missing data for whatever reason(s) are also managed. Also, oversampling techniques such as SMOTE that involves process of oversampling instances from the minority class thereby creating new synthetic instances of that minority class were used to manage data imbalance. The *set().union() function* is deployed to balance the training and testing datasets

However, in the *third phase - data scaling & normalization*; it usually very common to have a dataset that it's features magnitudes of the units and range varies. Thus, the developed model tends to lean toward parameters with higher weight. In other to avoid any bias of model leaning toward parameters with higher weight, data parameters are scaled down between the range of 0 & 1 using Zscore normalization techniques. This is to calculate each observation in the dataset for the feature.

D. Training the Detection Model

The training procedure model uses the pre-processed data to create train & test dataset, compile and fit the model. The parameters and hyperparameters settings as itemized in Proposition [3] is used to configure the training module. Each

Algorithm 1: Data Pre-processing

```
Begin
    Get Raw Dataset   /* gather raw data */
    while Get data do
        Step 1: Input the sample dataset
        Step 2: Convert the symbolic attributes features
        Step 3: return new set of data
        Step 4: Separate the instances of dataset into
              classes (y) (such as
          Normal, Naïve Malicious Response Injection (NMRI)
          Complex Malicious Response Injection (CMRI)
          Malicious State Command Injection (MSCI)
          Malicious Parameter Command Injection (MPCI)
          Malicious Function Code Injection (MFCI)
          Denial of Service (DoS), and
          Reconnaissance (Recon))
        /* encode classes text values to indexes */
        Input: Categorical data
        Output: Classes indexes
        Data: Classes text value
        Def encode_text_index(df, name):
            le = preprocessing.LabelEncoder()
            df[name] = le.fit_transform(df[name])
            return le.classes /* classes indexes */

    end
    Step 5: Scale & Normalize data ($x_t$) into values
        /* encode a numeric column as Zscores */
    Input: Training & Testing Datasets
    Output: Mean & Standard deviation (sd)
    Data: train_data & test_data
    Def encode_numeric_zscore($df, name, mean = None, sd = None$):
        if mean is None: then
            mean = df[name].mean()
        else
            if sd is None: then
                sd = df[name].std()
            end
        end
        df[name] = (df[name] - mean) / sd
        return df, mean, sd
    Step 6: Split dataset into training and testing data
```

```
/* Data Pre-processing continued */
    /* balance & reshape training & testing dataset */
    while Step 7: Balance & Reshape Training & Testing sets do
        Step 7a: Balance & reshape the training & testing data features
    end
    Input: Training & Testing Datasets
    Output: Balanced & Reshaped Datasets
    Data: train_data & test_data
    Def balance_df(train_data, test_data):
        train_data_columns = list(train_data)
        test_data_columns = list(test_data)
        column_union = list(set().union(train_data, test_data))
    for i in column_union: do
        if i not in list(train_data): then
            train_data[i] = 0
            reshape train_data[i]        /* reshape train_data */
        else
            if i not in list(test_data): then
                test_data[i] = 0
                reshape test_data[i]     /* reshape test_data */
            end
        end
    end
    return balanced & reshaped train_data, test_data
        Step 7b: return transformed new balanced & reshaped training & testing datasets
        /* pickle train_data */
        Step 7c: Pickle transformed training data into a byte stream and store it in a file/ database (.pki)
    for i in range(n): do
        Pickle train_data into a byte [$x$]
        while i ¡ n: do
            repeat
                Pickle train_data into a byte [$x$]
            until there is no sample left in the training set
        end
    end
    return (classes instances indexes)
    return (processed test_ data)
    return (processed pickled train_ data)
end
```

of these parameters in the dataset represents a sensor time series data which will be used one at a pre-determined time window w of each iteration until no data is remaining.

The RNN variants and decision tree are used to train the model. The prediction accuracy of each trained model is calculated and the best model with highest prediction accuracy is saved to be used for model overall detection performance.

E. Applying the Ensemble Module for Detection

This module uses the probability function (created specific for this module) for multi-classification to classify individual

- ☐ class/attack stages
- ☐ predicts
- ☐ calculate accuracy
- ☐ calculate overall performance accuracy

It uses the newly correlated data attributes (best saved model) which has been trained and saved as the best predicted accuracy. The probability function is invoked, which then loads the predicted output best model of each individual algorithm to determined the probability that the best saved model and then apply the ensemble function to generate the overall performance accuracy and the f1-score of the model.

This sub-section demonstrate the overall procedure of model outcome prediction phase. It is important to note that this phase is design to predict trends of individual attack steps and calculate the overall classification of the predicted accuracy. When all the steps are individually predicted, the overall correlated predicted output is feed into the classification phase to classify the attack steps for individual model performance.

At this point, the network security team can use the achieved result to determine the probability of the these individual predicted step to complete the APT lifecycle, and then apply the all the necessary required procedure to stop and mitigate the attack before completion, thereby terminate their actions towards achieving the attackers final goal.

V. Evaluation Results and Discussion

The evaluation of APT_{DASAC} and the achieved results are discussed, highlighting the evaluation metrics used.

A. Evaluation Metrics

Accuracy is usually used as a conventional way of classification performance measure. As mentioned by [37]. However, using accuracy as a metric measure is not appropriate when dealing with multi-class imbalance data since the minority class may have little or no contribution when compared to majority classes toward achieved accuracy.

Proposition 4: Evaluation Metrics Used

Precision, recall, f1-score, overall accuracy, area under the curve (AUC) receiver operating characteristic (ROC) and confusion matrix are utilized to validate the approach of using RNN variants for APT detection system *"APT_{DASAC}"* proposed to get a clearer understanding of the output. All the metrics calculation are based on true positive (TP), true negative (TN), false positive (FP), false negative (FN), and *Loss*.

B. Experimental Evaluation of APT_{DASAC}

The evaluation experiments were performed to test the developed APT_{DASAC} framework in terms of effectiveness and detection capabilities in three different domains.

C. Application Domains

Three different case studies were implemented.

- ***Case Study One: Application to The NGP Dataset:*** APT_{DASAC} was applied to a simulated data containing attacks and normal network behaviour from a laboratory scale industrial control system. This data contains network transaction captured over serial line that contains network information, payload information and label. In [35], the focus was on command injection (CI) attack (alters the system behaviour through injection of false control and configuration commands into a control system) and response injection (RI) attacks (attack that modifies the response from server to client, thereby providing false information about system state). A multi-stage approach based on DL techniques was implemented, where this approach is validated with two case scenarios; a network transactions between a RTU & MTU in-house SCADA gas pipeline control system and a case study of CI and RI attacks detection. The implemented approach achieved competitive attack detection capability with 0% FAR and TPR of 96.50%. This was investigated further in [6], where stacked ensemble-LSTM variants for APT_{DASAC} framework were applied to optimize attack detection rate and achieved overall average mean detection accuracy of 85%.
- ***Case Study Two: Application to KDDCup'99 Dataset:*** This data was created with the intent for researchers to test viable IDS effectiveness. Although, this dataset has played a vital part in research community for evaluating computer network IDSs and as a benchmark dataset for other researchers to compare and validate results, this data was found to contain unintended patterns which has led to algorithms to easily learn differences among patterns, making the detection rate very high. All the data features were used as input vector, trained on full set, and then evaluated with the full test set. Study in [2], applied an approach based on DL models which includes RNN variants algorithms and ML models on KDDCup'99 dataset. The achieved results indicates that DL based model seems to be more efficient when compared with result derived from ML based models as implemented on this study. However, the study did not consider the imbalanced distribution of class elements, which may affect detection/classification rate. During the training, it was observed that RNN variants seems to be more suitable when classifying high-frequency attacks and also the low frequency attacks with very low detection capability, achieving 62.50%, 56.20% and 37.50% for LSTM, GRU and RNN respectively on multi-attack classification, while achieving a very high average accuracy of 99.99% for RNN variants on differentiating attacks from normal instances.
- ***Case Study Three: Application to UNSW-NB15 Dataset:*** The UNSW-NB15 dataset is uesed to establish the classification and detection capability of the proposed framework, *"APT_{DASAC}* framework" in identifying a single attack, able to correctly differentiate normal & attack connection records, and categorize these attacks to their attack family. Previous study [22], explored the applications of heterogeneous ensemble approach and SMOTE data resampling technique with focus on capturing the rare attack and achieved a maximum average mean accuracy of 81.02% with a significant validation loss for UNSW-NB15 dataset. Applying data resampling is to provide a balanced data distribution. Attackers utilizes multiple attack tactics to deliver an attack on their target, this has leads to generation of uneven distribution of attack payload information among examples of different attacks. Learning from imbalance data distribution in multi-attack detection and multi classification problem, poses a significant challenges for ML algorithms, especially in detecting the rare attack. Studies based on ensemble supervised learning are still an active research field in ML community as demonstrated by the authors [38].

D. Results and Discussions

Each attack type within each dataset was split up into train and test at 67% and 33%, ensuring an even distribution of each class labels for the three cases, then utilizing the hyperparameter settings as outlined in Section B, each model is trained over a total of 300 epochs. To validate this approach for detecting APT step attacks, the chosen statistical metrics are calculated to (i) evaluate the ability of this approach to accurately detect and identify an abnormal network as an attack, (ii) to detect different type of attacks accurately (iii) get a clearer understanding of the output, and (iv) deduce the actual overall performance of the model. Each algorithm returned a competitive average detection rate as recorded in Table [I], with insignificant *FPR* and validation loss. It was observed that most of the implemented algorithm appear to be suitable for classifying high-frequency attacks with less detection capability for the low-frequency attacks as it returned low detection rate. It was also noted that, each of these algorithm's result is slightly different in each case.

From the first case study, the *average weighted precision, recall* and *f1-score* are 88%, 86%, & 82% respectively for APT_{DASAC} and 77%, 66% & 70% for ML-DT algorithm. While obtained 88%, 86% & 81% on NGP_6 dataset with overall probability average prediction accuracy of **86.30%** & **86.36%** with loss as **0.32%**. ML-DT achieved *weighted average precision, recall* and *f1-score* of 95% on the second experiment. Since our main concern is to capture all the individual attack steps, it can be seen that out of the total predicted actual values of each of the individual attack, this study model were able to correctly predict an overall average weighted recall of 86% for which 88% were actually predicted as attacks. However, a closer observation of the individual precision and recall of each of the algorithms indicates that the model did struggle to correctly identify RI attack, even though it achieved 100% precision when considering four ICS attack types as contained within the dataset.

TABLE I
OVERALL SUMMARY RESULT OF ALL ALGORITHMS ON ON NGP_5, NGP_6, UNSW-NB15 AND KDDCUP'99 DATASETS. WITH TPR, FPR, F1-SCORE, AND MICRO/MACRO-AVE_ROC CURVE RECORDED FOR DIFFERENT PROBLEM DOMAINS NETWORKS CROSS EVALUATION. THE METRICS PARAMETER WITH BEST RESULT FOR EACH INDIVIDUAL ALGORITHM ARE WRITTEN IN BOLD TEXT.

Summary Result of all Aglorithms on the three Evaluated Datasets					
Datasets	**Algorithm**	**TPR (%)**	**FPR (%)**	**f1-score (%)**	**micro / macro-ave_roc curve**
NGP_5	RNN variants	96.12	**0.31**	**82.07**	**0.91 /0.72**
	APTDASAC	**96.28**	0.62	82.01	**0.91 /0.72**
	ML-DT	46.26	11.01	69.52	0.79 / 0.77
NGP_6	RNN variants	**97.87**	0.32	81.97	0.92 / 0.76
	APTDASAC	97.41	**0.00**	81.45	0.92 / 0.76
	ML-DT	97.61	1.42	**94.6**	**0.97 / 0.94**
UNSW-NB15	RNN variants	86.99	**1.75**	**79.8**	**0.89 / 0.80**
	APTDASAC	**88.18**	2.15	79.57	**0.89 / 0.80**
	ML-DT	31.84	7.05	75.02	0/86 / 0.78
KDDCup'99	RNN variants	99.98	0.08	**99.93**	0.93 / 1.00
	APTDASAC	99.98	**0.05**	**99.93**	**0.96 / 1.00**
	ML-DT	**99.99**	**0.05**	99.92	0.95 / 1.00

The implemented model were able to achieve a significant *precision, recall, f1-score* and *overall average accuracy* of 100% with *TPR* of 99.98% and *FPR* of 0.05%. Also, achieved a macro/micro average roc of 96%/1.00 and an *overall average prediction accuracy* of 99.92%, which indicates a good model performance result. However, this data contains unintended patterns which may have contributed to algorithms to learn differences among patterns so easily, making the results and the detection rate very high.

The authors in [22], acknowledged the need to investigate application of data re-sampling techniques to manage imbalance data distribution, to ascertain the impact of applying SMOTE oversampling techniques with focus on detecting the minority class "Worms" among the multi-class attacks samples. Applying SMOTE techniques did slightly improve the overall average detection rate of Worms attack from 0.03%, 0.06% & 0.09% to 32.50%, 35.50% & 23.2% for the individual algorithms implemented.

While the *ROC* curve of minority class of interest "Worms" also improves from 50% to 59%, with micro & macro-average *ROC* curve of 73% and 65% respectively, and maximum average mean accuracy of 81.02%. However, the achieved results are still below average, this demonstrate that uneven data distribution as discussed by Arthur's in [37] and [39] is not the only factor that could impact model performance since high classification error discussed in [39] also contributed to the poor model performance as presented in [22].

Nevertheless, in the third case study, stacked ensemble-RNN variants for APT_{DASAC} approach were implemented on UNSW-NB15 dataset without any re-sampling techniques applied to establish the capability of this model to detect each individual attack within an enterprise environment. The classification report achieved the average weighted precision, recall and f1-score are 83%, 82%, & 80% respectively for APT_{DASAC} and 77%, 76% & 75% for ML-DT algorithm, with overall probability average prediction accuracy of **82.19%** and loss of **42%**.

Also implemented the same approach based on ML-DT algorithm on UNSW-NB15 data, and achieved 75.02% weighted average accuracy rate. The overall average detection accuracy rate of 82.19% were achieved, which is slightly lower than 86.36% & 99.92% achieved with NGP and KDDCup'99 dataset respectively. The average curves of the classes are evaluated and consolidated into a single graph representing their respective *AUC* curve and obtain *micro-average ROC curve area* of **89%** and *macro-average ROC curve area* of **80%** for the APT_{DASAC}, and obtain *micro-average ROC curve area/macro-average ROC curve area* of 86%/78% respectively from implementing ML-DT approach. The classification of APT attack detection in class 4 stage achieved ROC curve area of **99%** from both models, this is largely attributed to the number of connection record exhibited in this stage, while the class 1 stage has the lowest *ROC* curve area of 52% for APT_{DASAC} model, and 53% for ML-DT model.

VI. CONCLUSION

The study developed a novel system named APT_{DASAC} to detect and predict an APT attack steps, and then investigate the ability of the proposed multi-stage ensemble DL-based model that utilizes ensemble techniques for optimizing detection accuracy by combining network results. The study, demonstrated that using stacked ensemble model, configured with appropriate parameter settings, as well as the correct use of data pre-processing will make a good choice towards developing a system capable of dealing with this type of attack. Thus, ensemble techniques with the probability combiner are used. The final decision is made based on the outputs of all the implemented approaches, where the strength of each individual approach compliment the weakness of the other approach.

The first case study, achieved an *average weighted precision, recall* and *f1-score* of 88%, 86%, & 82% respectively for APT_{DASAC} and 77%, 66% & 70% for ML-DT algorithm from the first experiment. While obtained 88%, 86% & 81% on NGP_6 dataset with overall probability average prediction accuracy of **86.30%** & **86.36%** with loss as **0.32%**, for both experiments. ML-DT achieved *weighted average precision, recall* and *f1-score* of 95% on the second experiment. While in the second case study, the *precision, recall, f1-score* and *overall average accuracy* of 100% with *TPR* of 99.98% and *FPR* of 0.05% were obtained with an *overall average prediction accuracy* of 99.92%. In the third case study, the average weighted precision, recall and f1-score are 83%, 82%, & 80% respectively for APT_{DASAC} and 77%, 76% & 75% for ML-DT algorithm, with overall probability average prediction accuracy of **82.19%** and loss of **42%**.

Considering different results obtained from these application domains, this approach showed a significant attack detection capability and has demonstrated that performance of attack detection approach applied in any domain, can be influences by the nature of network transactions with respect to the domain of application. This suggest that the ability and resilience of operational system state to withstand attack and maintain system functionalities are regulated by the safety and security measures in place, which is specific to that system, suggesting that a hybrid of a model based on combination of ”DL” & ”ML” approaches could make a good model for APT attack steps detection. This is to take advantage of the combined effort of both model, to achieved a more effective detection capability. Hence, this paper main contributions are in the domain of cyber security.

REFERENCES

[1] Yuan, X. (2017, May). Phd forum: deep learning-based real-time malware detection with multi-stage analysis. In 2017 IEEE International Conference on Smart Computing (SMARTCOMP) (pp. 1-2). IEEE.

[2] Eke, H. N., Petrovski, A., & Ahriz, H. (2019, September). The use of machine learning algorithms for detecting advanced persistent threats. In Proceedings of the 12th International Conference on Security of Information and Networks (pp. 1-8).

[3] Giura, P., & Wang, W. (2012, December). A context-based detection framework for advanced persistent threats. In 2012 International Conference on Cyber Security (pp. 69-74). IEEE

[4] Siddiqi, M. A., & Ghani, N. (2016). Critical analysis on advanced persistent threats. International Journal of Computer Applications, 141(13), 46-50.

[5] Five, C. (2011). Advanced persistent threats: A decade in review. Command Five PTY LTD, 1-13.

[6] Eke, H. N., Petrovski, A., Ahriz, H., & Al-Kadri, M. O. (2022). Framework for Detecting APTs Based on Steps Analysis and Correlation. In Security and Resilience in Cyber-Physical Systems (pp. 119-147). Springer, Cham.

[7] Brand, M., Valli, C., & Woodward, A. (2010). Malware forensics: Discovery of the intent of deception. Journal of Digital Forensics, Security and Law, 5(4), 2.

[8] Shashidhar, N., & Chen, L. (2011). A phishing model and its applications to evaluating phishing attacks.

[9] Zimba, A., Chen, H., Wang, Z., & Chishimba, M. (2020). Modeling and detection of the multi-stages of Advanced Persistent Threats attacks based on semi-supervised learning and complex networks characteristics. Future Generation Computer Systems, 106, 501-517.

[10] McClure, S., Gupta, S., Dooley, C., Zaytsev, V., Chen, X. B., Kaspersky, K., ... & Permeh, R. (2010). Protecting your critical assets-lessons learned from operation aurora. Tech. Rep.

[11] Alperovitch, D. (2011). Revealed: operation shady RAT (Vol. 3, p. 2011). McAfee.

[12] Villeneuve, N., Bennett, J. T., Moran, N., Haq, T., Scott, M., & Geers, K. (2013). Operation” Ke3chang: Targeted Attacks Against Ministries of Foreign Affairs. FireEye, Incorporated. villeneuve2013operation

[13] Singh, S., Sharma, P. K., Moon, S. Y., Moon, D., & Park, J. H. (2019). A comprehensive study on APT attacks and countermeasures for future networks and communications: challenges and solutions. The Journal of Supercomputing, 75(8), 4543-4574.

[14] Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2011). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare & Security Research, 1(1), 80.

[15] Kriaa, S., Bouissou, M., & Piètre-Cambacédès, L. (2012, October). Modeling the Stuxnet attack with BDMP: Towards more formal risk assessments. In 2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS) (pp. 1-8). IEEE.

[16] Alperovitch, D. (2011). Revealed: operation shady RAT (Vol. 3, p. 2011). McAfee.

[17] Noor, U., Anwar, Z., Amjad, T., & Choo, K. K. R. (2019). A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise. Future Generation Computer Systems, 96, 227-242.

[18] MITRE ATT&CK (2021). https://attack.mitre.org/groups/G0009/

[19] Joloudari, J. H., Haderbadi, M., Mashmool, A., GhasemiGol, M., Band, S. S., & Mosavi, A. (2020). Early detection of the advanced persistent threat attack using performance analysis of deep learning. IEEE Access, 8, 186125-186137.

[20] Kaspersky, (2021). The Epic Turla (snake/Uroburos) attacks. ”https://www.kaspersky.co.uk/resource-center/threats/epic-turla-snake-malware-attacks”

[21] Kaufman, E. K., Adeoye, S., & Batarseh, F. A. (2023). Leadership for CyberBioSecurity: The Case of Oldsmar Water.

[22] Eke, H., Petrovski, A., & Ahriz, H. (2020). Handling minority class problem in threats detection based on heterogeneous ensemble learning approach. *International Journal of Systems and Software Security and Protection (IJSSSP)*, 11(2), 13-37.

[23] Assante, M. J., & Lee, R. M. (2015). The industrial control system cyber kill chain. SANS Institute InfoSec Reading Room, 1.

[24] Kedgley, M. (2015). If you can’t stop the breach, at least spot the breach. Network Security, 2015(4), 11-12.

[25] KP, S. (2019). RNNSecureNet: Recurrent neural networks for Cyber security use-cases. arXiv preprint arXiv:1901.04281

[26] Zhou, Y., Xu, K., He, F., Zhang, Z. (2022). Application of time series data anomaly detection based on deep learning in continuous casting process. ISIJ International, 62(4), 689-698.

[27] Li, X., Jiang, Y., Liu, Y., Zhang, J., Yin, S., Luo, H. (2022). RAGCN: Region aggregation graph convolutional network for bone age assessment from X-ray images. IEEE Transactions on Instrumentation and Measurement, 71, 1-12.

[28] Marchetti, M., Pierazzi, F., Colajanni, M., & Guido, A. (2016). Analysis of high volumes of network traffic for advanced persistent threat detection. Computer Networks, 109, 127-141.

[29] Moon, D., Im, H., Kim, I., & Park, J. H. (2017). DTB-IDS: an intrusion detection system based on decision tree using behavior analysis for preventing APT attacks. The Journal of supercomputing, 73(7), 2881-2895.

[30] Coulter, R., Zhang, J., Pan, L., & Xiang, Y. (2022). Domain adaptation for Windows advanced persistent threat detection. Computers & Security, 112, 102496.

[31] Cho, D. X., & Nam, H. H. (2019). A method of monitoring and detecting APT attacks based on unknown domains. Procedia Computer Science, 150, 316-323.

[32] Vukalović, J., & Delija, D. (2015, May). Advanced persistent threats-detection and defense. In 2015 38Th international convention on information and communication technology, electronics and microelectronics (MIPRO) (pp. 1324-1330). IEEE.

[33] Eke, H.N., (2016). Study of Web-based Communities: Study of the structure of online communities and application of statistical methods to evaluate users types & behaviours, 1–168 LAP LAMBERT Academic

[34] Berrada, G., Cheney, J., Benabderrahmane, S., Maxwell, W., Mookherjee, H., Theriault, A., & Wright, R. (2020). A baseline for unsupervised advanced persistent threat detection in system-level provenance. Future Generation Computer Systems, 108, 401-413.

[35] Eke, H., Petrovski, A., & Ahriz, H. (2020). Detection of false command and response injection attacks for cyber physical systems security and resilience.. In13th International Conference on Security of Information and Networks (SIN 2020), November 4–7, 2020, Merkez, Turkey.ACM, NewYork, NY, USA, 8 pages. https://dl.acm.org/doi/10.1145/3433174.3433615

[36] Morris, T., & Gao, W. (2014, March). Industrial control system traffic data sets for intrusion detection research. In International Conference on Critical Infrastructure Protection (pp. 65-78). Springer, Berlin, Heidelberg.

[37] Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of imbalanced data: A review. International journal of pattern recognition and artificial intelligence, 23(04), 687-719.

[38] Nguyen, T. T., Dang, M. T., Liew, A. W., & Bezdek, J. C. (2019). A weighted multiple classifier framework based on random projection. Information Sciences, 490, 36-58.

[39] Sáez, J. A., Krawczyk, B., & Woźniak, M. (2016). Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. Pattern Recognition, 57, 164-178.