Received July 6, 2020, accepted July 23, 2020, date of publication July 29, 2020, date of current version August 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3012907

MVFCC: A Multi-View Fuzzy Consensus Clustering Model for Malware Threat Attribution

HAMED HADDADPAJOUH[1], (Member, IEEE), AMIN AZMOODEH[1], (Member, IEEE), ALI DEHGHANTANHA[1], (Senior Member, IEEE), AND REZA M. PARIZI[2], (Senior Member, IEEE)

[1]Cyber Science Lab, University of Guelph, Guelph, ON N1G 2W1, Canada
[2]College of Computing and Software Engineering, Kennesaw State University, Marietta, GA 30060, USA

Corresponding author: Reza M. Parizi (rparizi1@kennesaw.edu)

ABSTRACT The rise of emerging cyberthreats has led to a shift of focus on identifying the source of threat instead of the type of attack to provide a more effective defense to compromised environments against malicious acts. The most complex type of cyberthreat is the Advanced Persistent Threat (APT) attack that is usually backed by one or more states and lunched using a range of clandestine techniques aiming at high-value targets. Finding the source of the attackers and the associated campaign behind the threats can lead to taking an optimum defense decision in a more timely fashion. Threat attribution is an act of attributing an attack to the source of the attack. Threat attribution can not be fully achieved by a single piece of evidence (i.e. single view) from malicious actors as the evidence could get obfuscated by the actor to evade the detection mechanism. In this article, we propose a multi-view fuzzy consensus clustering model for attributing cyber threat payloads (malware) to its actor. We conduct over 4000 experiments to find out the best combinations of all 12 extracted views for the attribution task. Our experiments use five well-know APT families payloads. To avoid bias in the results, we apply a fuzzy pattern tree and multi-modal fuzzy classifier for our inference engines of all views. To define an optimum distinction among the malicious actor's behavior we implemented the consensus clustering technique. The comparison analysis of a single-view versus multi-view result justifies a significant improvement in the accuracy rate of attribution for all actors. The obtained results from the multi-view aspect of our proposed model give 95.2% accuracy.

INDEX TERMS Advanced persistent threat, consensus clustering, cybersecurity, fuzzy, machine learning, malware, multi-view learning, threat attribution.

I. INTRODUCTION

Nowadays, cyberthreats are becoming more complex in their tactics, techniques, and procedures (TTP). Most attack campaigns can be attributed to their TTP while analyzing and profiling a certain threat actor [1]. A TTP can be a specific signature or a backdoor dropped on the victim machine or a specific pattern in a network traffic flow used by a malicious actor.

Most large-scale malware threats follow similar procedures that exist in highly risk threats named Advanced Persistent Threat (APT) attacks [2]. The best action against a malware threat is to find out how it works. Since most malware threats like APT actors use different evasion hacking techniques, it would be a difficult task to identify their campaign behind that threat [3]. Cyberthreat attribution, or the identification of the actor responsible for a cyberattack, consists of many procedures, and detecting malware threat family/campaign by their nature of behaviors is an effective action that leads to making a suitable decision upon their nature.

Machine Learning (ML) is a state-of-the-art approach that has been used and helped cyberthreat attribution process from manual to a semi-automated or even fully automated manner. The majority of previous efforts, include having multiple ML threat hunting module [1] instead of a single comprehensive model. On the other hand, deep learning approaches have shown to have a significant impact on detecting malware [4]–[9] in different domains. The most complex challenge in any attack attribution task is building a verifiable and scientific method to relate different pieces of evidence

The associate editor coordinating the review of this manuscript and approving it for publication was An-An Liu.

and building a complete picture that depicts possible sources of an attack. This is an even more complicated task in the cyber domain. Digital traces and cyber evidence are much more fragile, and actors are well educated in this domain. We have identified the following as challenges for cyberthreat attribution:

- Overlaps in different threat actors' tactics, techniques, and procedures (TTP): There is a significant degree of similarity between different actors TTP. Overlaps are not only in (Command and Control) C2 IP addresses or domain names but also in malicious payloads and even attack strategies. As threat actors are actively monitoring each other's campaign, they are very quick in adopting other groups successful techniques which further increase similarities between different groups campaigns which are running at the same time.
- Utilizing standard administrative tools instead of customized hacking tools: While early APT actors tend to use customized tools for exploitation or privilege escalation, threat actors quickly realize that there is an abundance of network administration tools that can be used for the same purpose while these tools are not raising any red flags. Using standard network administration PowerShell[1] scripts or task scheduling scripts are now very common techniques for persistence on a target network. Utilizing such general admin tools further complicates the task of threat attribution as it reduces the number of unique remnants that can be used to identify sources of an attack.
- Inserting false flags and fake clues: Many threat actors are intentionally dropping payloads used by other groups or inserting other groups' code in their payload. This technique is usually used to mislead the investigator and disrupt the investigation or incident response process.
- Emergence of multi-stage attacks: With ever-increasing layers of defense that are deployed in different organizations, different threat actors are becoming specialized in specific stages of a compromise. For example, there are threat actors specialized in conducting reconnaissance against targets and others who are specialized in developing exploit kits. Hence, many recent campaigns contain remnants pointing to different threat actors. This leaves security analysts with pieces of evidence pointing to different hacking groups which further complicates the task of attributing an attack to any specific actor.

Despite these challenges, attribution remains an important aspect of any attack investigation. Identifying (possible) sources of an attack would assist incident handlers to quickly devise the best course of action to contain and eradicate a threat, helps forensics examiners to identify possible sources of evidence, and aid threat analysts to detect gaps within an organization cyber defense posture. Advancements in cyber defense and attack detection technologies open new opportunities to conduct efficient threat attribution. More specifically, there are two classes of defense techniques for threat attribution that have been of high importance in the research community.

[1]https://github.com/PowerShell/PowerShell

A. MULTI-VIEW ML AGENT FOR THREAT ATTRIBUTION

The majority of fake flags or evasion techniques are focused on blinding a specific view of an analyst. For example, threat actors may pack their malware to evade opcode-based detection or malware may call fake libraries to evade dynamic malware analysis techniques. However, it is very difficult to evade all different views of a campaign, i.e. packing code, by inserting fake libraries, changing file headers and calling misleading IP addresses all at the same time. Therefore, bypassing AI-based systems that can analyze multiple different views of a system is an extremely difficult task. This makes multi-view AI systems an optimal choice for threat attribution activities. These systems evaluate sources of an attack based on remnants collected from different views and provide a good estimate of all possible sources of a campaign. Moreover, multi-view systems are very effective in identifying sources of a multi-stage attack. These systems may provide a good estimate of the origin of the tools used at different stages as well as an overall estimate of possible sources of an attack.

ML agents are able to established clusters based on similarity metrics of the entity such as static properties of malware. Therefore, for attributing a threat based on different sources of data an ensemble of classifiers needs to work in tandem. Consensus clustering is an approach which includes all data sources of an object for its decision making process [10]. In recent years, applying different sources of data which belong to a single object for training ML agent by consensus clustering is increased [11]–[14].

B. UTILIZING FUZZY-LEARNING SYSTEMS FOR THREAT ATTRIBUTION

Pattern-based machine learning agents such as deep learners or classifiers are good in identifying weaponized payloads with similar patterns to those samples used during training tasks. However, most of the pattern-based detection techniques are failed in identifying permuted malware or in the detection of weaponized payloads generated using polymorphic or metamorphic techniques which are commonly used by APT actors. Fuzzy machine learning techniques are very suitable to tackle this issue. These techniques are generating fuzzy (loosely defined) patterns of malicious payloads that offer a higher degree of similarity to polymorphed or meta-morphed malware. Trained engines using a fuzzy representation of malicious payloads may estimate similarities between previously seen malware generated by an actor and the given sample that can be used to identify possible sources of an attack.

In this article, we propose a multi-view fuzzy consensus clustering model for attributing APT malware groups based on their different associated artifacts. The outline of the contributions of this article relative to the prior methods in the field can be summarized as:

- Developing a customized Sandbox for extracting APT malware views as a data channel.
- Building a multi-view fuzzy consensus clustering model to attribute APT malware to their associated campaigns.
- Providing a comprehensive comparison between single view attribution and multi-view malware attribution.

The proposed method not only benefits from involving different sources of information for attributing a malware threat to its malicious threat actor but also can make a distinction by the fuzzy rules on existing overlaps among different types of malicious campaigns, which in turn can help tackle the threat attribution problem more effectively. To the best of our knowledge, this work creates one of the first instances of a multi-view approach for attributing APT malware to malicious campaigns.

The paper is organized as follows. In Section II, we review the related work in cyberthreat hunting and threat attributions. In Section III, we present our MVFCC model for cyberthreat attribution. In section IV, we analyze the obtained results from MVFCC and present an analytical comparison from the obtained results of multi-view and single-view attribution. Finally, In Section V, we give the concluding remarks and future works related to cyberthreat attribution.

II. RELATED WORK

Due to increasing the degree of complexity in malware threats, finding the source of the attack can be led to take an optimum decision after a potential threat transforms into a serious attack. Therefore, cyberthreat attribution using machine learning (ML) has been attracted by more researchers than before for finding an automated solution against critical damage caused by malicious actors. Most ML-based cyberthreat attribution models usually consist of one or more likewise threat hunting engines. Thus, in this section, we briefly review the recent works on cyberthreat hunting and attribution.

Thonnard *et al.* [15] proposed a multi-criteria culturing approach to address attack attribution in cloud-based platforms. The proposed model used an unsupervised learning approach that analyzed clusters of IP address to find the real source of the attack. In their approach, each cluster consists of several graph nodes that represent an IP address associated with another graph node. Then, it creates a profile of attack and normal behaviors with a set of connected graph nodes named behavior clusters. They defined each cluster member by some unique features as (the geolocation of IP sources, distribution of sources IP addresses, targeted platforms, targeted port sequences, and the ratio of common IP addresses). After each cluster formed, they extracted cliques of attackers to analyze clusters for attributing a similar attack to the source of them. Their work applied to two years worth of attack traced by 40 honey pots appointed all around the world to show the robustness of their model.

Thonnard *et al.* [16] also proposed another model based on knowledge discovery and the fuzzy decision-making approach to attribute an attack to its sources. Their model generates fuzzy clusters in different attack dimensions. Each attack has four dimensions named geolocation IP subnet (means which IP class is sample associated with), targeted platform, and port sequences. After these clusters are formed, they generated their appropriated fuzzy rules based on membership functions of each sample to each malicious campaign. They used the Sugeno inference model for fuzzy rule generation. After the model rule generation is completed, they set multi-criteria for fine-tuning their model to a desirable result. For tuning the fuzzy engine they used Order Weight Aggregation (OWA) technique [17]. In OWA all membership functions which attributed to a malicious actor are aggregated to deliver a final output with a high rate of confidence.

Consensus clustering is a suitable approach against uncertain object clustering problems. In recent years a wide range of efforts has been proposed to provide effective clustering using different sources of data. Huang *et al.* [10] proposed a novel ensemble approach for sparse graph representation and probability analysis. In their proposed model some microclusters were established to speed up the consensus clustering process. In another work [12], the authors proposed an ensemble clustering model based on uncertainty estimation by considering cluster labels in the entire clustering process. Moreover, they proposed a model [14] which follows two clustering algorithms based in K-means and K-nearest representation for addressing the scalability of consensus clustering challenge.

In order to use a different source of a malicious object data source, Appice *et al.* [18] proposed a clustering-aided multiview classification approached for Android malware detection. The construct of their consensus clusters is based on different Android malware properties including User Permissions, API Call, and Intent. To build their multi-view clusters they used the K-Means algorithm based on a dissimilarity measure and obtained 96.7% of detection accuracy in their experimental study.

Finding the source of a severe attack like Distributed Denial of Service (DDoS) always being a difficult task of cyberthreat attribution. Saied *et al.* [19] proposed a model based on a deep neural network engine to detect DDoS attack sources. They build their model based on three detector instances called ICMP, UDP, TCP source code. These indicators led to model precepts from the multi-source of input rather one. Therefore the proposed model can make its decision based on different sources and this model will be able to find the similarity between attack types. Knowing the similarity of attacks, the attribution of each attack to its actor can be feasible. The authors obtained over 98% detection accuracy and desirable rate of attribution associated to attack detection.

Kang and Kang [20] proposed an Intrusion Detection System for autonomous vehicles based on convolution neural networks to find out the source of the attacks on vehicles.

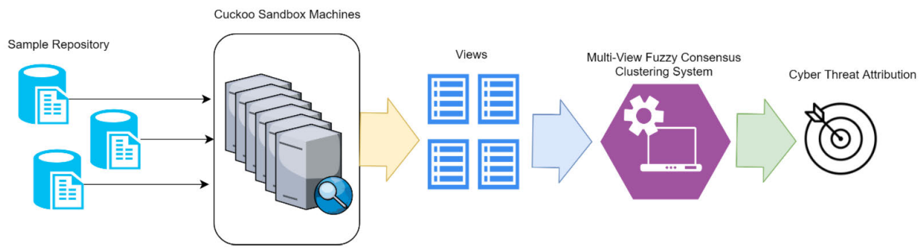


FIGURE 1. The proposed MVFCC architecture design for APT malware threat attribution.

Their proposed model trained by Central Area Network packets features. The main purpose of their proposed model was finding the property targeted by the attacker and then find the best solution against the attack. Their model equipped from a novel feature selection method for learning from unseen attack samples. The proposed model performed by the probability of the attack on the learned classes of a malicious actor. Although this model can attribute an attack to the class of target property, the procedure was still in a semi-auto manner.

In the fintech domain, Noor *et al.* [21] used Indicator of Compromise (IoC) as input, the feed which produces by cyberthreat intelligence teams, and Natural Language Processing Based on deep neural network engine to attribute a FinTech attack to its actor. They obtained over 98% accuracy to attribute an attack which had IoCs to its actor. They used a huge data set that contains attacks IoC from 2012 to 2018. One of the advantages of their model was the number of data samples which led to a suitable learning rate.

One of the most common resources of cyberthreat attribution is malware analysis using machine learning techniques. Malware analysis is divided into two main categories named Dynamic and Static Analysis. In static analysis finding the abnormal pattern in static features of malfunction software like operation code (Opcode), byte code or header is the matter. On the other hand in dynamic analysis finding an abnormal pattern when malfunction software is running will be matter. Thus for finding an attack source or attribute an attack to its actor both dynamic and static approaches are useful materials.

In recent years, a wide range of researchers in the realm of cybersecurity used malware analysis for threat hunting and attribution. Haddadpajouh *et al.* [22] proposed a deep recurrent neural network to detect malware threats from a static view. They used Long Short Term Memory (LSTM) structure for the learning engine of their model and also used a sequence of the OpCode as the input to the proposed model. Their proposed model was shown to be able to detect threats that weren't seen by the engine during the training phase. Although the proposed model obtained the desire rate in threat detection, it could not find unknown attacks' actors.

III. THE PROPOSED METHOD (MVFCC)

To propose a multi-view system for cyberthreat attribution, it is necessary to demonstrate the multi-view aspect of the malicious executable from the recognized APT group. Figure 1 illustrates a holistic view of the proposed system from generating the views of APT malware samples by running them through a customized Sandbox to attribute each sample to corresponded malicious campaign.

In this section, firstly, the collected dataset is described, and then the pre-processing operations to prepare it for the learning phase are presented. Next, the process of converting the sample to the fuzzy domain using fuzzy classification and clustering methods are proposed. Figure 2 gives further details about how the MVFCC generates an integrated fuzzy dataset from different views for attributing the APT malware samples to an appropriate APT actor.

A. PRE-PROCESSING AND VIEW GENERATION

In order to collect adequate data from malicious samples, the Sandbox's output was processed and different views were generated in four categories: *Opcode*, *Bytecode*, *SystemCall* and *Header*.

1) OPCODE

Each executable sample is a sequence of Operational Codes (OpCode) belongs to the running platform microprocessor instructions. To generate different views from OpCode sequence, the dictionary D_{OpCode} that represents the unique set of all available OpCodes within samples was obtained, which include:

- **Binary**: Each sample is transmuted to a vector based on its OpCode sequence($Sample_{OpCodeSequence}$) to generate this view. Length of each sample is equal to length of D_{OpCode} and the view is generated using Equation 1.

$$Sample_{View=BinaryOpCode} = \{x_i = 1 \; if \; D_{OpCode}[i] \in Sample_{OpCodeSequence}\} \quad (1)$$

- **Count**: Each sample is transmuted to a vector based on $Sample_{OpCodeSequence}$ to generate this view. Length of each sample is equal to length of D_{OpCode} and the view is generated using Equation 2.

$$Sample_{View=CountOpCode} = \{x_i = Count \; of \; D_{OpCode}[i] \; in \; Sample_{OpCodeSequence}\} \quad (2)$$

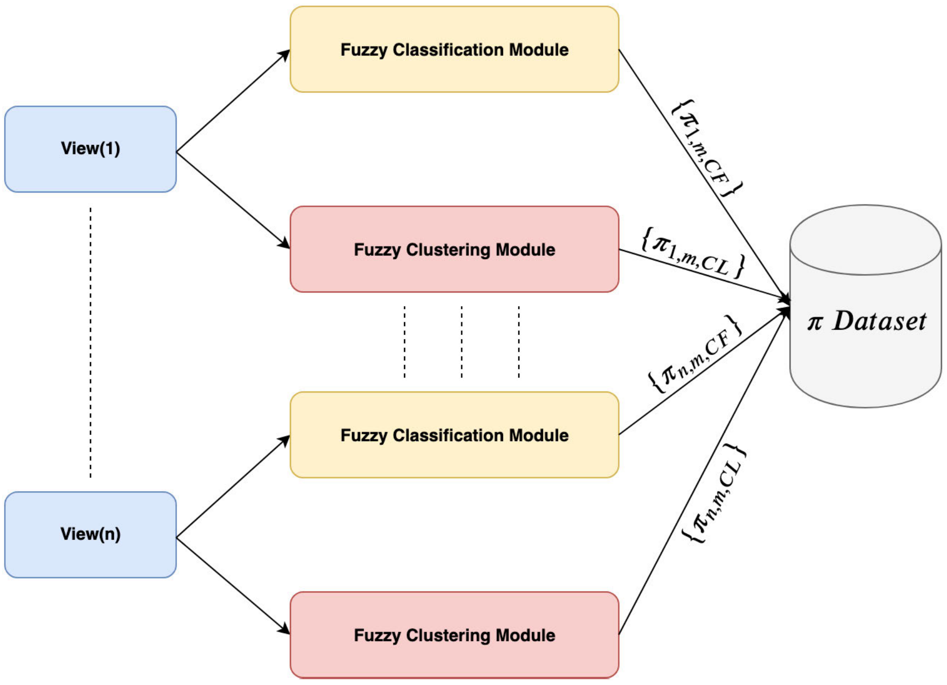


FIGURE 2. The proposed MVFCC ML model.

- **Frequency of Occurrence**: Each sample is transmuted to a vector$Sample_{OpCodeSequence}$ to generate this view. Length of each sample is equal to length of D_{OpCode} and the view is generated using Equation 3.

$$Sample_{View=FrequencyOpCode} = \{x_i = \frac{Count\ of\ D_{OpCode}[i]\ in\ Sample_{OpCodeSequence}}{|Sample_{OpCodeSequence}|}\} \quad (3)$$

- **Term Frequency-Inverse Document Frequency (TF-IDF)**: Each sample is transmuted to a vector $Sample_{OpCodeSequence}$ to generate this view. Length of each sample is equal to length of D_{OpCode} and the view is generated using Equation 4.

$$tf(OpCode_x, Sample) = \frac{Number\ of\ times\ OpCode_x\ appears\ in\ Sample}{|Sample|}$$
$$idf(OpCode_x, Samples) = \log[\frac{|\{Samples\}|}{|\{Samples\ include\ OpCode_x\}|}]$$
$$Sample_{View=tf-idfOpCode} = tf(OpCode_i, Sample) * idf(OpCode_i, Samples)\} \quad (4)$$

- **EigenVector**: Using Hashemi *et al.* [23] method, graph of executable's control flow is generated (OpCodes are the graph's nodes and an edge between $OpCode_x$ and $OpCode_y$ represents the number of $<OpCode_x, OpCode_y>$ occurrence in the $Sample_{OpCodeSequence}$). Then the graph is embedded into a vector $Sample_{View=EigenOpCode}$ that is a combination of all graph's eigenvectors. This vector is a low-dimensional representation of sample's control flow and $|Sample_{View=EigenOpCode}| = |D_{OpCode}|$.

2) BYTECODE

Sequence of sample's ByteCodes are processed and views are generated similar to OpCode sequence using Equation 1, 2, 3, 4 and [23]'s method. As for ByteCode views, $D_{ByteCode} = 0, 1, \ldots, 255$ and length *Binary,Count, Frequency, Tf-idf and EigenVector* is 255.

3) SYSTEMCALL

Using Cuckoo Sandbox, systemcall's information for samples is extracted. Then, the dictionary $D_{Systemcall}$ that represents the unique set of systemcalls is generated and based on Equation 3, view $Sample_{View=SystemCallFrequency}$ is calculated for each sample.

4) HEADER

In order to include the header's information in our proposed method, header information is extracted using the Sandbox module. Since there was a large variance between different properties of the header, we used Equation 5 so as to normalize the header view.

$$Sample_{View=Header} = \{x_i = \log(1 + |SampleHeaderValue_i|)\} \quad (5)$$

Algorithm 1 gives the pseudocode of view generation procedures. This algorithm takes raw views as two different

Algorithm 1 Generating Multi-View From Raw View

```
Input: $rawView$: Raw view generated by customized
        Sandbox
Output: $V_i$ *[v]: List of parsed view with cetrain
        encoding
Function viewGenerator($rawView$):
    $sample \longleftarrow rawView$;
    $modes = \{binary, count, eigen, frequency, tf - idf\}$
    foreach $mode \in modes$ do
        if $sequential(rawView) == True$ then
            $V \longleftarrow parse(sample, mode)$;
        else
            $V \longleftarrow normalize(sample)$;
        end
    end
    return $V*$;
End Function
```

approaches named sequential or non-sequential. If a view consists of sequences of words like Opcode or numbers like Bytecode, it needs to be parsed and be discretized as a feature vector. Otherwise, if the raw view is not sequential (like header files), then it needs to be normalized for avoiding any biases.

B. FUZZIFICATION FOR FUZZY CLASSIFICATION

All features of all single-views should be fuzzified before training and classification tasks. For this purpose, we used Trapezoidal and Triangular fuzzifier [24] that generate fuzzy membership functions for features of every single view. Figure 3 and Figure 4 depict Triangular and Trapezoidal fuzzy membership function, and $\mu_1(x)$ and $\mu_2(x)$ definitions are shown in Equation 6 and Equation 7 respectively. Where a refers to lower limit, b refers to upper limit of value and m is the actual value in triangular function in 6. Also, a refers to lower limit, d refers to upper limit, a lower support b and upper support limit c and m is the actual value in trapezoidal function in 7

$$\mu_1(x) = \begin{cases} 0, & \text{if } x \le a. \\ \dfrac{b-x}{m-b}, & \text{if } m \ge x > a. \\ \dfrac{b-x}{m-b}, & \text{if } b \ge x > m. \\ 0, & \text{if } x \ge b. \end{cases} \tag{6}$$

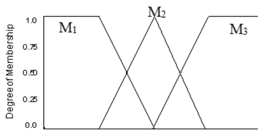


FIGURE 3. Triangular fuzzy membership function.

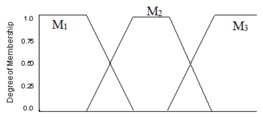


FIGURE 4. Trapezoidal fuzzy membership function.

$$\mu_2(x) = \begin{cases} 0, & \text{if } x < a \text{ or } x > d. \\ \dfrac{x-a}{b-a}, & \text{if } b \ge x \ge a. \\ 1, & \text{if } b \ge x \ge c. \\ 0, & \text{if } x \ge b. \end{cases} \tag{7}$$

C. FUZZY CLASSIFICATION

Fuzzy Pattern Tree (FPT) is a recently introduced fuzzy classification algorithm [25]. FPT starts with creating fuzzy partitions on each view using Equation 6 and Equation7. Then, it generates several basic FPTs on these partitions $F_{i,j}$ where $j \in \{\mu_1, \mu_2\}$ and $i \in \{single\ view's\ attributes\}$. Afterward, it iteratively expands basic FPTs by calculating *RMSE (Root Mean Square Error)* of the pattern trees and selects and expands the optimum tree based on the lowest RMSE measure. The algorithm considers an independent tree for each class label (here APT group). Algorithm 2 describes the FPT approach. The algorithm is performed once for each APT group. We defined each fuzzy set as a list variable and named it set in the Algorithm. For example, Set P and M include the initial fuzzy set of basic partitions and the best fuzzy set that resulted from the fuzzy pattern tree after a certain number of iterations respectively. Each FPT classifier includes leaves that accept the view's features and internal nodes that apply fuzzy operators. Finally, each FPT outputs a confidence value π_{target} for its corresponding APT group.

D. FUZZY CLUSTERING

In order to create fuzzy basic clusters, we employed Fuzzy C-Means partitioning [26] technique to divide malware in each APT group to several clusters that explain most similarity among cluster's members [27]. In this technique, the label of clusters are assigned using majority voting. Finally, $\{CM_{view,target,i}\}$ is available where $i \in \{1, \ldots, C\}$ and C is maximum number of clusters on each view,and in our experiment $C = 200$ was set. Each $CM_{view,target,i}$ calculates a fuzzy metric $\pi_{view,target}$.

E. π AGGREGATION

As a result of fuzzy classification and clustering (as presented in Sections III-C and III-D), a dataset of fuzzy metrics $\{\pi_{alg,view,target}\}$ were generated which *alg* refers to *Fuzzy Classification* and *Fuzzy Clustering* algorithms and each $\pi_{alg,view,target} \in [0, 1]$. At this step, a Decision Tree [28] is trained by π dataset which accepts $\{\pi_{alg,view,target}, TrueTarget\}$ as training data and learn how to make a decision based on calculated fuzzy metrics. This component plays the role of the final decision maker that

Algorithm 2 Fuzzy Pattern Tree Pseudocode

```
Input: $target$ and $View_x$: Generated View by
        Algorithm 1
Output: $\{PT_{view,target}\}$ : A set of FPT classifiers
Function FPT($View_x$, $target$):
    Generate Basic Partitions $F_{i,j}$;
    Set $P = \{F_{i,j}\}$;
    Set $M^* = \{\}$;
    $/ * M^*$ includes best fuzzy pattern tree $* /$
    Set Stopping Criteria $\lambda = 0.005$;
    Set $MaximumDepth = 10$;
    while $error > \lambda$ do
        foreach all $L \in leafs(M^*)$ do
            Set $temp = \{\}$
            if $Depth(L) <= MaximumDepth$ then
                $PT_{temp}$ = Expand $L$ using fuzzy
                operators and $PT$;
                Append $PT_{temp}$ to $temp$
            end
        end
        Set $M^* = temp_{x^*}$ where $RMSE(temp_{x^*}) <=$
        $RMSE(temp_x)$;
    end
    return $M^*$;
End Function
Applying FPT on each view, a set of $k$ classifier will be
trained where $k$ is number of APT groups. Finally,
$\{FPT_{view,target}\}$ is generated where the number of FPT
classifiers is $|\{FPT_{view,target}\}| = |Views| * |APT\ Groups|$
and each $FPT_{view,target} \in \{FPT_{view,target}\}$ calculates
fuzzy metric $\pi_{view,target}$.
```

aggregates information from different fuzzy learning algorithms that have been trained by different views.

IV. EXPERIMENTAL RESULTS

We evaluated the MVFCC based on different evaluation metrics to examine the attribution accuracy. Before that we defined the applied dataset and it's features.

A. DATASET

To conduct our experiment for malware threat attribution, we used a dataset that consists of 1200 APT malware samples[2] that belong to five different APT groups namely APT1, APT3, APT28, APT33, and APT37. The collected dataset includes some other attack campaigns which classified as these major groups. Afterward, we ran the samples in our customized Cuckoo Sandbox[3] to collect multiple static and dynamic views of each sample. We utilized Cuckoo version 2.0.61 as the base Sandbox to generate dynamic malware views. Since Cuckoo did not originally provide our proposed method's raw views, namely *Header, Opcode, Bytecode and Systemcall*, we had to write the required customized scripts. Figure 5 shows the process of generating different views (Opcode, Bytecode, System Call and Header) from each APT malware sample in the MVFCC Sandbox component. As it mentioned the collected dataset consists of five major attack campaigns namely APT1, APT3, APT28, APT33, and APT37. All other campaign names like Winniti are subcategories of these major campaigns.

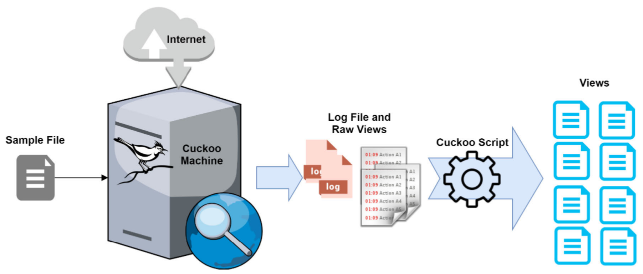


FIGURE 5. **Multi-view sample extraction from malicious file through customizing Cuckoo Sandbox.**

B. PERFORMANCE METRICS

There are four core metrics to evaluate the performance of machine learning algorithms named as True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). Since there are no normal classes in the collected APT dataset we needed to define the threat attribution performance metrics as a multi binary classification problem. Therefore, in each evaluation, we picked an APT group as a negative class like a normal class, and the rest of the other APT classes considered a positive malicious class. Based on this, evaluation metrics in our study are defined as follows:

- **True Positive (TP):** The positive APT samples where the true label is positive and whose class is correctly predicted to be positive.
- **True Negative (TN):** The negative APT samples where the true label is negative and whose class is correctly predicted to be negative.
- **False Positive (FP):** The negative APT samples where the true label is negative and whose class is correctly predicted to be positive.
- **False Negative (FN):** The positive APT where the true label is positive and whose class is incorrectly predicted to be negative.

Using the above core metrics, we can measure the performance of machine learning systems using the following metrics:

Precision: Precision for a certain APT group is the number of samples in a class that is correctly predicted, divided by the total number of samples that are predicted.

$$Precision = \frac{TP}{(TP + FP)} \tag{8}$$

[2]https://github.com/cyber-research/APTMalware
[3]https://cuckooSandbox.org

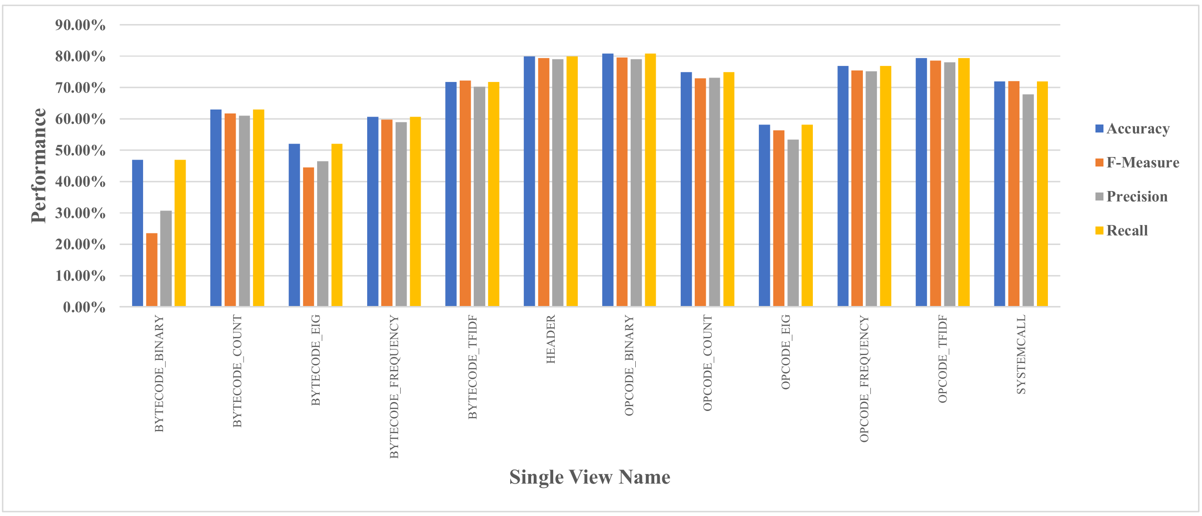
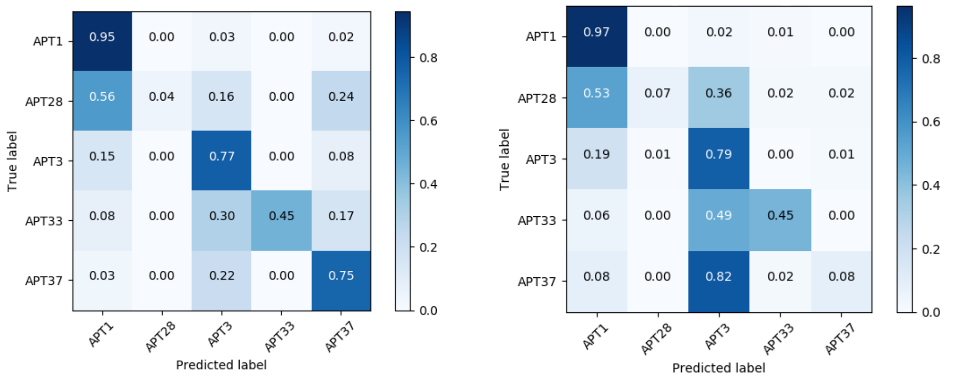
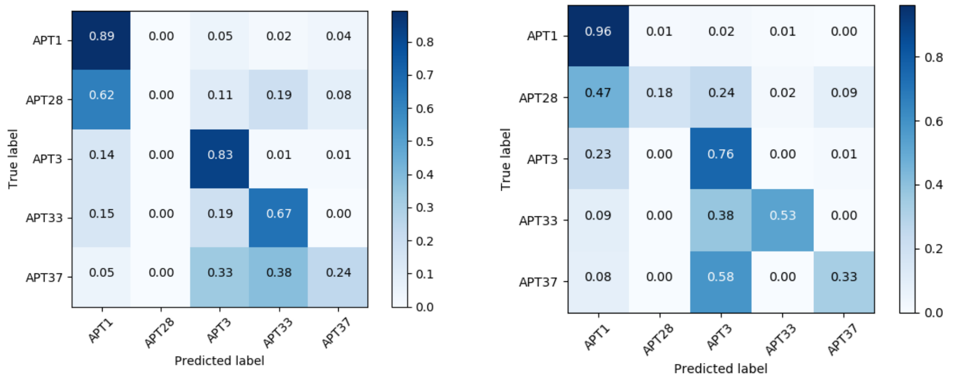


FIGURE 6. The obtained evaluation metrics from each single view by Fuzzy Pattern Tree approach.



(a) OpCodeBinary single-view confusion matrix for all five APT groups.

(b) OpcodeFrequency single-view confusion matrix for all five APT groups.



(c) Header single-view confusion matrix for all five APT groups

(d) OpCodeTFIDF single-view confusion matrix for all five APT groups

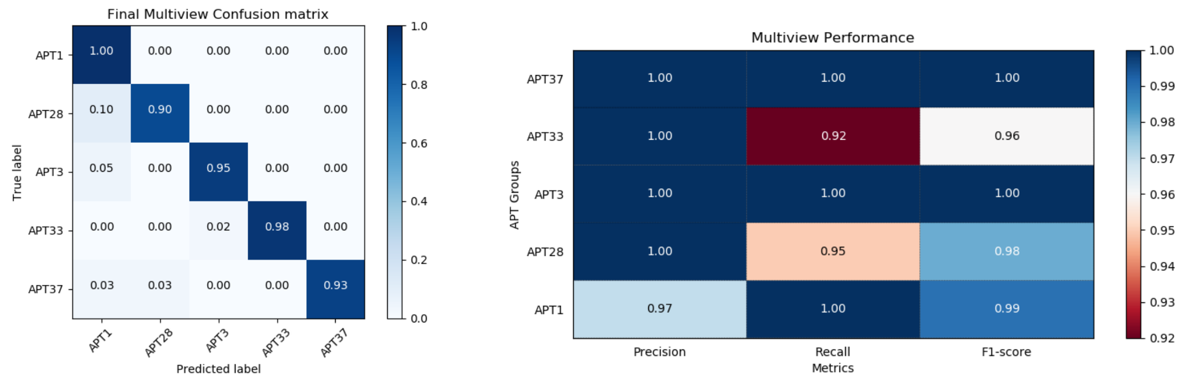
FIGURE 7. Confusion matrix for best single-view fuzzy pattern trees.

Recall: for a certain class, is the number of samples in a class that are correctly predicted, divided by total number of samples in that class.

$$Recall = \frac{TP}{(TP + FN)} \tag{9}$$

F-Score (F1): F-Score is the harmonic mean of Precision and Recall. It can be applied as a general classifier performance metric.:

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{10}$$



(a) Multi-view confusion matrix for all five APT groups. (b) Overall performance metrics by multi-view approach against all five APT groups.

FIGURE 8. The proposed model performance metrics and accuracy on multi-view approach.

Confusion Matrix is a specific table layout that allows visualization of the performance of a machine learning algorithm [29]. This matrix is only a better representation of previously defined metrics. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class. Diagonal cells represent the accuracy of the trained model for each APT group. In an ideal case, all diagonal cells should be 1 and non-diagonal cells are expected to be zero. More than 4000 experiments were conducted to analyze all possible combinations of all different views to train our machine learning agent and then test it for a threat attribution task. To demonstrate the impact of multi-view against a single-view approach for APT malware threat attribution, we trained and tested using all possible combinations of the extracted views. Since some APT groups samples in the collected dataset had not the proper view, we faced biased and imbalanced data regarding the specific groups. Therefore, we omitted that single view for the attribution process based on a specific view. Moreover, we treated the missing view as missing value and generate a synthetic view from the same cluster samples before the decision-making phase happens. All in all, the missing view issue will not affect the final result of the proposed approach. The proposed model first generates 12 views from OpCode, ByteCode, Header, and SystemCall of malicious payloads belonging to each and every APT actor. Afterward, it merges similar views of different APT actors to one single-view that represents a specific property among all APT actors. Then, the model trains two fuzzy models using each of 12 generated single-views which will result in 24 trained fuzzy models. We analyzed all performance metrics (precision, recall, F1 score, and confusion matrix) for all 24 trained models. These trained models are generating fuzzy metrics corresponding to every malware that will be used for consensus clustering.

Figure 6 shows all the evaluation metrics of Fuzzy Pattern Tree obtained from each single view approach. Figure 7 illustrates the confusion matrix for top 4 best single-view models that obtained higher performance based on Figure.6 outcomes. The results showed that the attribution outcome is not desirable by just considering a single view to find the APT actors. In contrast, when we combined all other views, the model results had a significant improvement and distinctions among APT actors. Figure 8 shows both confusion matrix and other performance criteria for MVFCC system. The obtained results show over 95% of attribution accuracy for APT malware threats.

All in all, the obtained results justify that the MVFCC is able to attribute malware sample to its the APT actor correctly with 95% of accuracy with sufficient properties from malicious payloads. Also, this result implies that if a malicious actor attempts to evade or fool the MVFCC, it will need to change all possible properties which is impossible due to executable file functionalities [30]. To best of the authors' knowledge, there was no similar approach for attributing APT malware based on a multi-view approach to compare the MVFCC with. Therefore, we hope the MVFCC and collected multi-view dataset could pave the way for further research on malware threat attribution in this particular research direction.

V. CONCLUSION AND FUTURE WORKS

Cyberthreat attribution is one of the complex tasks in the cybersecurity domain. Although machine learning may have a significant impact to automate some parts of this process, there are not many works in this domain due to serious challenges like lack of sufficient information about threat actors and the complex mechanism of attacks. Since most critical cyberthreats, e.g., Advanced Persistent Threats, are usually backed by one or more campaign/ state, effectively attributing such types of threats to the responsible campaign can lead to reducing the decision-making process, and in turn better defense solutions, after the occurrence of these threats.

In this article, we proposed an automated multi-view consensus fuzzy clustering model for attributing malicious payloads to their associated APT actor. For evaluating the proposed model, we applied five APT families along with 12 different extracted views for attribution. We justified the effectiveness of multi-view versus single-view by more than 4000 experiments over a combination of different views. The obtained results show over 95% accuracy in attributing the malicious payloads to their actors. To expand the current work, we are developing a stack of different machine learning models besides the current model for cyberthreat attribution.

REFERENCES

[1] H. Lin, "Attribution of malicious cyber incidents: From soup to nuts," *J. Int. Affairs*, vol. 70, no. 1, pp. 75–137, 2016.

[2] E. W. Burger, M. D. Goodman, P. Kampanakis, and K. A. Zhu, "Taxonomy model for cyber threat intelligence information exchange technologies," in *Proc. ACM Workshop Inf. Sharing Collaborative Secur. (WISCS)*, New York, NY, USA, 2014, pp. 51–60. [Online]. Available: http://doi.acm.org/10.1145/2663876.2663883

[3] S. Doherty, J. Gegeny, B. Spasojevic, and J. Baltazar. *Hidden Lynx—Professional Hackers for Hire*. pp. 1–28. Accessed: Oct. 12, 2019. [Online]. Available: http://www.symantec.com/content/en/us/enterprise/media/security_

[4] A. N. Jahromi, S. Hashemi, A. Dehghantanha, K.-K.-R. Choo, H. Karimipour, D. E. Newton, and R. M. Parizi, "An improved two-hidden-layer extreme learning machine for malware hunting," *Comput. Secur.*, vol. 89, Feb. 2020, Art. no. 101655.

[5] H. Karimipour, A. Dehghantanha, R. M. Parizi, K.-K.-R. Choo, and H. Leung, "A deep and scalable unsupervised machine learning system for cyber-attack detection in large-scale smart grids," *IEEE Access*, vol. 7, pp. 80778–80788, 2019.

[6] E. M. Dovom, A. Azmoodeh, A. Dehghantanha, D. E. Newton, R. M. Parizi, and H. Karimipour, "Fuzzy pattern tree for edge malware detection and categorization in IoT," *J. Syst. Archit.*, vol. 97, pp. 1–7, Aug. 2019.

[7] H. HaddadPajouh, R. Khayami, A. Dehghantanha, K.-K.-R. Choo, and R. M. Parizi, "AI4SAFE-IoT: An AI-powered secure architecture for edge layer of Internet of Things," *Neural Comput. Appl.*, pp. 1–15, Feb. 2020, doi: 10.1007/s00521-020-04772-3.

[8] M. Nassiri, H. HaddadPajouh, A. Dehghantanha, H. Karimipour, R. M. Parizi, and G. Srivastava, "Malware elimination impact on dynamic analysis: An experimental machine learning approach," in *Handbook of Big Data Privacy*. Cham, Switzerland: Springer, 2020, pp. 359–370.

[9] H. Darabian, A. Dehghantanha, S. Hashemi, M. Taheri, A. Azmoodeh, S. Homayoun, K.-K.-R. Choo, and R. M. Parizi, "A multiview learning method for malware threat hunting: Windows, IoT and Android as case studies," *World Wide Web*, vol. 23, no. 2, pp. 1241–1260, Mar. 2020.

[10] D. Huang, J.-H. Lai, and C.-D. Wang, "Robust ensemble clustering using probability trajectories," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1312–1326, May 2016.

[11] M. Pividori, G. Stegmayer, and D. H. Milone, "Diversity control for improving the analysis of consensus clustering," *Inf. Sci.*, vols. 361–362, pp. 120–134, Sep. 2016.

[12] D. Huang, C.-D. Wang, and J.-H. Lai, "Locally weighted ensemble clustering," *IEEE Trans. Cybern.*, vol. 48, no. 5, pp. 1460–1473, May 2018.

[13] D. Huang, C.-D. Wang, H. Peng, J. Lai, and C.-K. Kwoh, "Enhanced ensemble clustering via fast propagation of cluster-wise similarities," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Nov. 6, 2018, doi: 10.1109/TSMC.2018.2876202.

[14] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai, and C.-K. Kwoh, "Ultra-scalable spectral clustering and ensemble clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 6, pp. 1212–1226, Jun. 2020.

[15] O. Thonnard, W. Mees, and M. Dacier, "Addressing the attack attribution problem using knowledge discovery and multi-criteria fuzzy decision-making," in *Proc. ACM SIGKDD Workshop CyberSecur. Intell. Inform.*, 2009, pp. 11–21.

[16] O. Thonnard, W. Mees, and M. Dacier, "On a multicriteria clustering approach for attack attribution," *ACM SIGKDD Explorations Newslett.*, vol. 12, no. 1, pp. 11–20, Nov. 2010.

[17] G. Bonaccorso, *Mastering Machine Learning Algorithms*. Birmingham, U.K.: Packt Publishing, 2018.

[18] A. Appice, G. Andresini, and D. Malerba, "Clustering-aided multi-view classification: A case study on Android malware detection," *J. Intell. Inf. Syst.*, vol. 55, no. 1, pp. 1–26, Aug. 2020.

[19] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using artificial neural networks," *Neurocomputing*, vol. 172, pp. 385–393, Jan. 2016.

[20] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLoS ONE*, vol. 11, no. 6, Jun. 2016, Art. no. e0155781.

[21] U. Noor, Z. Anwar, T. Amjad, and K.-K.-R. Choo, "A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise," *Future Gener. Comput. Syst.*, vol. 96, pp. 227–242, Jul. 2019.

[22] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K.-R. Choo, "A deep recurrent neural network based approach for Internet of Things malware threat hunting," *Future Gener. Comput. Syst.*, vol. 85, pp. 88–96, Aug. 2018.

[23] H. Hashemi, A. Azmoodeh, A. Hamzeh, and S. Hashemi, "Graph embedding as a new approach for unknown malware detection," *J. Comput. Virol. Hacking Techn.*, vol. 13, no. 3, pp. 153–166, Aug. 2017.

[24] S. Abbasbandy and T. Hajjari, "A new approach for ranking of trapezoidal fuzzy numbers," *Comput. Math. Appl.*, vol. 57, no. 3, pp. 413–419, Feb. 2009.

[25] R. Senge and E. Hüllermeier, "Top-down induction of fuzzy pattern trees," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 2, pp. 241–252, Apr. 2011.

[26] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy c-means model," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 3, pp. 370–379, Aug. 1995.

[27] J. Wu, Z. Wu, J. Cao, H. Liu, G. Chen, and Y. Zhang, "Fuzzy consensus clustering with applications on big data," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1430–1445, Dec. 2017.

[28] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 3, pp. 660–674, May/Jun. 1991.

[29] A. M. Hay, "The derivation of global estimates from a confusion matrix," *Int. J. Remote Sens.*, vol. 9, no. 8, pp. 1395–1398, Aug. 1988.

[30] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial examples for malware detection," in *Proc. Eur. Symp. Res. Comput. Secur.* Cham, Switzerland: Springer, 2017, pp. 62–79.

HAMED HADDADPAJOUH (Member, IEEE) received the B.S.S. degree in computer software engineering and the M.Sc. degree in information technology engineering from Shiraz University. He is currently pursuing the Ph.D. degree with the University of Guelph, Canada. He is working as a Senior Researcher at the University of Guelph. His main research interests include machine learning and adversarial machine applications in cybersecurity, especially in the Internet of Things (IoT) devices. He also has several years' experience as the Research and Development Director and being the Co-Founder in different high-tech startup companies.

AMIN AZMOODEH (Member, IEEE) received the B.S.S. degree in computer engineering and the M.Sc. degree in artificial intelligence from Shiraz University. He is currently pursuing the Ph.D. degree with the University of Guelph, Canada. His main research interests include the theory of machine learning and artificial intelligence, and adversarial machine learning. He is also interested in the application of machine learning, especially in cybersecurity and digital forensics. He is a Microsoft Certified Professional and has several years' experience in analyzing and implementing enterprise resource planning software.

ALI DEHGHANTANHA (Senior Member, IEEE) received the Ph.D. degree in security in computing. He has a number of professional certifications, including CISSP and CISM. He is currently the Director of the Cyber Science Lab, University of Guelph, ON, Canada. His lab is focused on building AI-powered solutions to support cyber threat attribution, cyber threat hunting, and digital forensics tasks in the Internet of Things (IoT), the industrial IoT, and the Internet of Military of Things (IoMT) environments. Prior to joining the University of Guelph, he has served as a Senior Lecturer with The University of Sheffield, U.K., and an EU Marie-Curie International Incoming Fellow with the University of Salford, U.K. He has served for more than a decade in a variety of industrial and academic positions with leading players in cyber-security and artificial intelligence.

REZA M. PARIZI (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science, in 2005 and 2008, respectively, and the Ph.D. degree in software engineering, in 2012. He is currently the Director of the Decentralized Science Lab (dSL), Kennesaw State University, GA, USA. He is a Consummate AI Technologist and Software Security Researcher with an entrepreneurial spirit. Prior to joining KSU, he was a Faculty Member with the New York Institute of Technology. His research interests include research and development in decentralized AI, cybersecurity, blockchain systems, smart contracts, and emerging issues in the practice of secure software-run world applications. He is a Senior Member of the IEEE Blockchain Community and ACM.