IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 16, 2019, accepted July 10, 2019, date of publication July 22, 2019, date of current version August 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2930200

Hidden Markov Models and Alert Correlations for the Prediction of Advanced Persistent Threats

IBRAHIM GHAFIR[1], KONSTANTINOS G. KYRIAKOPOULOS[1,2], (Member, IEEE), SANGARAPILLAI LAMBOTHARAN[1], FRANCISCO J. APARICIO-NAVARRO[3], BASIL ASSADHAN[4], HAMAD BINSALLEEH[5], AND DIAB M. DIAB[6]

[1]Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough LE11 3TU, U.K.
[2]Institute for Digital Technologies, Loughborough University London, London E20 3BS, U.K.
[3]Faculty of Computing, Engineering and Media, De Montfort University, Leicester LE1 9BH, U.K.
[4]Department of Electrical Engineering, King Saud University, Riyadh 11421, Saudi Arabia
[5]Department of Computer Science, Imam Muhammad Ibn Saud Islamic University, Riyadh 11432, Saudi Arabia
[6]Department of Computer Science, King Saud University, Riyadh 11421, Saudi Arabia

Corresponding author: Ibrahim Ghafir (i.ghafir@lboro.ac.uk)

This work was supported in part by the Gulf Science, Innovation and Knowledge Economy Programme of the U.K. Government under UK-Gulf Institutional Link Grant IL 279339985 and in part by the Engineering and Physical Sciences Research Council (EPSRC), U.K., under Grant EP/R006385/1.

ABSTRACT Cyber security has become a matter of a global interest, and several attacks target industrial companies and governmental organizations. The advanced persistent threats (APTs) have emerged as a new and complex version of multi-stage attacks (MSAs), targeting selected companies and organizations. Current APT detection systems focus on raising the detection alerts rather than predicting APTs. Forecasting the APT stages not only reveals the APT life cycle in its early stages but also helps to understand the attacker's strategies and aims. This paper proposes a novel intrusion detection system for APT detection and prediction. This system undergoes two main phases; the first one achieves the attack scenario reconstruction. This phase has a correlation framework to link the elementary alerts that belong to the same APT campaign. The correlation is based on matching the attributes of the elementary alerts that are generated over a configurable time window. The second phase of the proposed system is the attack decoding. This phase utilizes the hidden Markov model (HMM) to determine the most likely sequence of APT stages for a given sequence of correlated alerts. Moreover, a prediction algorithm is developed to predict the next step of the APT campaign after computing the probability of each APT stage to be the next step of the attacker. The proposed approach estimates the sequence of APT stages with a prediction accuracy of at least 91.80%. In addition, it predicts the next step of the APT campaign with an accuracy of 66.50%, 92.70%, and 100% based on two, three, and four correlated alerts, respectively.

INDEX TERMS Advanced persistent threat, intrusion detection system, alert correlation, hidden Markov model, attack prediction.

I. INTRODUCTION

Cyber attacks have become more widespread and several attacks have made headline news over the past decade, targeting industrial companies and governmental organizations [1]. These attacks have caused substantial financial losses and were able to hinder operation of core public services. Furthermore, since the Internet of Things (IoT) has emerged, the number of devices connected to the Internet is increasing rapidly and becoming easy targets for cyber attacks [2]. The global cost of cybercrime has now reached $600 billion according to a McAfee report in 2018 [3]. The term cyber attack refers to criminal activities launched via the Internet, aiming usually at financial gain or confidential data exfiltration. Additionally, an attacker can spy and monitor the target organisation and disrupt its functions, due to political, ideological or criminal motivation [4].

To mitigate cyber attacks, cyber security analysts heavily depend on Intrusion Detection Systems (IDSs) which can detect malicious activities by matching patterns of

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

known attacks (i.e. signature-based) or observing anomaly activities (i.e. anomaly-based) [5]. In signature-based IDS, the detection is based on a comparison between the monitored data/information and a signatures-database. This database contains a list of known attack signatures. If a match is found, the monitored data/information is reported as malicious and an alert is raised. In anomaly-based IDS, the detection is based on a comparison between the monitored activity and a baseline profile. This baseline profile is built within the training phase and a threshold is set. Any deviation for the monitored activity from the baseline profile is considered as malicious.

Multi-Stage Attacks (MSAs) are a cyber security threat in which the attack campaign is performed through several stages. In recent years, Advanced Persistent Threats (APTs) have emerged as a sophisticated version of MSAs [6]. This type of cyber threat is conducted by highly skilled and motivated cyber-criminals, aiming at spying and data exfiltration. Individually, each APT stage can appear as a benign one and does not raise any suspicion. Furthermore, the attack could last for weeks or years, while exfiltrating the target's data without being detected. The detection of an APT requires an IDS to correlate several alerts during the APT life cycle to reveal the attack campaigns, i.e. to reconstruct the attack scenario [7]. The traditional pattern matching methods are not applicable for APT detection, as these methods require that all possible actions to be well defined. Unfortunately, this is not the case of an APT for several reasons. For instance, there is not any specific sequence of stages to perform the attack campaign. An attacker does not need to follow a precise order of stages. On the contrary, the attacker needs to adapt the implementation of a successful APT campaign to the actual characteristics and current configuration of the targeted system. The techniques used during this process may vary from pre-existing and well-known penetration tools to bespoke and unique software. Moreover, there are many factors that could halt the APT campaign, and there is not a particular point at which an APT finishes. Once the attacker takes control of a system and information has been exfiltrated, the attacker could abandon the attack, or keep access to the system for months or years after the initial attack. Additionally, the detection of one or more stages of APT can be missed due to technical limitations of network devices or due to the attacker using new techniques to compromise current or new vulnerabilities. These challenges have brought much interest in the research and investment towards developing new tools and approaches for APT detection.

The Hidden Markov Model (HMM) is a statistical model used for representing probability distributions over sequences of observations [8]. This model has been used in several domains such as speech recognition [9], text understanding [10], image identification [11] and microbiology [12]. HMM has also been utilized in [13], [14] to train models using observed network traffic under normal network conditions and to detect diverting sequences of traffic observations. These diversions could indicate network anomalies, either genuine or malicious. The HMM has the potential to detect MSAs even if an IDS misses the detection of certain stages of the attack. Hence, HMM addresses the challenge of providing complete information on the attack campaign.

This work proposes a novel IDS for APT detection and prediction. The proposed system undergoes two main phases, the first one is for attack scenario reconstruction, and the second phase is for attack prediction. The first phase of the proposed system is presented in our previous work [15]. This has been extended using the second phase of the proposed approach by incorporating HMM. The first phase of this approach achieves the attack scenario reconstruction based on alert type and spatiotemporal characteristics. This phase has a correlation framework to link the elementary alerts that belong to the same APT campaign. The correlation is based on matching the attributes of the elementary alerts which are generated over a configurable time window. The second phase of the proposed approach is the generation of probabilities and prediction of various stages of APT. This second phase, called the attack decoding, utilizes the HMM to determine the most likely sequence of APT stages for a given sequence of correlated alerts. Moreover, this phase predicts the next step of the APT campaign based on the current and past observations and the transition probabilities of the HMM model.

The contribution of this work is summarized as follows:

- Relevant HMM has been developed for APT prediction. This module employs the Viterbi algorithm to determine the most likely sequence of APT stages for the sequence of correlated alerts linked by the correlation framework in the first phase of the proposed IDS. Forecasting the APT stages not only reveals the APT life cycle in its early steps but also helps to understand the attacker's strategies and aims. Additionally, predicting the next step of the attacker plays a key role in the attack response and enables the network security team to take the required actions before the attacker reaches the final stage of data exfiltration.
- Potential of the new HMM approach for APT is demonstrated using a carefully designed synthetic data. Due to the lack of relevant publicly available data for APT scenarios, the demonstration of the HMM approaches using the synthetic data, as made available in [16], will be beneficial to stimulate further research interests in the research community.

The remainder of this paper is organized as follows. In Section II, the most relevant previous works are reviewed. Section III defines the APT life cycle and provides an overview of the methods and algorithms used in this work. The proposed system for APT detection and prediction is described in Section IV. Section V presents the performance evaluation of the proposed system and discusses the results. Finally, conclusions are drawn in Section VI.

II. RELATED WORK

The detection of APTs is a challenge for current IDSs, and much research has been conducted to address this type of MSA. In [17], the authors propose SPuNge, a host-based APT detector. The proposed system monitors the traffic of each host in the network and analyses malicious URLs. These malicious connections can be established by the hosts via an Internet browser or by a piece of malware installed on the compromised machine. Then, all the machines that show similar activities are grouped and considered as part of an APT campaign. Although this system can raise an alert on the APT attack, it does not consider any type of alert correlation between the APT stages, i.e. the APT scenario cannot be revealed.

The work in [18] utilizes the algorithm Data Leakage Prevention (DLP) to detect data exfiltration, which is the final stage of the APT campaign. In particular, the proposed methodology uses a DLP algorithm to monitor the network traffic and analyses the transferred data over the connections to detect any data leakage. Then, fingerprints are produced based on the attributes of the leaked information. This methodology employs external cyber counterintelligence (CCI) sensors in order to track the location or path of the leaked data. This system is based on detecting the final stage of APT, data exfiltration, but it does not detect the early stages of APT or provide an early alarm to mitigate the attack campaign before the damage of data exfiltration occurs. Furthermore, the external CCI sensors have access to the data traffic which might create an issue in terms of data privacy.

An approach for APT detection, called TerminAPTor, is presented in [19]. This approach tracks the information flow to correlate alerts generated within the APT campaign. This correlation is based on the similarity of alerts' attributes which can be generated by another IDS like Snort. The system performance was evaluated by simulating two APT scenarios and the authors mentioned that the false positive rate should be improved.

The authors in [20] describe an APT detector based on command and control (C&C) domain detection. After investigating the C&C communication of APTs, the authors proposed a new feature called Independent Access. This methodology assumes that the access to C&C domains is independent while the access to legal domains is correlated. Thus, this methodology analyses the Domain Name System (DNS) records and applies RIPPER classification algorithm to classify the domains into C&C domains and legal ones. One limitation of this detector acknowledged by the authors is that it cannot detect the C&C domains if the C&C communication occurs while the user is surfing the Internet.

An active-learning-based system was introduced in [4]. This system detects malicious PDF files which can be used to install malware and infect the network machines as part of the APT life cycle. The system analyses the network connections and gathers all PDF files. Then, a developed module is used to filter all known benign and malicious files. This module utilizes white lists, reputation systems and antivirus signature repository to achieve its functionality. Next, the remaining unknown files are checked for their compatibility as viable PDF files. According to the compatibility test, these files are added to white or blacklists to achieve the active learning of the system. This system detects an APT only if a malicious PDF file is used within the APT campaign. Meaning, the detection of the APT can easily be evaded when the attacker uses other techniques, rather than malicious PDFs, to infect the network machines.

Spear phishing is a common technique used to get the Point of Entry (PoE) into the targeted system in an APT. The work in [21] presents a methodology to detect this technique. This methodology is based on mathematical and computational analysis to detect spam emails. It looks for specific words and characters, called tokens (such as click here, free, Viagra, replica) to distinguish spam emails. These tokens should be defined to the detection algorithm. Nonetheless, it is not guaranteed that the spam email will include one of these tokens, which is a limitation for this methodology.

Focusing on HMM based techniques, the work in [22] utilizes the HMM to rank the APT scenarios which are reconstructed by other tools of alerts correlation. It considers the APT stages as the HMM states and the correlated alerts as observations. Then, using the HMM parameters (transition, emission and initial probabilities), the proposed approach computes a probability score for each sequence of linked alerts. Following that, the APT scenarios are ranked according to the highest probability. Thus, the APT scenarios with low probabilities are considered having wrongly-correlated alerts. In contrast, in our work, we utilize the HMM to estimate the most likely sequence of stages for a given sequence of correlated alerts. Furthermore, we use the HMM to predict the next step of the APT campaign.

A probabilistic approach to predict a network intrusion was presented in [14]. The proposed approach uses a Markov chain to model network events. This system computes the probability of a network event and makes a decision on the abnormality based on this probability. The system undergoes three main phases: in the first phase, the network states are defined using K-means clustering algorithm. In the second phase, the state transition probability matrix and the initial probability distribution are computed based on Markov chain assumptions. In the final phase, the probability of a malicious event is stochastically calculated. This system is predicting malicious activities in general, and the states of Markov chain model are defined by the k-means algorithm. However, our approach addresses MSAs and the states of HMM are modeled on the APT life cycle.

The work in [23] utilizes an HMM to propose an adaptive risk approach for the prediction of MSAs in cloud systems. This approach measures the potential impact of a malicious activity on assets based on its occurrence probability. First, the system gathers different events from several detectors. Then, these events are structured in the IDMEF protocol [24] for the purpose of event correlation. The events are compared

with a set of attack rules and considered correlated if they were triggered by the same rule. Next, the HMM is used to estimate the current state of the cloud system and measures the threat level. This approach assumes four states for the cloud system: Hale (H): indicates that there is no threat, Investigate (I): indicates that there is an attack attempt, Attack (A): indicates that an attack is in progress, Penetrate (P): indicates that an attack is complete. The observations are events or alerts gathered from different sensors and cause the system to transit from one state to another. These observations are linked with the system states and classified into four threat levels: Low, Medium, High, and Very high. Thus, the HMM can estimate the system status utilizing Viterbi algorithm and measures the risk based on the alert threat level. This system assumes the states of HMM and estimate the threat level based on the current observation. However, our system considers the typical APT stages as states. Furthermore, our system goes beyond estimating the current state by predicting the next step of the attacker.

FIGURE 1. Typical stages of an APT attack [25].

III. PRELIMINARIES

This section provides an overview of APT and the mathematical framework of relevant algorithms used by the proposed prediction system. These are the Markov chain, the Hidden Markov Model, the Viterbi algorithm and the Baum-Welch algorithm.

A. ADVANCED PERSISTENT THREAT LIFE CYCLE

An APT is a cybercrime category directed at espionage and confidential data exfiltration, which requires a high degree of stealthiness over a long period of operation in order to be successful. Figure 1 depicts various stages of an APT attack [25].

1) Intelligence gathering: This initial stage aims to get information regarding the target organisation, such as its structure, IT environment and even about its employees. For this purpose, the attacker can use public sources (social media, webpages, etc) and social engineering. The information gathered during this stage will allow the attacker to craft the spear phishing email which is the most common technique to get the point of entry [25].
2) Initial compromise (Point of Entry): Performed by use of social engineering and spear phishing, or by exploiting software vulnerabilities. Another popular infection method is to plant a piece of malware into a website which the victim employees are likely to visit.
3) Command and control (C&C) communication: After an organization's perimeter has been breached, continuous communication between the infected host and the C&C server should be preserved to instruct and guide the compromised machine. These communications are usually protected by Secure Sockets Layer (SSL), making it difficult to identify if the traffic is malicious. Another technique that can be used in this stage is domain flux technique [26]; an exploited host may try to connect to a large number of domain names which are expected to be C&C servers. The goal of this technique is to make it difficult or even impossible to shut down all of these domain names.
4) Lateral movement: Once the target's network has been accessed, the attacker laterally moves throughout the target's network searching for new hosts to infect. The attacker can use brute force attack to obtain information such as a user password or personal identification number (PIN) [27]. Another technique is pass the hash attack, in which the attacker steals a hashed user credential and, without cracking it, reuses it to trick an authentication system into creating a new authenticated session on the same network [28].
5) Asset discovery: This stage aims to identify and find the noteworthy assets within the target's network for future data exfiltration. Port scanning can be used for this step [29].
6) Data exfiltration: Data of interest is transmitted into external servers which are controlled by the attacker. There are some techniques used for data exfiltration like built-in file transfer, via FTP or HTTP and via the Tor anonymity network [30].

B. HIDDEN MARKOV MODEL

In a Markov chain, the states are visible and the transition probabilities can be obtained. Thus, the future state q_{t+1} is predicted based on the current state q_t. Therefore, the Markov chain is sometimes called the observed Markov model. However, in other cases, some of the states are not directly observed and only observations related to these hidden states are observed. The observation likelihoods are called emission probabilities. For a given set of N states, $S = (s_1, s_2, \ldots, s_N)$, and discreet observation symbols, $\bar{O}_M = \bar{o}_1, \bar{o}_2, \ldots, \bar{o}_M$, the Hidden Markov Model (HMM) is defined by the state

transition matrix, $A = \{a_{i,j}\}$, observation emission matrix, $B = \{b_i(\bar{o}_k)\}$, and initial matrix, π_i, where $i, j \epsilon [1, \ldots N]$ and $k \epsilon [1 \ldots M]$ [31]. $a_{i,j}$ is the probability of moving from state i to state j, $b_i(\bar{o}_k)$ is the probability of an observation, $\bar{o}_k$, emitted at state i, and π_i is the initial probability of HMM to start in state i. Thus, an HMM is fully described by $\lambda = (A, B, \pi)$.

For a sequence of observations, $O_t = (o_1, o_2, \ldots o_t)$, and a sequence of states, $Q_t = (q_1, \ldots, q_t)$, a first-order HMM assumes that the probability of a particular state depends only on the previous state:

$$P(q_t|q_1, q_2, \ldots q_{t-1}) = P(q_t|q_{t-1})$$

Another assumption for a first-order HMM is that the probability of an observation, o_t, does not rely on other observations. It is only based on the state q_t that generates the observation, regardless of other previous states or observations [31]:

$$P(o_t|q_1, \ldots, q_t, o_1, \ldots, o_{t-1}) = P(o_t|q_t)$$

1) THE FORWARD - BACKWARD ALGORITHM

To determine the probability of an observation sequence occurring under an HMM, $P(O_T|\lambda)$, all possible path probabilities would have to be considered, since the actual state sequence is hidden. For an HMM with N hidden states and T observations, there are N^T possible hidden sequences. Therefore, the number of possible paths increases exponentially with the length of the observation sequence O_T.

However, the complexity of the algorithm can be reduced by leveraging on the Markov property and using dynamic programming. A Forward and Backward algorithm is commonly used for this purpose. The Forward (FW) algorithm computes the observation probability by summing over the probabilities of all possible state paths that could generate the observation sequence, as in the following three steps [32]:

The initialization step ($t = 1$),

$$\alpha_1(i) = \pi_i b_i(o_1) \tag{1}$$

The induction step (for $1 < t \leq T$), which has complexity of O(NT):

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{i,j}\right] b_j(o_{t+1}) \tag{2}$$

And the termination step, $t = T$, has complexity O(N),

$$P(O_T|\lambda) = \sum_{i=1}^{N} \alpha_T(i) = \sum_{i=1}^{N} P(o_1, o_2, \ldots, o_T, q_T = s_i) \tag{3}$$

Thus, in total the FW algorithm has complexity in $O(N^2T)$.

Figure 2 shows an example of a forward trellis which has $N = 2$ states and $t = 3$ observation time slots. The output of the Forward algorithm is the $N \times t$ array α. Each $\alpha_t(j)$ value represents the probability of being in state j after seeing the first t observations, given the HMM model.

Assume, as an example, the HMM matrices to be:

$$A = \begin{matrix} & \begin{matrix} s_1 & s_2 \end{matrix} \\ \begin{matrix} s_1 \\ s_2 \end{matrix} & \begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix} \end{matrix} \quad (t \rightarrow t+1), \quad B = \begin{matrix} & \begin{matrix} \bar{o}_1 & \bar{o}_2 & \bar{o}_3 \end{matrix} \\ \begin{matrix} s_1 \\ s_2 \end{matrix} & \begin{bmatrix} 0.1 & 0.4 & 0.5 \\ 0.6 & 0.3 & 0.1 \end{bmatrix} \end{matrix}, \quad \pi = \begin{matrix} \begin{matrix} s_1 & s_2 \end{matrix} \\ \begin{bmatrix} 0.7 & 0.3 \end{bmatrix} \end{matrix}$$

Suppose the observation sequence is $\bar{O}_3 = (\bar{o}_2, \bar{o}_1, \bar{o}_3)$, the α parameters calculated in the first two stages of Figure 2 are shown below.

$$\begin{aligned}
\alpha_1(1) &= \pi_1 * b_1(\bar{o}_2) = 0.7 * 0.4 = 0.28 \\
\alpha_1(2) &= \pi_2 * b_2(\bar{o}_2) = 0.3 * 0.3 = 0.09 \\
\alpha_2(1) &= (\alpha_1(1) * a_{1,1} + \alpha_1(2) * a_{2,1}) * b_1(\bar{o}_1) \\
&= (0.28 * 0.6 + 0.09 * 0.3) * 0.1 = 0.0195 \\
\alpha_2(2) &= (\alpha_1(1) * a_{1,2} + \alpha_1(2) * a_{2,2}) * b_2(\bar{o}_1) \\
&= (0.28 * 0.4 + 0.09 * 0.7) * 0.6 = 0.105
\end{aligned}$$

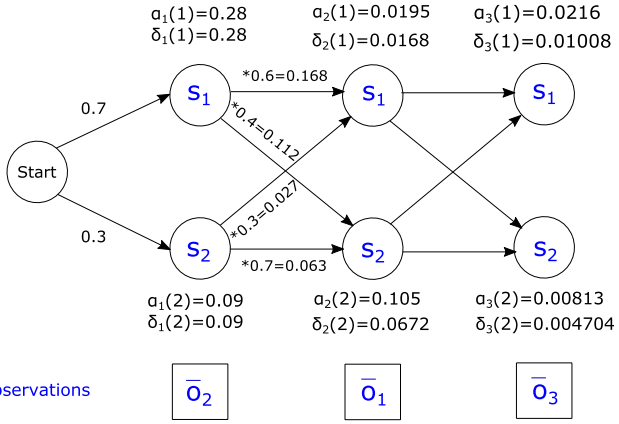

FIGURE 2. The FW and Viterbi algorithms for an HMM of two states and having three observations. For every time instance, the forward probabilities for the FW and Viterbi algorithms, i.e. the α and δ parameters, are indicated [33]. Note, observations might occur randomly and out of sequence.

The backwards part of the Forward-Backward algorithm computes β parameter, as follows [32]:

$$\beta_T(i) = 1, \quad \forall i = 1 \ldots N \tag{4}$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j), \tag{5}$$

$\forall i = 1 \ldots N$ and $t = T-1, T-2, \ldots, 1$

2) THE VITERBI ALGORITHM

Let us suppose that we have a sequence of observations, O_T, and we want to determine the most probable sequence of states, Q_T, given the HMM. This task is known as decoding. One approach to find the sequence of states is to calculate the probability of the observation sequence for each possible path based on the Forward algorithm. Then, the most likely sequence of states can be determined by tracing back the path with highest likelihood value starting from the most likely state at the end of observation.

The Viterbi algorithm introduces the δ parameter in a similar manner to α of FW algorithm, but instead of summing the probabilities from all prior states, only the maximum likelihood value is considered. In addition, the ψ parameter keeps track of the prior state that maximizes this likelihood towards all states and at each time instance. There are three steps for the Viterbi algorithm, as well:

The initialization step ($t = 1$),

$$\delta_1(i) = \pi_i b_i(o_1) \quad \text{same as for } \alpha \text{ in eq.(1)}$$
$$\psi_1(i) = 0$$

The recursion step, for $1 < t \leq T$ and for $1 \leq i \leq N$

$$\delta_t(j) = \max_{1 \leq i \leq N} \{\delta_{t-1}(i) a_{i,j}\} b_j(o_t) \tag{6}$$
$$\psi_t(j) = \arg\max_{1 \leq i \leq N} \{\delta_{t-1}(i) a_{i,j}\} \tag{7}$$

The termination step ($t = T$):

$$\text{Max. probability: } P(T) = \max_{1 \leq i \leq N} \delta_T(i) \tag{8}$$
$$\text{Best last state: } q_T = \arg\max_{1 \leq i \leq N} \delta_T(i) \tag{9}$$
$$\text{Previous best states: } q_t = \psi_{t+1}(q_{t+1}) \tag{10}$$

Figure 2 also shows the δ values. At $t = 1$ the calculation of δ is the same as for α. Then, we have:

$$\begin{aligned}\delta_2(1) &= max(\delta_1(1) * a_{1,1}, \delta_1(2) * a_{2,1}) * b_1(\bar{o}_1) \\ &= max(0.28 * 0.6, 0.09 * 0.3) * 0.1 \\ &= max(0.168, 0.027) * 0.1 = 0.168 * 0.1 = 0.0168 \\ \delta_3(1) &= max(\delta_2(1) * a_{1,1}, \delta_2(2) * a_{2,1}) * b_1(\bar{o}_3) \\ &= max(0.0168 * 0.6, 0.0672 * 0.3) * 0.5 \\ &= max(0.01008, 0.02016) * 0.5 = 0.01008\end{aligned}$$

3) THE BAUM-WELCH ALGORITHM

The Baum-Welch (BW) is a learning algorithm used to optimize the HMM transition and emission probabilities given a set of sequences of observations. It works by computing an initial estimate for these probabilities and then iteratively refining these estimates [31].

The algorithm starts with setting random initial parameters for the HMM. It initially uses the forward-backward parameters α and β to achieve its functionality, but then also introduces, using Bayes' theorem and expectation maximization [32], the following two parameters:

$\xi_t(i, j)$ is the probability of being in state i at time t, transitioning to state j at time $t + 1$, given the observed sequence:

$$\begin{aligned}\xi_t(i,j) &= P(q_t = s_i, q_{t+1} = s_j | O_T, \lambda) \\ &= \frac{\alpha_t(i) a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j)}\end{aligned} \tag{11}$$

$\gamma_t(i)$ is the ξ probability marginalized over j:

$$\begin{aligned}\gamma_t(i) &= P(q_t = s_i | O_T, \lambda) = \sum_{j=1}^{N} \xi_t(i,j) \\ &= \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)}\end{aligned} \tag{12}$$

Next, the HMM parameters λ are updated, considering one observation sequence from the set, by computing π_i^*, $a_{i,j}^*$ and $b_i^*(\bar{o}_k)$, with π_i^* being the expected frequency spent in state i at time 1.

$$\pi_i^* = \gamma_1(i) \tag{13}$$

$a_{i,j}^*$ is the expected number of transitions from state i to state j over the overall number of transitions from state i.

$$a_{i,j}^* = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{14}$$

$b_i^*(\bar{o}_k)$ is the number of expected transitions from state i, when emitted observation is $o_t = \bar{o}_k$, over the number of expected transitions.

$$b_i^*(\bar{o}_k) = \frac{\sum_{t=1}^{T-1} \gamma_t(i) \text{ , if } o_t = \bar{o}_k, \text{ else } 0}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{15}$$

IV. PROPOSED SYSTEM

The proposed system, for APT detection and prediction, undergoes two main phases: attack scenario reconstruction and attack decoding. This work focuses on the second phase of the proposed approach. This section introduces briefly the attack scenario reconstruction and explains the attack decoding phase.

A. ATTACK SCENARIO RECONSTRUCTION

As shown in Figure 3, elementary alerts, raised by individual detection modules or third party IDS, are fed to this phase. These elementary alerts are triggered for each malicious activity observed over the APT life cycle. Each alert has seven attributes: alert type *(alert_type)*, alert time *(timestamp)*, source and destination IP addresses *(src_ip, dest_ip)*, source and destination ports *(src_port, dest_port)* and infected host IP *(infected_host)*. A reconstruction framework has been developed to find alerts that could belong to the same APT scenario. This framework runs through three main steps: (1) Alert filtering, which filters redundant or repeated alerts; (2) Alerts clustering, which clusters alerts that may belong to the same APT scenario; and (3) Correlation indexing, which evaluates the correlations between alerts of each cluster. The correlation is based on matching the attributes of the elementary alerts which are generated over a configurable time window (correlation period).

For each new alert, the alert correlation framework (ACF) checks all stored alerts, which have been triggered over the last time window and proceeds to assign each alert to a specific APT scenario based on the following rules:

- Infected host: Two alerts are considered correlated if they have the same *infect_host attribute*, i.e. the same *src_ip* or *dest_ip*.
- Alert stage: Multiple alerts of different types pointing to a particular stage belong to their respective APT scenario.

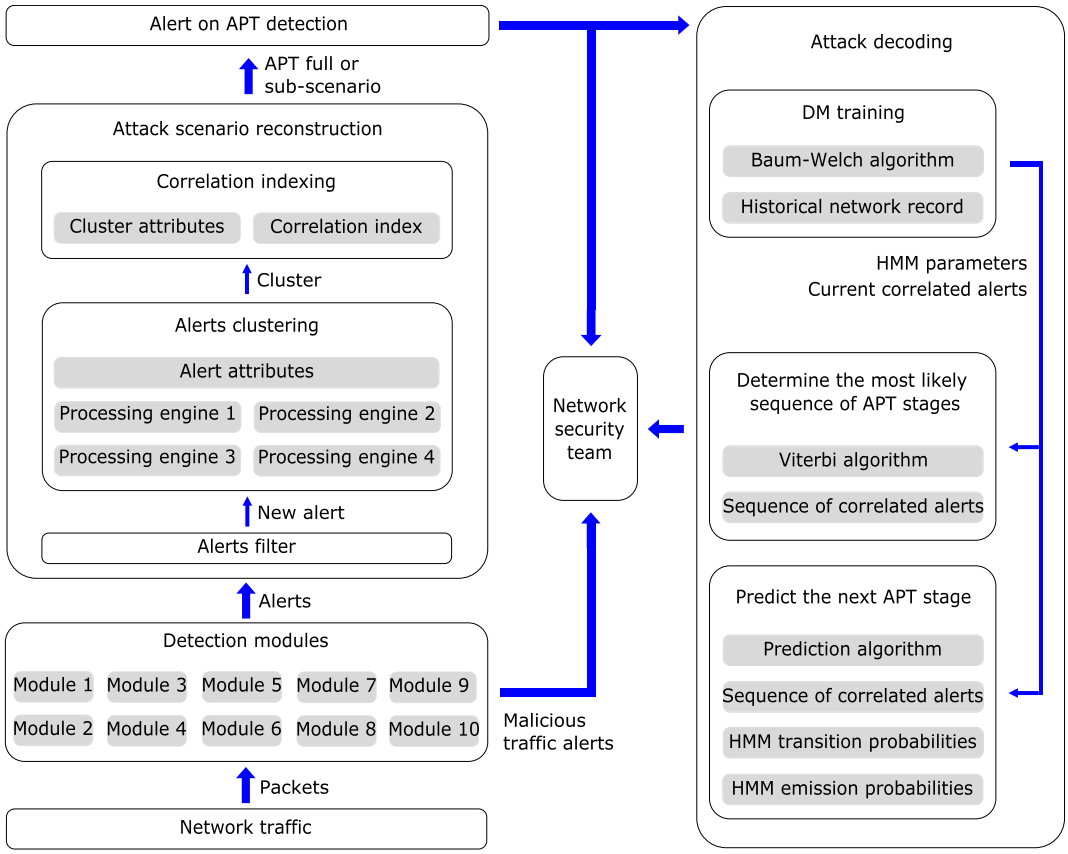

FIGURE 3. The architecture of the proposed system. There are two phases: the first one is attack scenario reconstruction using an alert correlation framework [15] left side, and the second phase is attack decoding using a HMM right side.

- Alert type: Similarly, multiple alerts of the same type pointing to a particular stage belong to their respective APT scenario.
- Alert time: Alerts belonging to a specific APT scenario should be triggered within a correlation time, and correspond to the APT life cycle.

Thus, the ACF generates two types of alerts:

- *apt_full_scenario_alert:* This alert is generated when ACF detects a *full* APT attack scenario during the correlation time. A full attack scenario is the one in which all possible stages are detected.
- *apt_sub_scenario_alert:* This alert is generated when ACF detects a subset of correlated stages during a configurable time window.

For more details about the attack scenario reconstruction phase, the reader is referred to our previous work in [15].

B. ATTACK DECODING

Within the attack decoding phase, based on the current sequence of correlated alerts from the ACF, the following two functions are performed:

- Determine the most likely sequence of APT stages for a given sequence of correlated alerts, using the Viterbi algorithm.
- Predict the next stage of the APT campaign, using the transition and emission probabilities.

These functions assist the network security team to perform forensic analysis on the alerts and proceed by denying the attacker to complete the APT life cycle by mitigating actions against the predicted next stage.

This phase leverages HMM to develop the attack Decoding Module (DM). This module makes use of the ACF output in two ways. Firstly, during training, the ACF correlation dataset can be gathered to construct a historical record of the APT strategies occurring in the monitored network and learn the HMM parameters. Secondly, during testing, the ACF will generate a sequence of alert observations that will be used by the proposed HMM to predict the next APT stage.

As illustrated in Figure 1, the APT life cycle has six stages and the attacker might go through all or several of them to complete the campaign. Table 1 shows the APT stages and alerts generated within the attack scenario reconstruction phase. Specifically, there are eleven types of alert observations considered for the HMM, as seen in Table 1, with the addition of the *nc_alert* generated by ACF.

Additionally, DM considers a non-complete state as a sixth state for the HMM. This state is considered when the correlation time of ACF is passed and the APT campaign does not reach the final stage of APT (data exfiltration). To indicate the sixth state, ACF generates *nc_alert* (non-complete alert) to imply that the correlation time has passed before the current correlated alerts complete the APT scenario. Thus,

TABLE 1. The proposed detection modules for the APT life cycle .

APT stage	Detection modules
Stage 1 Intelligence gathering	For this stage the attacker is using public sources and social engineering techniques, which makes this stage undetectable through network traffic monitoring.
Stage 2 Point of entry	Disguised exe file detection (disguised_exe_alert) [15] Malicious file hash detection (file_hash_alert) [34] Malicious domain name detection (domain_alert) [15]
Stage 3 C&C communication	Malicious IP address detection (ip_alert) [35] Malicious SSL certificate detection (ssl_alert) [15] Domain flux detection (domain_flux_alert) [36]
Stage 4 Lateral movement	Brute force detection (brute_force_alert) [27] Pass the hash detection (pass_hashe_alert) [28]
Stage 5 Asset/Data discovery	Scanning detection (SD) [37]
Stage 6 Data exfiltration	Tor connection detection (TorCD) [38]

DM considers six states for the HMM, which are: point of entry (S_1), C&C communications (S_2), lateral movement (S_3), asset/data discovery (S_4), data exfiltration (S_5) and non-complete state (S_6).

Table 2 shows a sample of 10 APT scenarios from the correlation dataset. Each scenario contains a sequence of correlated alerts corresponding to a known sequence of stages. Note that some of these scenarios are incomplete, i.e. the correlation time passed without ACF being able to link other alerts for the remaining stages. For such a case, ACF generates the *nc_alert*.

TABLE 2. A sample of 10 APT scenarios from the correlation dataset used to train the HMM.

Time_1	Time_2	Time_3	Time_4	Time_5
domain_alert	ip_alert	scan_alert	tor_alert	
disguised_exe_alert	ip_alert	pass_hash_alert	nc_alert	
hash_alert	ssl_alert	brute_force_alert	scan_alert	nc_alert
hash_alert	domain_flux_alert	brute_force_alert	scan_alert	tor_alert
disguised_exe_alert	ssl_alert	nc_alert		
disguised_exe_alert	nc_alert			
domain_alert	ip_alert	brute_force_alert	scan_alert	tor_alert
disguised_exe_alert	ssl_alert	scan_alert	tor_alert	
hash_alert	ip_alert	brute_force_alert	scan_alert	tor_alert
disguised_exe_alert	domain_flux_alert	tor_alert		

To train the HMM, DM applies the Baum-Welch algorithm on the correlation dataset to train the HMM transition, emission and initial probabilities. Using these parameters (A, B, π), DM utilizes the Viterbi algorithm to determine the most likely sequence of APT stages, given the alert observations. To predict the next stage of the attack, DM uses the FW α parameters and the transition probabilities, A, as explained later in the end of this section.

1) DM TRAINING

DM utilizes the historical record of alert observations to learn and optimize the HMM parameters, A, B and π using the Baum-Welch (BW) algorithm as shown in Algorithm 1. In the beginning, the HMM parameters are randomly initialized, as seen on line 2. Then the α and β parameters of the FW and BW algorithms are calculated, on line 3, as shown in equations 2 and 5. As described in Section III-B.3, BW proceeds

Algorithm 1 Implementation Pseudo-Code of DM Training to Learn the HMM Parameters

1: Input: Set of sequence of correlated alerts O_T
2: Randomly initialize the HMM parameters A, B, π
3: Compute FW and BW parameters: $\alpha_{t+1}(i)$, $\beta_t(i)$
$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) a_{i,j} b_j(o_{t+1})$$
$$\beta_t(i) = \sum_{j=1}^{N} a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j)$$
4: Calculate the values $\xi_t(i,j)$, $\gamma_t(i)$
$$\xi_t(i,j) = \frac{\alpha_t(i) a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j)}$$
$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)}$$
5: Update the HMM parameters π_i^*, $a_{i,j}^*$, $b_i^*(v_k)$
$$\pi_i^* = \gamma_1(i)$$
$$a_{i,j}^* = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$
$$b_i^*(\bar{o}_k) = \frac{\sum_{t=1}^{T-1} \gamma_t(i) \text{ , if } o_t = \bar{o}_k \text{, else } 0}{\sum_{t=1}^{T-1} \gamma_t(i)}$$
6: Iterate the previous step until convergence
7: Output optimized A, B, π

by calculating the ξ and γ parameters, as shown on line 4. Finally, BW considers all training observations sequences to iteratively update the HMM parameters based on the equations 13, 14 and 15. At the end, BW will output the learned, optimized $\lambda = (A, B, \pi)$ parameters.

2) DETERMINING THE MOST LIKELY SEQUENCE OF APT STAGES

The most likely stages of the APT life cycle is estimated given a sequence of correlated alerts, O, and the trained HMM, λ. To accomplish this function DM utilizes the Viterbi algorithm to calculate the probability of the observation sequence for each possible path.

Algorithm 2, follows the foundations described in Section III-B.2 to decode a sequence of observations into the most probable sequence of states. In particular, lines 2-6 initialize the δ and ψ parameters. Lines 7-13, calculate the values of δ at each time instance and for each state. Similarly, the prior state that maximizes the likelihood of transitioning to each next state and at each time instance is stored in the ψ variable. Therefore, the ψ variable is very important for identifying the most likely path, given the specific sequence of observation. This task is referred to as backtracking, but before doing this, the last state needs to be identified. So, for the last time instance, $t = T$, the highest probability, $P(T)$, and its respective state, q_T, are determined on lines 14 and 15, respectively. Finally, during the termination/backtracking phase (lines 16-19), the most likely states, at every time instance, are identified using the ψ variable.

3) PREDICTING THE NEXT STEP OF THE APT CAMPAIGN

The next stage of the APT campaign is predicted given the current sequence of the correlated alerts, O, the trained HMM, λ, and leveraging the Forward algorithm. This module

Algorithm 2 Implementation Pseudo-Code of DM Determining the Most Likely Sequence of APT Stages

1: Input: HMM parameters $\lambda = A, B, \pi$, and sequence of correlated alerts O_T.
2: #Initialisation:
3: **for** each state $i = 1, 2, \ldots, N$ **do**
4: $\quad \delta_1(i) = \pi_i \cdot b_i(o_1)$
5: $\quad \psi_1(i) = 0$
6: #Recursion:
7: **for** each observation time $t = 2, 3, \ldots, T$ **do**
8: $\quad$ **for** each possible prior state $i = 1, 2, \ldots, N$ **do**
9: $\quad\quad \delta_t(j) = \max_{1 \le i \le N} \{\delta_{t-1}(i) a_{i,j}\} b_j(o_t)$
10: $\quad\quad \psi_t(j) = \arg\max_{1 \le i \le N} \{\delta_{t-1}(i) a_{i,j}\}$
11: $P(T) = \max_{1 \le i \le N} \delta_T(i)$
12: $q_T = \arg\max_{1 \le i \le N} \delta_T(i)$
13: #Termination:
14: **for** $t = T - 1, \ldots, 1$ **do**
15: $\quad q_t = \psi_{t+1}(q_{t+1})$
16: Output: The most likely sequence of states Q_T.

Algorithm 3 Implementation Pseudo-Code of DM Predicting the Next Step of the APT Campaign

1: Input: HMM parameters $\lambda = A, B, \pi$ and sequence of correlated alerts O_T.
2: **for** each possible next state $j = 1, 2, \ldots, N$ **do**
3: $\quad$ **for** each possible intermediate state $i = 1, 2, \ldots, N$, and their respective observations **do**
4: $\quad\quad$ Calculate the probability $\alpha_t(i)$ as per eq. 2
5: $\quad\quad \alpha_t(i) = \sum_{r=1}^{N} \alpha_{t-1}(r) a(r, i) b_i(o_t)$, where r denotes index of all possible prior states
6: $\quad$ Compute the probability $P(q_{t+1} = s_j)$ by multiplying all α parameters, at time t with their respective transition probabilities.
7: $\quad P(q_{t+1} = s_j) = \sum_{i=1}^{N} \alpha_t(i) a_{i,j}$
8: Determine the two likely next states according to the largest two probabilities on line 8.
9: Output: Prioritised probabilities for the next possible APT stage.

computes the probability of each APT stage to lead to any possible next stage based on the occurred observations and transition probabilities. Algorithm 3 shows the implementation pseudo-code of DM predicting the next step of the APT campaign.

More specifically, the FW α parameters (see eq. 2) of every intermediate stage, i, are calculated (lines 3-6). Considering any possible next stage, j, might occur, the α parameters are multiplied by the transition probability from any intermediate stage, i, to each considered next stage, j, (lines 7-8). Note that this task does not consider any observation for the next time instance, o_{t+1}, as this has not been emitted yet.

The stage that has the greatest probability is predicted as the next stage of the APT campaign. It is helpful to alert the security team in terms of a number of possible likely threats, particularly when the greatest two probabilities are very close. For this reason, in addition to the prediction of the most likely next stage, the second most likely next stage is also predicted (line 10).

V. EVALUATION RESULTS

Due to the lack of publicly available data of APT traffic, we have constructed a 6000 alert synthetic dataset. Out of these 6000 alerts, 3700 are APT alerts and 2300 are uncorrelated alerts, i.e. they do not belong to any APT campaign. The proposed system might not be able to detect all the stages of the APT campaign, yet it still can raise an alert, *apt_sub_scenario_alert*, when two or more stages of APT are correlated.

In a real network, the number of APT alerts should be smaller than the number of uncorrelated alerts. Moreover, it might not be possible to get as many as 3700 APT alerts to train the prediction model. Therefore, a second synthetic dataset has been built with a smaller number of APT alerts. The second dataset contains 361 APT alerts and 2300 uncorrelated alerts. This second dataset has been used to further evaluate the robustness of the prediction model when trained on a smaller number of APT alerts.

The APT alerts were generated to simulate APT scenarios targeting a university campus network, i.e. using random IP addresses from the campus network IP address range. Within each APT scenario, the transition between stages follows the APT life cycle as described in Section III-A. Furthermore, the related alerts within each stage are randomly generated.

The datasets consider the attacker is moving forward within the APT life cycle, i.e. the attacker does not go back to a previous stage. Usually, the attacker can go back to a previous stage if the current attack fails and the attacker has to find another way to complete the APT campaign.In such cases, the attacker's alternative action will trigger a new APT scenario within the simulation.

The sequence of correlated alerts generated within each APT scenario should match at least one of the IP address attributes, i.e. the correlated alerts should either have the same *src_ip*, *dest_ip* or *infected_host*. Moreover, these alerts should be triggered within the same time window, i.e. the *timestamp* difference between the first and final alert of the APT scenario should not exceed the specified correlation time.

As six states were considered for the HMM, Figure 4 shows all possible transitions between the HMM states over five time slots. Note that S_5 represents the final stage of APT, data exfiltration, and S_6 is the non-complete state. Therefore, there are no transitions after these two states. For more details about generating the dataset, the reader is referred to our previous work in [15].

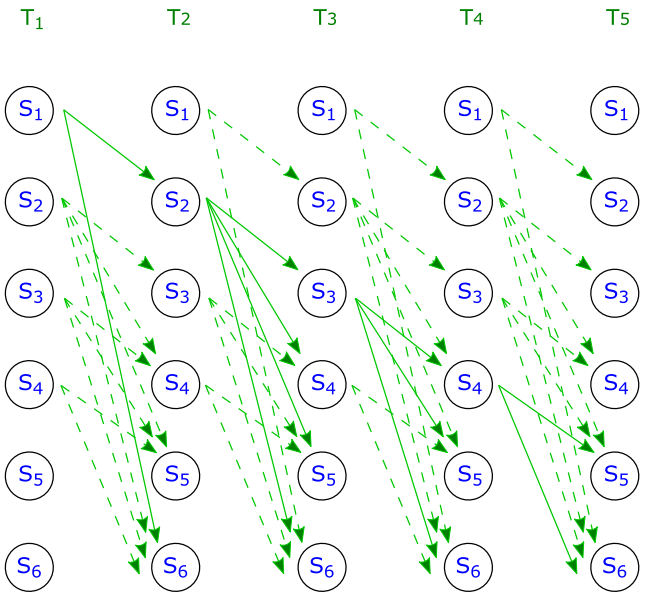

FIGURE 4. All possible transitions between the states of APT. The continuous arrow lines represent all possible transitions starting from S_1. The dashed arrows lines represent all possible transitions starting from the other states $S_2, S_3, \ldots, S_6$.

Once the dataset was created, the ACF was applied to reconstruct the attack scenarios and generate the correlated alerts. The correlation dataset was split into two equal datasets, the first one was used to train the HMM (*train_dataset*) and the second dataset was used to test the DM functions (*test_dataset*).

A. TRAINING THE HMM

To train the HMM, DM applies the Baum-Welch algorithm on the *train_dataset*. Considering the 6 states and 11 observations for the HMM, the following transition, A, and emission, B, probabilities were obtained.

$$A = \begin{bmatrix} 0.0096 & 0.8798 & 0.0096 & 0.0096 & 0.00961 & 0.0817 \\ 0.0098 & 0.0098 & 0.3814 & 0.2593 & 0.1950 & 0.1446 \\ 0.0097 & 0.0097 & 0.0097 & 0.4445 & 0.2999 & 0.2263 \\ 0.0096 & 0.0096 & 0.0096 & 0.0096 & 0.7241 & 0.2374 \\ 0.9523 & 0.0095 & 0.0095 & 0.0095 & 0.0095 & 0.0095 \\ 0.9523 & 0.0095 & 0.0095 & 0.0095 & 0.0095 & 0.0095 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.5454 & 0.2889 & 0.0917 & 0.0093 & 0.0093 & 0.0093 \\ 0.0093 & 0.0093 & 0.0093 & 0.4584 & 0.3360 & 0.1315 \\ 0.0092 & 0.0092 & 0.0092 & 0.0092 & 0.0092 & 0.0092 \\ 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.0091 \\ 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.0091 \\ 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.0091 \end{bmatrix}$$

$$\begin{bmatrix} 0.0093 & 0.0093 & 0.0093 & 0.0093 & 0.0093 \\ 0.0093 & 0.0093 & 0.0093 & 0.0093 & 0.0093 \\ 0.6288 & 0.2886 & 0.0092 & 0.0092 & 0.0092 \\ 0.0091 & 0.0091 & 0.9091 & 0.0091 & 0.0091 \\ 0.0091 & 0.0091 & 0.0091 & 0.9091 & 0.0091 \\ 0.0091 & 0.0091 & 0.0091 & 0.0091 & 0.9091 \end{bmatrix}$$

It is worth noting that both matrices A an B should not have any zero-value element, otherwise, we might get *NaN* error or *zero* probabilities when running the DM algorithms. To avoid this, the training algorithm replaces every zero probability with a very small, but non-zero, value.

B. DM FUNCTIONS

DM initially estimates the sequence of stages based on the first two alerts of the sequence. Then, it keeps updating the predicted sequence of stages when detecting the successive alerts. The Viterbi algorithm was run on each sequence of observations, in the *test_dataset*, utilizing the trained HMM. Then, the estimated states were matched with the ground truth and the prediction accuracy was calculated as,

$$Prediction\_accuracy = n/N, \tag{16}$$

where n is the number of APT scenarios whose stages are correctly estimated, N is the total number of APT scenarios in the *test_dataset*. Table 3 shows the results of DM estimating the most likely stages in terms of prediction accuracy.

TABLE 3. The prediction accuracy of DM estimating the most likely states for a sequence of observations (3700 APT alerts and 2300 uncorrelated alerts).

Number of observations	Prediction accuracy
Two observations	91.80%
Three observations	100%
Four observations	100%
Five observations	100%

These results are based on the number of observations of each sequence. To find the results based on two observations, from the *test_dataset*, only the scenarios which have at least two observations were fed to DM, and similarly for the results based on three, four and five observations. For two-observations sequence, the prediction of the most likely states has an accuracy of 91.80%. Furthermore, by having more than two observations, the system achieves a prediction accuracy reaching 100%.

To evaluate the prediction of the next step of the APT campaign, given the current sequence of the correlated alerts, Algorithm 3 was run on each sequence of observations (each APT scenario), in the *test_dataset*, utilizing the trained HMM. Thus, DM calculates the probability of each APT stage to be the next step. Then, the two stages with the two greatest probabilities are reported as a prediction for the next step of the APT campaign. Table 4 shows the results of DM predicting the next step of the attack. The prediction accuracy is computed for two cases: (1) one-stage prediction, when DM reports only one stage as a prediction for the next step of the attacker; (2) two-stage prediction, when DM reports two stages as a prediction for the next step of the attacker.

For one-stage prediction, predicting the next step based on two observations generates an accuracy of 43.60%. Having only two observations or detecting early stages does not provide enough information to DM for an accurate prediction. The prediction accuracy based on three observations is almost 30% better than the two observation one. Based on four observations, the system achieves a prediction accuracy

TABLE 4. The prediction accuracy of DM predicting the next step of the APT campaign (3700 APT alerts and 2300 uncorrelated alerts).

Number of observations	One-stage prediction accuracy	Two-stage prediction accuracy
Two observations	43.60%	66.50%
Three observations	72.77%	92.70%
Four observations	93.31%	100%

TABLE 5. The prediction accuracy of DM estimating the most likely states for a sequence of observations (361 APT alerts and 2300 uncorrelated alerts).

Number of observations	Prediction accuracy
Two observations	90%
Three observations	100%
Four observations	100%
Five observations	100%

TABLE 6. The prediction accuracy of DM predicting the next step of the APT campaign (361 APT alerts and 2300 uncorrelated alerts).

Number of observations	One-stage prediction accuracy	Two-stage prediction accuracy
Two observations	40.00%	61.00%
Three observations	63.74%	90.11%
Four observations	91.74%	100%

of 93.31%. Thus, correlating more alerts, which are added to the sequence of observation, significantly improves the DM performance. This is because the probabilities of these observations contribute to the estimation of the next step. Nevertheless, predicting the next step based on a less number of observations provides the network security team with an early alarm to perform more forensics and mitigate the attack.

For two-stage prediction, reporting two most likely stages rather than one, as a prediction of the next step, has significantly improved the system performance. Based on two and three observations, the prediction accuracy is about 20% better than the one-stage approach. Based on four observations, the system yields an accuracy reaching 100%. The better performance of the two-stage prediction is achieved by avoiding the incorrect predictions when the values of the two greatest probabilities are very close to each other.

Tables 5 and 6 show further evaluation results using the smaller synthetic dataset having 361 APT alerts. For estimating the most likely sequence of states, the prediction accuracy based on two observations is just 1.8% less than that gained when training on the larger dataset. The prediction based on three, four and five observations remains perfect with an accuracy of 100%.

Regarding the next step of the attacker, the prediction accuracy of the one-stage prediction approach is about $2\% - 9\%$ less than that produced when training on the larger dataset. For the two-stage prediction approach, based on two and three observations, the prediction accuracy decreases by only 5.5% and 2.59%, respectively. However, based on four observations, the system yields an accuracy reaching 100%.

In general, the system yields significant results even when a smaller and more realistic number of APT alerts are used for training. This indicates the robustness of our system in terms of predicting the next step of the attacker.

VI. CONCLUSION

We proposed a probabilistic IDS for APT detection and prediction. The proposed approach runs through two main phases: the first one is for attack scenario reconstruction using a correlation framework, and the second phase is for attack decoding using an HMM. First, the APT scenario is reconstructed by linking alerts which are observed during an APT life cycle. The correlation of these alters is based on matching the attributes of the generated alerts over a configurable time window. Second, the sequence of the correlated alerts are fed to a decoding module, which utilizes the HMM, to achieve two functions. The first one uses Viterbi algorithm to estimate the most likely sequence of APT stages for the sequence of alerts correlated in the attack reconstruction phase. The second function predicts the next step of the attack campaign. This prediction algorithm makes use of the HMM parameters to compute the probability of each APT stage to be the next possible step of the attacker. Then, the stage of the highest probability is predicted as the next step.

For the first dataset considered, the proposed system estimates the sequence of APT stages with a prediction accuracy of 91.80%, for two-observation sequence, and 100% for a sequence of more than two observations. Forecasting the APT stages not only reveals the APT life cycle in its early stages but also helps to understand the attacker's strategies and aims. For predicting the next step of the attacker, the prediction accuracy depends on the number of observations. The prediction accuracy will be higher, as more information is fed to the prediction algorithm. For one-stage prediction, this system has an accuracy of 43.60%, 72.77% and 93.31% based on two, three and four observations, respectively. Furthermore, the two-stage prediction improves the accuracy by nearly 20% based on two and three observations. Moreover, the prediction accuracy reaches 100% when four observations are used. A similar performance has also been seen for reduced number of training data samples demonstrating robustness of the methods against size of training dataset. Predicting the next step of the attacker plays a key role in the attack response and enables the network security team to take the required actions before the attacker reaches the final stage of data exfiltration.

REFERENCES

[1] The Guardian. *Petya' Ransomware Attack: What is it and How can it be Stopped?*. Accessed: May 1, 2019. [Online]. Available: https://www.theguardian.com/technology/2017/jun/27/petya-ransomware-cyber-attack-who-what-why-how

[2] I. Ghafir, K. G. Kyriakopoulos, F. J. Aparicio-Navarro, S. Lambotharan, B. Assadhan, and H. Binsalleeh, "A basic probability assignment methodology for unsupervised wireless intrusion detection," *IEEE Access*, vol. 6, pp. 40008–40023, 2018.

[3] McAafee-report. (2018). *The Economic Impact of Cybercrime—No Slowing Down*. [Online]. Available: https://www.mcafee.com/enterprise/en-us/assets/executive-summaries/es-economic-impact-cybercrime.pdf

[4] N. Nissim, A. Cohen, C. Glezer, and Y. Elovici, "Detection of malicious PDF files and directions for enhancements: A state-of-the art survey," *Comput. Secur.*, vol. 48, pp. 246–266, Feb. 2015.

[5] D. Santoro, G. Escudero-Andreu, K. G. Kyriakopoulos, F. J. Aparicio-Navarro, D. J. Parish, and M. Vadursi, "A hybrid intrusion detection system for virtual jamming attacks on wireless networks," *Measurement*, vol. 109, pp. 79–87, Oct. 2017.

[6] F. J. Aparicio-Navarro, K. Kyriakopoulos, I. Ghafir, S. Lambotharan, and J. Chambers, "Multi-stage attack detection using contextual information," in *Proc. IEEE/AFCEA Mil. Commun. Conf. (MILCOM)*, Los Angeles, CA, USA, Oct. 2018, pp. 1–9.

[7] Center-Mandiant-Intelligence, "Apt1: Exposing one of chinas cyber espionage units," Mandiant, Alexandria, VA, USA, Tech. Rep., 2013.

[8] Z. Ghahramani, "An introduction to hidden Markov models and Bayesian networks," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 15, no. 1, pp. 9–42, Jun. 2001.

[9] X. D. Huang, Y. Ariki, and M. A. Jack, *Hidden Markov Models for Speech Recognition*. Edinburgh, U.K.: Edinburgh Univ. Press, 1990.

[10] Y. Wen, "Text mining using HMM and PPM," Ph.D. dissertation, Dept. Comput. Sci., Univ. Waikato, Hamilton, New Zealand, 2001.

[11] H. Bunke and T. Caelli, *Hidden Markov Models: Applications in Computer Vision*, vol. 45. Singapore: World Scientific, 2001, pp. 91–107.

[12] R. J. Boys, D. A. Henderson, and D. J. Wilkinson, "Detecting homogeneous segments in DNA sequences by using hidden Markov models," *J. Roy. Stat. Soc., C, Appl. Statist.*, vol. 49, no. 2, pp. 269–285, 2000.

[13] Z. Anming and J. Chunfu, "Study on the applications of hidden Markov models to computer intrusion detection," in *Proc. 5th World Congr. Intell. Control Automat.*, Hangzhou, China, vol. 5, Jun. 2004, pp. 4352–4356.

[14] S. Shin, S. Lee, H. Kim, and S. Kim, "Advanced probabilistic approach for network intrusion forecasting and detection," *Expert Syst. Appl.*, vol. 40, no. 1, pp. 315–322, 2013.

[15] I. Ghafir, M. Hammoudeh, V. Prenosil, L. Han, R. Hegarty, K. Rabie, and F. J. Aparicio-Navarro, "Detection of advanced persistent threat using machine-learning correlation analysis," *Future Gener. Comput. Syst.*, vol. 89, pp. 349–359, Dec. 2018.

[16] I. Ghafir, "Dataset of advanced persistent threat (APT) alerts," Lougborough Univ., Loughborough, U.K. Accessed: Jan. 15, 2019. doi: 10.17028/rd.lboro.7577750.

[17] M. Balduzzi, V. Ciangaglini, and R. McArdle, "Targeted attacks detection with SPuNge," in *Proc. 11th Annu. Conf. Privacy, Secur. Trust*, Jul. 2013, pp. 185–194.

[18] J. Sigholm and M. Bang, "Towards offensive cyber counterintelligence: Adopting a target-centric view on advanced persistent threats," in *Proc. Intell. Secur. Inform. Conf. (EISIC)*, Uppsala, Sweden, Aug. 2013, pp. 166–171.

[19] G. Brogi and V. V. T. Tong, "Terminaptor: Highlighting advanced persistent threats through information flow tracking," in *Proc. 8th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Nov. 2016, pp. 1–5.

[20] X. Wang, K. Zheng, X. Niu, B. Wu, and C. Wu, "Detection of command and control in advanced persistent threat based on independent access," in *Proc. IEEE Int. Conf. Commun.(ICC)*, Kuala Lumpur, Malaysia, May 2016, pp. 1–6.

[21] J. V. Chandra, N. Challa, and S. K. Pasupuleti, "A practical approach to E-mail spam filters to protect data from advanced persistent threat," in *Proc. Int. Conf. Circuit, Power Comput. Technol. (ICCPCT)*, Mar. 2016, pp. 1–5.

[22] G. Brogi and E. Di Bernardino, "Hidden Markov models for advanced persistent threats," *HAL Arch.*, pp. 1–5, Oct. 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01549196/

[23] H. A. Kholidy, A. Erradi, S. Abdelwahed, and A. Azab, "A finite state hidden Markov model for predicting multistage attacks in cloud systems," in *Proc. IEEE Int. Conf. Dependable, Autonomic Secure Comput. (DASC)*, Aug. 2014, pp. 14–19.

[24] H. Debar, D. Curry, and B. Feinstein, *The Intrusion Detection Message Exchange Format (IDMEF)*, document 4765, 2007.

[25] Trend Micro. *The Custom Defense Against Targeted Attacks*. Accessed: May 15, 2019. [Online]. Available: http://www.trendmicro.fr/media/wp/custom-defense-against-targeted-attacks-whitepaper-en.pdf

[26] S. Yadav, A. K. K. Reddy, A. L. N. Reddy, and S. Ranjan, "Detecting algorithmically generated domain-flux attacks with DNS traffic analysis," *IEEE/ACM Trans. Netw.*, vol. 20, no. 5, pp. 1663–1677, Oct. 2012.

[27] K. Apostol, *Brute-Force Attack*. Leechburg, PA, USA: SaluPress, 2012.

[28] J. R. Johnson and E. A. Hogan, "A graph analytic metric for mitigating advanced persistent threat," in *Proc. IEEE Int. Conf. Intell. Secur. Inform.*, Jun. 2013, pp. 129–133.

[29] A. K. Kaushik, E. S. Pilli, and R. C. Joshi, "Network forensic system for port scanning attack," in *Proc. IEEE 2nd Int. Advance Comput. Conf. (IACC)*, Feb. 2010, pp. 310–315.

[30] R. Jansen, F. Tchorsch, A. Johnson, and B. Scheuermann, *The Sniper Attack: Anonymously Deanonymizing and Disabling The Tor Network*, DTIC, Office Naval Res., Arlington, VA, USA, 2017.

[31] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, vol. 3. London, U.K.: Pearson, 2017.

[32] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[33] N. M. Ryan, "Citation data-set for machine learning citation styles and entity extraction from citation strings," May 2018, *arXiv:1805.04798*. [Online]. Available: https://arxiv.org/abs/1805.04798

[34] I. Ghafir and V. Prenosil, "Malicious file hash detection and drive-by download attacks," in *Proc. 2nd Int. Conf. Comput. Commun. Technol.*, vol. 379. New Delhi, India: Springer, 2016, pp. 661–669.

[35] I. Ghafir and V. Prenosil, "Blacklist-based malicious IP traffic detection," in *Proc. Global Conf. Commun. Technol.*, Tamil Nadu, India, Apr. 2015, pp. 229–233.

[36] I. Ghafir and V. Prenosil, "DNS query failure and algorithmically generated domain-flux detection," in *Proc. Int. Conf. Frontiers Commun., Netw. Appl.*, Kuala Lumpur, Malaysia, Nov. 2014, pp. 1–5.

[37] Bro-Project. *TCP Scan Detection*. Accessed: Dec. 17, 2018. [Online]. Available: https://www.bro.org/sphinx/_downloads/scan.bro

[38] I. Ghafir, J. Svoboda, and V. Prenosil, "Tor-based malware and Tor connection detection," in *Proc. Int. Conf. Frontiers Commun., Netw. Appl.*, Kuala Lumpur, Malaysia, Nov. 2014, pp. 1–6.

IBRAHIM GHAFIR received the Ph.D. degree in computer science from Manchester Metropolitan University, U.K. He is currently a Research Associate with the Wolfson School Mechanical, Electrical and Manufacturing Engineering, Loughborough University, U.K. His research interests include several areas in computer science, including cyber security, network traffic analysis, anomaly detection, wireless communication, and performance optimization.

KONSTANTINOS G. KYRIAKOPOULOS (M'07) received the B.Sc. degree in electrical engineering from the Technological Education Institute of Larisa, Greece, in 2003, and the M.Sc. degree in digital communication systems and the Ph.D. degree in computer networks from Loughborough University, Loughborough, U.K., in 2004 and 2008, respectively.

From 2008 to 2016, he was a Research Associate with the School of Electronic, Electrical, and Systems Engineering, Loughborough University, working mainly in EPSRC projects and successfully licensing research output from his work. Since 2016, he has been an Academic Member with the Wolfson School of Mechanical, Electronic and Manufacturing Engineering, Loughborough University, and the Institute of Digital Technologies, Loughborough University London, London, U.K. His research interests include the areas of computer networks, including network security, intrusion detection, vehicular communications, intelligent decision making based on network situational awareness, and network performance measurements in emerging network paradigms and their applications.

SANGARAPILLAI LAMBOTHARAN received the Ph.D. degree in signal processing from Imperial College London, U.K., in 1997, where he remained as a Postdoctoral Research Associate, until 1999.

He was a Visiting Scientist with the Engineering and Theory Centre, Cornell University, USA, in 1996. From 1999 to 2002, he was with Motorola Applied Research Group, U.K., and investigated various projects, including physical link layer modeling and performance characterization of GPRS, EGPRS, and UTRAN. He was with Kings College London and Cardiff University as a Lecturer and a Senior Lecturer, from 2002 to 2007, respectively. He is currently a Professor of digital communications and the Head of Signal Processing and Networks Research Group, Wolfson School Mechanical, Electrical and Manufacturing Engineering, Loughborough University, U.K. His current research interests include 5G networks, MIMO, radars, smart grids, machine learning, network security and convex optimizations, and game theory. He has published approximately 200 technical journal and conference articles in these areas.

FRANCISCO J. APARICIO-NAVARRO received the B.Eng. degree in telecommunications engineering, specialized in computer networks, from the Technical University of Cartagena, Spain, in 2009, and the Ph.D. degree in computer network security from Loughborough University, Loughborough, U.K., in 2014. From 2013 to 2018, he was a Research Associate with the School of Electronic, Electrical, and Systems Engineering, Loughborough University, and the School of Engineering, Newcastle University, Newcastle upon Tyne, U.K. Since 2018, he has been a Lecturer (Assistant Professor) in cyber security with the Faculty of Computing, Engineering and Media, De Montfort University, Leicester, U.K., a GCHQ/EPSRC Academic Centre of Excellence in Cyber Security Research (ACE-CSR). He is also an Expert in the areas of computer networks, cyber security, and anomaly detection. He is also a Certified Ethical Hacker/Penetration Tester. He is currently an Academic Visitor with the Signal Processing and Networks Research Group, Loughborough University. His research interests include the areas of intrusion detection, computer networks, connected vehicles, and cyber–physical systems cyber security.

BASIL ASSADHAN received the M.S. degree in electrical and computer engineering from the University of Wisconsin and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University. He is currently an Assistant Professor with the Electrical Engineering Department, King Saud University. His research interests include the areas of cybersecurity, network security, network traffic analysis, and anomaly detection.

HAMAD BINSALLEEH received the master's degree in information systems security and the Ph.D. degree in computer science degree from Concordia University, Canada. He is currently an Assistance Professor with Imam Muhammad Ibn Saud Islamic University. He has published several research papers in journals and conferences. His current research interests include cybersecurity, network security, network traffic analysis, anomaly detection, passive DNS traffic analysis, malware analysis, and malware reverse engineering. He has been awarded many honors, including international recognition and best papers.

DIAB M. DIAB received the M.S. degree in computer science from The University of Jordan and the Ph.D. degree in computer science from King Saud University. He is currently a full-time Researcher with King Saud University. He is also doing his Postdoctoral research under U.K.–Gulf Institutional Links Program (Loughborough University and King Saud University). His research interests include the areas of machine learning, classification algorithms, instance weighing, ensembles of classifiers, similarity distance metrics, neural networks, cybersecurity, network security, network traffic analysis, and anomaly/intrusion detection.