Serdica Journal of Computing 15(2), 2021, pp. 129-147, 10.55630/sjc.2021.15.129-147
Published by Institute of Mathematics and Informatics, Bulgarian Academy of Sciences

Markov Models for Malware and Intrusion Detection: A Survey

Evgeniya Nikolova

Faculty of Computer Science and Engineering,
Burgas Free University, Bulgaria
Institute of Mathematics and Informatics,
Bulgarian Academy of Sciences, Bulgaria
enikolova@bfu.bg

Abstract

Malicious attacks are one of the main threats facing today's most used Android and Windows operating systems, as well as the Internet of Things (IoT) and web environments. Markov models and hidden Markov models have been used successfully over the past few decades to identify a variety of malicious activity, including as viruses, worms, Trojan horses, rootkits, ransomware, and phishing assaults. But they have their limits. One of their main limitations is that they are unable to detect subtle changes in malicious behaviour. This paper presents Markov models and hidden Markov models as a tool for detecting malicious attacks and briefly reviews different studies from the past five years that use these models as a detection tool. This review, based on publications drawn from three databases, outlines the continuing interest of security researchers in these models. Most of the chosen research papers show that these models are applied to create systems that have a detection accuracy of malicious attacks above 94%. This study can be helpful to beginners who are interested in starting their research in the field of detecting malicious attacks.

Keywords: Markov model, hidden Markov model, malware, intrusion detection system

ACM Computing Classification System 2012: Mathematics of computing → Probability and statistics → Stochastic processes → Markov processes; Security and privacy → Intrusion/anomaly detection and malware mitigation

Mathematics Subject Classification 2020: 62M05

Received: February 20, 2023, *Accepted:* March 21, 2023, *Published:* March 24, 2023

Citation: Evgeniya Nikolova, Markov Models for Malware and Intrusion Detection: A Survey, Serdica Journal of Computing 15(2), 2021, pp. 129-147, https://doi.org/10.55630/sjc.2021.15.129-147

1 Introduction

Computer viruses are intended to propagate through programs and systems to disrupt operations, result in significant operational problems, and cause data loss and leakage. There are many distinct types of viruses, and each type has a specific mechanism and is utilized for a certain purpose [1]. The term "malware" is shorthand for "malicious software," which encompasses viruses, worms, Trojan horses, spyware, adware, ransomware and etc.. The attacks are attempts to exploit a system or network using malicious software or other means. In other words, malware is a type of attack, but not all attacks involve malware. In many cases, malware uses stealth and mimicry techniques, either by mimicking benign code or known processes. It can hide from both traditional and next-generation anti-malware solutions. Kaur [2] offered a chronology and taxonomy of malware in 2019. In the literature, several taxonomies have been presented to aid in the comprehension of current threats, the creation of defines mechanisms, and to be used to identify open problems, such as [3–5].

Statistics from [6] for the years 2015 to 2022 show that the first half of 2022 saw 2.8 billion malware attacks worldwide. Malware is becoming increasingly sophisticated and is designed to avoid detection by intrusion detection systems (IDS)/system to detect malicious activity. Therefore, finding unknown and hidden malware is the most difficult problem for the designers of such systems.

Various techniques have been developed to detect viruses/malware over the years, most of which can be attributed to one of the following: static signature-based methods, generic signature scanning, heuristic analysis, integrity checking, and machine learning techniques. Numerous publications have described these techniques, along with their advantages and limitations [7, 8].

Static, dynamic, and hybrid approaches are the main types of malware analysis techniques. Static malware analysis performs similarly to statistical-based and signature-based analysis, and as such, frequently incorporates functions of both techniques. Dynamic analysis techniques monitor and track a suspected program's or application's activity while it runs. In hybrid analysis, run-time data acquired from dynamic analysis is integrated into a static analysis algorithm to detect behaviour or malicious functionality, covering each other's shortcomings from the two previous methodologies. Many studies compare static, dynamic, and hybrid analysis methodologies and examine the advantages and disadvantages of each type of analysis [8–11].

In the construction of such systems, various models are used to characterize normal behaviour, such as Markov models and hidden Markov models [12]. The Markov model searches for patterns that indicate malicious activity by exam-

ining a system's behaviour. It can find patterns to identify malware even if the virus is unknown. Because the model employs a probabilistic approach to detecting malicious conduct, it can see patterns in the data that can be suggestive of harmful activity. To identify suspicious behaviour, the model also considers how frequently specific behaviours occur. By examining the typical malware obfuscation techniques and contrasting the many research studies that utilize HMM as a detection tool, Ling and Sani [13] in 2017 introduced the Hidden Markov Model as an efficient metamorphic tool for malware detection. In a comprehensive analysis of intrusion detection methods based on hidden Markov models that was published in 2018, Ramaki, Rasoolzadegan, and Jafari [14] highlight the following six key benefits of these methods: The main benefits include accurate intrusion detection, the capacity to identify new intrusions, the capacity to predict an attacker's probable next steps, the ability to be used in real-time applications by processing data streams on the fly, the use of heterogeneous data sources as input, and the ability to visualize the knowledge acquired in comparison to other machine learning techniques.

The purpose of this study is to determine whether there is sustained researcher interest in the application of Markov models/HMMs in the field of malware detection systems/IDS. The following research questions are posed to achieve the stated objective: 1. In what aspects of computer security are Markov models/HMMs used? 2. Are they used in the development of malware detection systems/IDS for various platforms? 3. How successful are IDS and malware detection systems based on Markov/HMM? 4. Do the proposed Markov/HMM model-based methods provide answers to new security issues?

This paper is divided into the following sections. Section 2 presents the research methodology. A statistical analysis of the literature under study is provided in Section 3. Markov models and the hidden Markov model are briefly reviewed in Section 4. Finally, Section 5 provides an overview of the most cited publications of the last five years about Markov models and Hidden Markov Models for Malware Detection.

2 Research methodology

Finding answers to the questions asked requires a review of the literature on the subject. The systematic literature review was implemented according to the guidelines presented in [15].

To conduct a literature review, we examined several articles and papers from various journals and conferences based on the formulated research questions

Database	*Returned papers*	*Irrelevant papers*	*Extracted papers*
Google Scholar	1 010	507	503
Scopus	123	27	96
IEEE Xplore Digital Library	206	74	132

Table 1: A summary of the material taken from academic libraries and the filtration procedure.

mentioned in the introduction. For this study, we used the Google Scholar search library (https://scholar.google.com/), the Scopus bibliographic and reference database (https://www.scopus.com/), and the IEEE Xplore Digital Library database (https://ieeexplore.ieee.org/). To retrieve only relevant studies focused on Markov models and techniques used in malware detection and intrusion detection systems, we identified keywords based on the research questions. The formulated search query is using "AND" and "OR" operators were entered into each library's search engine to retrieve relevant articles and reports published in the period 2017–2022. The period of the last six years was chosen because the study focuses on the latest scientific research. Articles and reports were extracted from the databases by title and abstract analysis of the publications. To exclude papers not related to the topic of the current study, all selected papers were read in full. The resulting set of publications was examined to answer the research questions posed.

A search query must be formulated, containing all important keywords, names, synonyms, and abbreviations associated with the logical "OR" and "AND" operators. The final blurb includes various keywords that may appear in articles and reports. Its wording is: ("Markov model" or "Markov techniques" or "Markov" or "Markov chain" or "HMM") and ("malware") and ("detection" or "detect" or "identification" or "intrusion detection").

The selected libraries were searched using the formulated search query, filtering the search results to include only relevant articles published in 2017–2022. The search query returned a total of 1339 publications. The first step was to reject any papers whose title or abstract did not match with the study's goals. Detailed search results for each library are shown in Table 1.

The list is then reduced by removing duplicate posts from the three databases. As a result, 226 publications were obtained. Inclusion and exclusion rules were used to obtain a final list of publications.

Inclusion criteria

1. The paper should present an intrusion or malware detection system based on a Markov model, or HMM.
2. The article must be published in a scientific journal.
3. The article should have citations.
4. The article must be published in the period 2017–2022.

Exclusion criteria

1. Reports or articles focusing on other aspects of the impacts of malware detection and intrusion detection systems
2. Blog posts or reports
3. Review reports/articles
4. Articles not published in a scientific journal

Reading of the selected literature outlined which types of threats and platforms researchers are targeting and how accurately Markov model-based systems can detect security threats when trained with a training dataset.

3 Statistical analysis of retrieved and filtered literature

The analysis of the documents that were finally selected starts with a statistical analysis. This analysis will answer the following questions: to what extent the Markov model is preferred by malware researchers, and which platforms are the focus of the researchers who use the Markov model when building IDS.

Figures 1-2 present the distributions of the final selected documents from the respective databases by years for the period 2017–2022. The graphs in the figures show that researchers are using these models to develop more advanced systems, and interest in Markov models and HMMs is ongoing.

What percentage of the papers these publications make up in relation to the entire number of articles addressing malware detection is an intriguing question. As it turns out, the percentage fluctuates throughout time, going from 1.91% in 2018 to 1.02% in 2022. The overall number of articles on this topic increased by 1.8 times in the Scopus database between 2017 and 2022, while those that focused on the use of Markov processes or HMM saw a rise of 1.1. In this aspect, in recent years, a downward trend in the number of these publications has been observed.

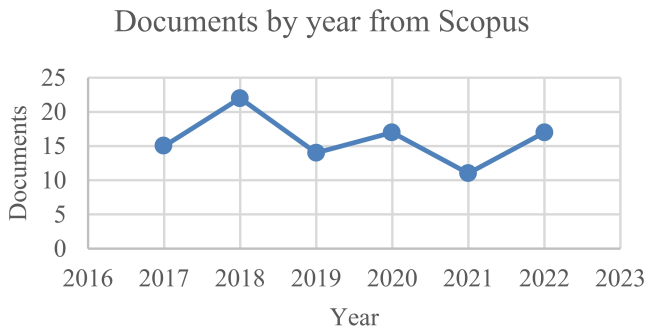

Figure 1: Distribution of documents retrieved from Scopus by years.

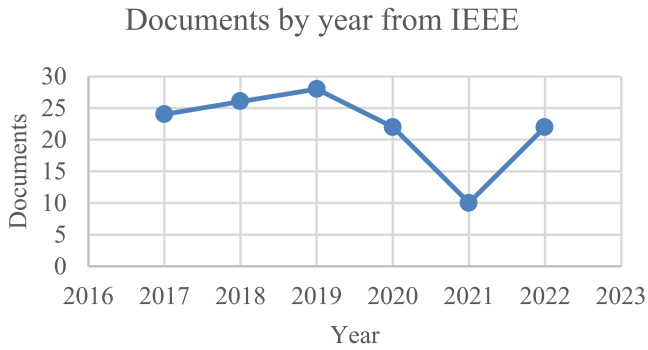

Figure 2: Distribution of documents retrieved from IEEE by years.

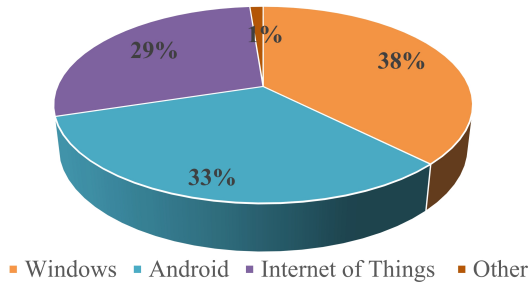

Figure 3: Platform-wise distribution of the research papers.

There are many intrusion detection/malware detection systems based on Markov models/HMMs that are specifically designed for the Windows platform, such as [16–19]. Similarly, there are systems specifically designed for the Android platform, such as MaMaDroid, CANDYMAN, ProDroid, XDroid, as well as for the IoT platform, such as OwlEye. The examination of the selected papers revealed that, even though different researchers have concentrated on various platforms for malware detection, the two platforms – Windows and Android – have received most of the attention. Researchers are focusing almost as much on malware on the Internet of Things environment as they do on these two platforms. Distribution of selected documents by platform is shown in Figure 3. But if only the publications with citations are considered, then the attention of researchers is primarily directed to malware detection systems for Android (45%).

Accuracy is the most important performance indicator of IDS/malware detection systems, which assesses the system's ability to classify normal and abnormal behavior. Other commonly used performance metrics are recall and F-score. Recall is the ratio of correctly observed positives to the total observations in a given class. The F-score is the harmonic mean of precision and recall. In most cases, systems based on Markov models or HMMs achieve high accuracy. The effectiveness achieved by the systems outlined in Section 5 is shown in Tables 2-3. Table 2 demonstrates that the achieved accuracy is over 97% for the most of Markov model-based systems and over 94% for HMM-based systems (Table 3).

Data quality is a fundamental component for malware/attack detection systems. In the literature, several publicly available malware/attack datasets are used by researchers to carry out their research activities. In certain instances, researchers create their own databases, like in [28]. Researchers have examined a variety of assaults and viruses from various families in the Windows environment via API calls (Application programming interfaces – API). Tables 2-3 show that different publicly accessible data are used in this situation for training and testing the algorithms, while for Android-based systems, the DREBIN [29] database is mainly used.

Viruses, worms, Trojan horses, rootkits, ransomware, and phishing attacks are just a few examples of the types of malicious behaviour that a system based on Markov models or HMMs can identify. ProDroid, XDroid, MaMaDroid can detect a wide variety of malicious behaviour, including viruses, rootkits, and ransomware. OwlEye can detect viruses, worms, Trojans, rootkits, ransomware, and phishing attacks. It is noteworthy that in very few publications the authors mention the possibility of the system reporting the presence of new

Reference	*Platform*	*Dataset used*	*Accuracy/F-score*
[20]	Android	benign Google Play applications malware families [21]	F-score = 98.2773%
[22]	Android	DREBIN	Accuracy = 81.8%
MaMaDroid	Android	DREBIN	F-score > 99%
[16]	Windows		Accuracy = 97.3%
[18]	Windows	CSDMC2010 Dataset, https://github.com, [24], [23]	Accuracy = 99%
[19]	Windows	sequences of API calls from [24–26]	Accuracy = 99%

Table 2: Summary of the data extracted from the literature on Markov model-based malware detection.

Reference	*Platform*	*Dataset used*	*Accuracy/F-score*
[27]	Android	DREBIN	Accuracy = 80%
[28]	Internet of Things	Self-generated	Recall = 98.6%
[17]	Windows	VirusShare database	Accuracy > 94%
[30]	Android	DREBIN	Accuracy = 94.5%

Table 3: Summary of the data extracted from the literature on HMM-based malware detection.

malware/attack. In the presented literature review, there is one example of a malware detection system that can detect new malware families that even appear years after the system was trained, and that is MaMaDroid. Another important issue, the detection of mimic malware, is also rarely discussed in the literature on malware detection systems/IDS based on Markov models or HMMs. One example is found in the literature review – publication [19].

4 Markov models

For evaluating statistical patterns in a variety of applications, including virus detection, Markov models have shown to be a useful tool. They will be introduced in this section briefly.

The Markov property, first described by Russian mathematician Andrey Markov in 1906, is included in Markov models [31]. An outcome is only predicted based on the information supplied by the current state, not on the previous series of occurrences. The Markov model tries to explain a random process that is dependent only on the present event and not on earlier occurrences. The Markov Chain Model is used when the states of a dynamic system are completely observable, whereas the Hidden Markov Model is used when the states of the system are only partially visible.

All Markov models have two fundamental elements: a collection of states and a set of transitions between those states. The model operates in the following way: the system is always in one of the states throughout the time period of interest. The system can only be in one state, and occasionally it will move from one state to another by executing one of the interstate transitions. Discrete Time Markov Chain and Continuous Time Markov Chain are the two types of models.

When an observation X_t at time t is produced by a stochastic process but the process's state Z_t cannot be directly observed, Hidden Markov model is a tool for modeling probability distributions over sequences of observations [32, 33]. It is assumed that this hidden process satisfies the Markov condition, where state Z_t at time t only depends on state Z_{t-1} at time $t-1$. This is known as the first-order Markov model. The n-th-order Markov model depends on the n previous states.

Variants of HMMs, such as profile HMMs, pairwise HMMs, and context-sensitive HMMs, are used for different sequence analysis problems. In 1994, Anders Krogh and colleagues [34] presented an extended implementation of HMM, called the Profile Hidden Markov Model (PHMM), for modelling sequence similarity in DNA and proteins. They are probabilistic models that encapsulate the evolutionary changes that have occurred in a set of related sequences. PHMM is a highly linear, left-right model that contains three classes of states: the match state, the insert state, and the delete state; and two sets of parameters: transition probabilities and emission probabilities. The match and insert states are states that emit a character, while the delete state is a state with no probability of emission. The emission probability is the probability of emission of a character x from the alphabet α that is in state q. The transition

probability is the probability of going from one state to another. The fact that PHMM contains match, insert, and delete states whereas a conventional HMM does not allow insertions or deletions is a key distinction between PHMM and HMM. A PHMM explicitly takes positional information included in sequences into consideration, whereas an HMM does not. The strengths and weaknesses of PHMM for detecting metamorphic viruses have been discussed in several articles [35,36].

5 Literature review

The main challenges facing researchers in building a malware detection system include: identifying the features of malicious code; designing and implementing effective detection algorithms; accurately detecting new and unknown threats; and accurately classifying malicious files. Additionally, researchers must also consider the computational resources and time required to build a system, as well as the cost of maintaining and updating the system. Finally, researchers must also consider the privacy implications of their system, as well as the potential for misuse.

Markov models can provide a solution for the challenges of accurately detecting new and unknown threats, as they are capable of recognizing patterns in dynamic environments. Additionally, Markov models can also help to accurately classify malicious files, as they are able to consider the probability of certain actions being taken by malware. Finally, Markov models can also help reduce the computational resources and time required to build a system, as they require less data to be stored to make accurate predictions. Numerous articles on the use of Markov and HMM models for malware detection in Windows, Android, and IoT have been published during the past five years. An overview of the most significant (mostly cited) papers is provided in this section.

5.1 Markov models for malware detection

In the last years, Markov models has become an important tool for cybersecurity professionals and is used for a variety of purposes, such as identifying malicious software – worms, viruses, and Trojans, detecting intrusions, and finding anomalies in network traffic – unexpected traffic patterns or unusual amounts of traffic coming from a single source. Recently, Markov models have been used to detect sophisticated attacks, such as those that use machine learning or artificial intelligence. In addition, the model has been used to identify

malicious actors and their activities, as well as to detect malicious websites and online services.

The implementation of a Markov model for detecting malware typically involves the following steps:

1. Data collection: collecting data from the network, either from logs or network traffic.
2. Pre-processing: pre-processing the data to identify patterns that indicate malicious activity.
3. Modelling: Building a Markov model to model the data and identify suspicious patterns.
4. Evaluation: Evaluating the model against a test set of data to ensure accuracy.
5. Deployment: Deploying the model in the network to detect malicious activity.

In [20], a novel malware detection strategy is offered that uses a back-propagation neural network to identify malware and interprets a series of system calls as a homogenous stationary Markov chain. Character extraction from system call sequences is done using homogeneous stationary Markov chains. Since it is assumed that each system call corresponds to a unique state of the Markov chain, the transition probability matrix may be calculated by counting the number of transitions from one system call to the next. After generating the transition probability matrices for each application in the training set, all the rows of one matrix are joined head-to-tail to create a vector.

The resulting vectors are fed to the classifier, an artificial neural network. The classification process consists of two phases: a training phase and a detection phase. During the first phase, the artificial neural network is trained using a back-propagation algorithm (back-propagation neural networks). In the second phase, the vectors are fed to the trained back-propagation neural networks to realize classification. The authors use 1189 benign Google Play applications and 1227 dangerous apps from 49 malware families proposed by Zhou and Jiang [21] as experimental data. The experiments show that a back-propagation neural network on Markov chains from system call sequences achieves an F-score as high as 0.982773.

To categorize Android malware families, Alejandro Martin et al. offer a tool called CANDYMAN [22]. The states that are experienced when running a malware sample are modelled by combining dynamic analysis and Markov

chains. Each malicious sample is represented as a feature vector and trained using a variety of classification algorithms, including decision trees, random forests, K-nearest neighbours, batch classifiers, support vector machines with linear, rbf, and sigmoid kernels, and a variety of oversampling, under sampling, and hybrid methods from unbalanced learning techniques as well as deep learning algorithms. The classification ability of the method is tested by running several experiments using the DREBIN dataset, from which sample of volume 4442 was extracted, grouped into 24 different malware families: Adrd, FakeRun, Jifake, BaseBridge, Gappusin, Kmin, Boxer, Geinimi, MobileTx, DroidDream, GinMaster, Opfake, DroidKungFu, Glodream, Plankton, ExploitLinuxLotoor, Hamob, SMSreg, FakeDoc, Iconosys, SendPay, FakeInstaller, Imlog, and Yzhc. The accuracy of 81.8% is revealed by experimental findings.

The paper [37] introduces MaMaDroid, a malware detection system for Android that employs a Markov chain behavioural model to classify a series of abstracted API calls performed by an application. Depending on the mode of operation chosen, MaMaDroid constructs a Markov chain for each application, using the series of abstracted API calls as input (packages or families). Each package/family represents a state, and transitions reflect the likelihood of changing from one state to another. Each application's feature vector is made up of the transition probabilities between states in a Markov chain. By applying Principal component analysis to the feature set and choosing the principal components, the classification accuracy is increased.

Different classification methods, which make use of the previously acquired feature vectors, are utilized in the last phase of the classification process, including Random Forests, 1-Nearest Neighbor, 3-Nearest Neighbor, and Support Vector Machines. For the experimental evaluation of MaMaDroid, two sets of benign data are used: the first, which consists of 5,879 apps collected by the PlayDrone, and the second, obtained by downloading the top 100 apps from each of the 29 Google Play Store categories using the googleplayapi tool, as well as malware samples, such as DREBIN testing software and apps that have been uploaded to the VirusShare website. By developing a behavioural model of an Android application, the authors demonstrate through experiments that MaMaDroid is resistant to some techniques used by malware authors to effectively avoid detection. This malware detection system not only effectively detects malware (with up to 99% F-measure) but can also detect new malware families that even appear years after the system was trained.

Windows API call sequence patterns are the focus of Jinsoo Hwang and colleagues' work [16] as they construct a two-stage mixed ransomware detection methodology. To control the false-positive and false-negative error rates, they

first apply a Markov model to capture the characteristics of ransomware. Then, they develop a Random Forest machine learning model using the remaining data. The total accuracy of the suggested approach is 97.3%.

In [18], a method for analysing, detecting, and predicting Windows malware based on the dynamic sequence of API calls is proposed. The relationship between API functions that represent malicious and good software in sequences of API calls is modelled using Markov chain sequences. The proposed system has three phases: initialization, learning, and testing phase. The initialization phase is implemented in three steps: word embedding (a form of representation of words in an n-dimensional space), API similarity calculation (using the model.similarity method), and cluster similarity matrix (using the k-means algorithm).

In the training phase, behavioural models for malicious and good software are created. As a result of execution, two outputs are obtained: the good/malware cluster transition matrix and the good/malware transition model. In this phase, the transition sequences are described using a first-order Markov chain. During the test phase, the accuracy of the model in identifying and classifying unseen sequences into their respective categories is measured. The validation of the proposed model is realized with the following bases: [23, 24], CSDMC2010 Dataset, `https://github.com`. Experiments show an average detection accuracy of 0.990, a false positive rate of 0.010, and an average prediction accuracy of 0.997.

In 2020, Amer, El-Sappagh, and Hu [19] proposed a malware detection mechanism relying on context-awareness between APIs within a call sequence that also detects mimicking malware call sequences. In the initialization phase, the contextually related API functions are founded by extracting the context patterns from huge sequences of malware API calls, and word embedding is applied. The similarity calculation gives two outputs: an API similarity matrix for good software and an API similarity matrix for malware. Finally, in this step, a k-means algorithm is used to implement the clustering of the malware/good similarity matrix.

In the training phase, first-order Markov chain is used to model transition sequences. In the test phase, the model uses the maximum cumulative likelihood of transition probabilities to assess whether or not a sequence is malicious. The performance of the model against the size of the data, which was extracted from a variety of sequences of API calls from [24–26], was investigated. Experimentally, it is founded to have average malware detection accuracy of 0.990, a false positive rate of 0.010, and average malware mimic detection accuracy of 0.993.

5.2 Hidden Markov models for malware detection

It is possible to model a series of events or observations using a HMM, a form of Markov model. A hidden state exists in an HMM, unlike in a standard Markov model, which can only be inferred from the data that has been observed. Complex systems, involving probability and uncertainty, are frequently modelled using an HMM.

In recent years, HMMs have been used to detect malware in a variety of ways. For example, HMMs have been used to detect malicious activity in computer networks by analysing network traffic and looking for patterns that indicate malicious behaviour. It has also been used to detect malicious software, such as worms, viruses, and Trojan horses, by analysing system events and identifying patterns that indicate malicious activity. Finally, it has been used to identify malicious actors by analysing network traffic and looking for patterns that indicate malicious intent.

The authors of the article [27] present a dynamic XDroid approach based on HMM for examining the behaviour of Android applications as they run. The suggested method leverages time, ad libraries, API calls, and private permission requests as inputs to construct an HMM. The authors initially used the DroidCat tool they developed to instrument applications to record behaviour logs. Captured behaviours are synthesized and organized using a filtering and analysing method. The final step is to train and test the HMM model with the DREBIN malicious app dataset and normal apps. In the proposed method, the HMM parameter sets for each application are updated using an online HMM training mechanism. This enables the hidden Markov model to examine behaviour sequences and give users information about the risk levels associated with resource access. A resource access risk level is a quantitative assessment of how likely it is that accessing a resource from an application will cause harm to users. The accuracy of the model was investigated with respect to risk levels, and it was found that at a risk threshold of 0.7, the risk assessment system achieved 88% accuracy for the training set and 80% accuracy for the test set. Experimentally, the authors show that 0.7 is an ideal risk threshold, that a training dataset size of 800 (400 malicious apps and 400 benign apps) is sufficient, and that the risk level is an effective criterion for separating malicious apps from normal applications.

A HMM-based defensive solution against parameter injection attacks in Internet of Things systems, called OwlEye, was presented by Yong and colleagues in 2019 [28]. The system is consisting of four HMM-trained detection modules: a normality detection module, an anomaly detection module, a WAF detection

module, and a custom-specific API detection module. It also has a mechanism that allows web attack detection to be more accurately detected by automatically learning from attack records. In this system, the HMM models contain two hidden states that, respectively, represent malicious and benign requests, while the observable states are sequences of key-value pairs that have been taken from HTTP requests. To evaluate OwlEye, the authors use two sets of data: the first set is used for general discovery purposes, and the second is used for specific API requests. This system is able to detect a wide variety of malicious behaviour, including viruses, worms, Trojans, rootkits, ransomware, and phishing attacks. The conducted experiments show that the recall for detecting SQL (Structured query language) Injection and Cross-Site Scripting (XSS) attacks in the first data set reaches 99.9% and 80.0%, and in the second set, the recall for detecting XSS attacks reaches 98.6%.

The HMM is effective in detecting malware using sequences of API calls, but if minimal data-stealing code is inserted into a large set of legitimate instructions, it is not effective. To solve this problem, Suaboot and his colleagues propose an approach called Sub-curve HMM [30], which focuses on finding subsets of matching patterns rather than the average probabilities of the entire sequence. The proposed approach observes sequence subsets from the API INS and searches for high matching probabilities with the training sequences that are converted into curves. A change in the slope of the curve is used to detect breaks in the series of probabilistic outcomes, called a sub-curve. These curves are used as features by classifiers to detect fragments of suspicious malware behaviour. The architecture of the approach consists of components for feature extraction in three steps in the selection of discriminating feature vectors: API feature extraction, HMM training and subcurve extraction, and classification. The study focused on Windows-based malware and used malware binary samples from VirusShare's database over a 12-month period (`https://virusshare.com/`). Six malware families are selected for the experiment: Keylogger, Zeus, Ransom, Ramnit, Hivecoin, and Lokibot. The experimental results show that for the Sub-Curve HMM approach, the detection accuracy is over 94%.

Using profile HMMs and encoded patterns, Sasidharan and Thomas proposed the ProDroid framework for Android malware detection and classification [30]. Initially, the data from the DREBIN database is processed with the IDA Pro disassembler, and the resulting files are analyzed to identify the API classes and methods used in the Android executables. A list of suspicious Android API classes is generated, which is grouped into different categories based on the suspicious Android API types. These groups make it possible to represent the sequence of API class in FASTA format [38]. In the second phase,

the Multiple Sequence Alignment file is generated from the FASTA file with the MUSCLE computer program to identify a common subsequence of similar malware groups within a family. These files are used in the third phase to train the Profile Hidden Markov Model and generate the profile for each family. The authors use the HMMER software package [39] to generate the profile HMM. Experimentally, the authors show that ProDroid provides 94.5% accuracy in detecting malicious applications with a 7% false-positive rate.

6 Conclusion

A Markov model is a probabilistic approach favoured by researchers because it is able to detect patterns in data that may be indicative of malicious activity and take into account the frequency of certain activities, helping it identify suspicious behaviour. It can be successfully used to detect malware and malicious actors by analysing system events and network traffic. But on the other hand, it is limited by its assumptions and may not be able to detect more sophisticated malware, may produce false positives, and may require a lot of processing power. Having a hidden state in HMM, can be used to detect more sophisticated malware. Compared to the Markov model, the model is more accurate, results in fewer false positives, and uses less computer power. However, if the training data is insufficient or imprecise, it might be less accurate than a Markov model.

Although researchers are still interested in Markov/HMM models for building malware/IDS reporting systems, the increasing complexity of malware and the difficulty in designing efficient Markov/HMM-based models to detect such threats may shift researchers' focus to other detection methods and techniques, such as artificial intelligence.

References

[1] C. R. Manjunath, G. Sudhamsu, H. K. Shashikala, "Review on Types of Computer Viruses and Their Impacts", *Journal of Emerging Technologies and Innovative Research*, 6(1), 2019, pp. 204-209.

[2] J. Kaur, "Taxonomy of Malware: Virus, Worms and Trojan", *International Journal of Research and Analytical Reviews*, 6(1), 2019, pp. 192-196.

[3] S. Hansman, R. Hunt, "A taxonomy of network and computer attacks", *Computers and Security*, 24(1), 2005, pp. 31-43.

[4] A. R. A. Grégio, V. M. Afonso, D. S. F. Filho, P. L. de Geus, M. Jino, "Toward a Taxonomy of Malware Behaviors", *The Computer Journal*, 58(10), 2015, pp. 2758-2777.

[5] N. Pitropakis, E. Panaousis, T. Giannetsos, E. Anastasiadis, G. Loukas, "A taxonomy and survey of attacks against machine learning", *Computer Science Review*, 34, 2019, Art. 100199.

[6] A. Petrosyan, Number of malware attacks per year 2015 – H1 2022, Aug 3, 2022, *Statista*, [23/03/2023].

[7] A. Abusitta, M. Q. Li, B. C. M. Fung, "Malware classification and composition analysis: A survey of recent developments", *Journal of Information Security and Applications*, 59, 2021, Art. 102828.

[8] A. Damodaran, F. Di Troia, C. A. Visaggio, T. H. Austin, M. Stamp, "A comparison of static, dynamic, and hybrid analysis for malware detection", *Journal of Computer Virology and Hacking Techniques*, 13, 2017, pp. 1-12.

[9] S. Talukder, Z. Talukder, "A survey on malware detection and analysis tools", *International Journal of Network Security and Its Applications*, 12(2), 2020, pp. 37-57.

[10] S. Alqurashi, O. Batarfi, "A Comparison of Malware Detection Techniques Based on Hidden Markov Model", *Journal of Information Security*, 7(3), 2016, pp. 215-223.

[11] S. A. Roseline, S. Geetha, "A comprehensive survey of tools and techniques mitigating computer and mobile malware attacks", *Computers and Electrical Engineering*, 92, 2021, Art. 107143.

[12] C. Annachhatre, T. H. Austin, M. Stamp, "Hidden Markov models for malware classification", *Journal of Computer Virology and Hacking Techniques*, 11(2), 2015, pp. 59-73.

[13] Y. T. Ling , N. F. M. Sani, "Short Review on Metamorphic Malware Detection in Hidden Markov Models", *International Journal of Advanced Research in Computer Science and Software Engineering*, 7(2), 2017, pp. 62-69.

[14] A. A. Ramaki, A. Rasoolzadegan, A. J. Jafari, "A systematic review on intrusion detection based on the Hidden Markov Model", *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 11(3), 2018, pp. 111-134.

[15] B. Kitchenham, "Procedures for Performing Systematic Reviews", Keele University, Keele, 33, 2004, pp. 1-26.

[16] J. Hwang, J. Kim, S. Lee, K. Kim, "Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques", *Wireless Personal Communications*, 112, 2020, pp. 2597-2609.

[17] J. Suaboot, Z. Tari, A. Mahmood, A. Y. Zomaya, W. Li, "Sub-curve HMM: A malware detection approach based on partial analysis of API call sequences", *Computers and Security*, 92, 2020, Art. 101773.

[18] E. Amer, I. Zelinka, "A dynamic Windows malware detection and prediction method based on contextual understanding of API call sequence", *Computers and Security*, 92, 2020, Art. 101760.

[19] E. Amer, S. El-Sappagh, J. W. Hu, "Contextual Identification of Windows Malware through Semantic Interpretation of API Call Sequence", *Applied Sciences*, 10(21), 2020, Art. 7673.

[20] X. Xiao, Z. Wang, Q. Li, S. Xia, Y. Jiang, "Back-propagation neural network on Markov chains from system call sequences: a new approach for detecting Android malware with system call sequences", *IET Information Security*, 11(1), 2017, pp. 8-15.

[21] Y. Zhou, X. Jiang, "Dissecting Android Malware: Characterization and Evolution", *2012 IEEE Symposium on Security and Privacy*, San Francisco, USA, 2012, pp. 95-109.

[22] A. Martín, V. Rodríguez-Fernández, D. Camacho, "CANDYMAN: Classifying Android malware families by modelling dynamic traces with Markov chains", *Engineering Applications of Artificial Intelligence*, 74, 2018, pp. 121-133.

[23] F. O. Catak, A. F. Yazı, "A Benchmark API Call Dataset for Windows PE Malware Classification", arXiv:1905.01999, 2019.

[24] Y. Ki, E. Kim, H. K. Kim, "A Novel Approach to Detect Malware Based on API Call Sequence Analysis", *International Journal of Distributed Sensor Networks*, 11(6), 2015, Art. 659101.

[25] Intelligence and Security Informatics Data Sets, BWorld Robot Control Software, `https://www.azsecure-data.org/`, [23/03/2023].

[26] C. W. Kim, "NtMalDetect: A Machine Learning Approach to Malware Detection Using Native API System Calls", arXiv:1802.05412, 2018.

[27] B. Rashidi, C. Fung, E. Bertino, "Android resource usage risk assessment using hidden Markov model and online learning", *Computers and Security*, 65, 2017, pp. 90-107.

[28] B. Yong, X. Liu, Y. Qingchen, L. Huang, Q. Zhou, "Malicious Web traffic detection for Internet of Things environments", *Computers and Electrical Engineering*, 77, 2019, pp. 260-272.

[29] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, K. Rieck, "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket", *Annual Symposium on Network and Distributed System Security (NDSS)*, 2014.

[30] S. K. Sasidharan, C. Thomas, "ProDroid – An Android malware detection framework based on profile hidden Markov model", *Pervasive and Mobile Computing*, 72, 2021, Art. 101336.

[31] A. A. Markov, "Rasprostranenie zakona bolshih chisel na velichin, zavisjashie drug ot druga", *Izvestija Fiziko-matematicheskogo obshestva pri Kazanskom universitete*, 15(2), 1906, pp. 135–156.

[32] L. E. Baum, T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains", *The Annals of Mathematical Statistics*, 37(6), 1966, pp. 1554-1563.

[33] M. Stamp, "Chapter 2: A Revealing Introduction to Hidden Markov Models", *Introduction to Machine Learning with Applications in Information Security*, Chapman and Hall/CRC, New York, 2017.

[34] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, D. Haussler, "Hidden Markov Models in Computational Biology: Applications to Protein Modeling", *Journal of Molecular Biology*, 235(5), 1994, pp. 1501-1531.

[35] S. Attaluri, S. McGhee, M. Stamp, "Profile hidden Markov models and metamorphic virus detection", *Journal in Computer Virology*, 5, 2009, pp. 151-169.

[36] M. Ali, M. Hamid, J. Jasser, J. Lerman, S. Shetty, F. Di Troia, "Profile Hidden Markov Model Malware Detection and API Call Obfuscation", *8th International Conference on Information Systems Security and Privacy – Volume 1: ForSE*, Online Streaming, 2022, pp. 688-695.

[37] L. Onwuzurike, E. Mariconti, P. Andriotis, E. De Cristofaro, G. Ross, G. Stringhini, "MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Models", *ACM Transactions on Privacy and Security*, 22(2), 2019, Art. 14.

[38] W. R. Pearson, D. J. Lipman, "Improved tools for biological sequence comparison", *Proceedings of the National Academy of Sciences*, 85(8), 1988, pp. 2444-2448.

[39] HMMER: biosequence analysis using profile hidden Markov models, `http://hmmer.org/`, [23/03/2023].