Survivability Using Artificial Intelligence Assisted Cyber Risk Warning

Nikolaos Doukas, Peter Stavroulakis, Vyacheslav Kharchenko, Nikolaos Bardis, Dimitrios Irakleous, Oleg Ivanchenko, and Olga Morozova

Abstract The dependence of everyday human endeavours to information systems of different sorts is continuously increasing, simultaneously as important activities such as work and healthcare are evolving so as to exploit the capabilities of computers and networks. At the same time, malicious cyber activities are becoming ever more often and more destructive as criminals also exploit technological progress. In this context, the necessity for system survivability is becoming more important than expecting that computer system security will avert all possible attacks. Artificial intelligence is a technology that is achieving maturity and contributing in a variety of applications. This chapter presents approaches for applying artificial technology schemes in order to promote survivability by detecting evidence of cyber attacks. This chapter presents three recently proposed schemes that detect such behavior in different contexts. The first scheme aims at the detection of threats within data from emails, programs and network traffic. The second scheme pursues the detection of unexpected system behavior by using a clone of the operational system. The third scheme is focusing on the use of redundant resources, such as those encountered in cloud computing schemes, and on the events following a cyber-attack that is already partially successful and affecting the pooled computing resources. The scheme can be used as a toolkit for preventing the negative effects of computer virus cyber-attacks and ensuring high availability for cloud pooled resources.

N. Doukas · N. Bardis · D. Irakleous
Hellenic Army Academy, Athens, Greece
e-mail: nd@ieee.org; bardis@ieee.org; diracleous@sse.gr

P. Stavroulakis (✉)
Technical University of Crete, Chania, Greece
e-mail: pete_tsi@yahoo.gr

V. Kharchenko · O. Morozova
National Aerospace University "KhAI", Kharkiv, Ukraine
e-mail: v.kharchenko@csn.khai.edu; o.morozova@khai.edu

O. Ivanchenko
National Technical University "Dnipro Polytechnic", Dnipro, Ukraine
e-mail: vmsu12@gmail.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

M. Stamp et al. (eds.), *Artificial Intelligence for Cybersecurity*, Advances in Information Security 54, https://doi.org/10.1007/978-3-030-97087-1_12

1 Introduction

Economic, cultural and virtually all sectors of everyday life are over a period of several decades now becoming increasingly dependent on information systems and the electronic services provided via them, while critical fields like medicine and defense are following the same trend. With cyber-crime and other forms of malicious cyber activity continuing to rise, information systems need to display increased survivability i.e., the capacity to continue operating despite attempted or partially successful cyber-attacks. Artificial intelligence-based techniques are therefore being sought that are capable of providing early warnings about intrusions to information systems.

During the last two years, a large proportion of the population saw their ability to work becoming dependent to the existence of internet connectivity and the health of computers. The measures taken to prevent the spreading of COVID19 caused a dramatic increase in the use of computer systems for working remotely while services involving critical sectors such as finance, health, commerce etc. are being offered on-line. The already existing requirement for effective cyber-security has become even more crucial and urgent, whilst new cybersecurity threats are emerging and evolving rapidly. System users and operators require trustworthy and adaptive frameworks to ensure security. The use Artificial Intelligence techniques has also been increasing over the years, with the technology exhibiting maturity. Computer information systems are designed to get more and more intelligent, becoming able to perform difficult decision-making tasks and in less time. In military environments mission areas are studied using simulation and object-oriented architectures [1]. They imitate the human cycle of sensing, reasoning, and acting quite satisfactorily that their significance in process automation [2]. Furthermore, machine learning leverages analytical model building to provide more than the expected performance [3]. It is evident that security is a crucial factor for the uninterruptible operations within this highly complex environment [4, 5]. AI algorithms are therefore being employed as a tool to constantly monitor computer information systems and produce warnings of imminent cyber-threats.

With an ever-increasing number of human activities depending on information systems, the notion of survivability of an information system has been proposed [5–7] that describes the ability of such a system to avert aspiring cyber attackers, avoid total collapse and maintain a reduced service level during a successful attack and promptly recover after the attack has been stopped. Cyber-security efforts aim to promote survivability by focusing on the three R’s, namely robustness, response and resilience [8]. Artificial Intelligence (AI) based cyber defend systems follow the same principles and AI techniques have been proposed that pursue increases in survivability by targeting the 3Rs. AI techniques promote robustness by enhancing a system’s ability to maintain expected behavior in the event when it is processing unexpected input by developing self-testing and self-healing software [8]. Such input may arise from errors, random events or malicious activity. In the context of response, AI enables a system to defeat an attack without intervention and simultaneously optimize its response strategy and adjust its aggressiveness based on previous successes [8]. For example, systems exist that create their own honeypots for attackers and their own decoys. Finally, resilience is promoted by AI by enhancing the system's ability to detect threats and anomalies and hence increasing their ability to withstand attacks [8].

Given the complexity of current information systems, as well as of cyber-attacks, that are inherently of a deceptive nature, constant monitoring is required for survivability and the 3R's [8]. Monitoring is used for detecting deviations of the actual system from expected behavior early, in order to trigger the appropriate response. An AI system watching for such deviations needs profound knowledge of the expected behavior. This can be achieved via the use of a clone copy of the deployed system that acts as a control system, operating in a controlled environment and providing benchmarks for the real systems expected behavior [8]. The sensitivity of this type of monitoring can be adjusted to be compatible to the requirements of the application. A formal model for developing the clone system has been presented [9] that is suitable for operation in conjunction with a decision support system for promoting business goals, including cyber resilience.

This chapter presents three recently proposed schemes that detect such behavior in different contexts.

The first scheme aims at the detection of threats within data. Malicious code fragments are a critical risk to progressively more complex military computer systems. Data from emails, programs and network traffic are collected and analyzed to provide the datasets to model the threats and provide the tools to enhance detection algorithms and evaluate existing protection schemata. In this work, open datasets of threats for training and testing AI detection algorithms are used that have been classified to benign and malicious code based on the features extracted. Natural Language Processing algorithms have been used to train the classifiers using a combination of the methods to provide better results. The overall detection rate achieved is 87.76% in the tests and provides the basis for the usage of this methodology and the integration to existing protection schemata.

The second scheme is targeting the facilitation of detection of unexpected system behavior. One approach to this end, is the use of a clone of the original system that is deployed in a controlled environment and its behavior is considered as a benchmark for the expected behavior of the original system. AI algorithms are trained via adversarial exercises and simulated attacks to recognize divergence between the two systems and produce relevant cyber risk warnings. The presented technique focuses on the implementation of the clone system with emphasis on decision support and the prediction of breakdowns, optimization of service and quality improvement.

The third scheme is focusing on the use of redundant resources, such as those encountered in cloud computing schemes, and on the events following a cyber-attack that is already partially successful and affecting the pooled computing resources. A mathematical model is proposed that considers the impact of the malicious activity on resources and the failures of individual machines. A Semi-Markov approach is used to create a technical subsystem that monitors states in order to solve the problem of analyzing overall availability level, detecting failures and quantifying the impact of the operation of malicious software. The scheme can be used as a toolkit for preventing the negative effects of computer virus cyber-attacks and ensuring high availability for cloud pooled resources.

2 Related Work

An availability model has been presented [10] for an Infrastructure–as–a–Service (IaaS) Cloud with multiple pools of physical machines (PMs). An independent, autonomous mathematical model was proposed that considers abrupt failures of PMs of the pool caused by the impact of deliberate malicious activity, hardware and software failures or other unforeseen interactions with on information resources of the IaaS Cloud. The model is constructed using a stochastic Semi-Markov (SMP) approach. The model employs monitoring of the states of the observed system and its subsystems in order to solve task of determining and analyzing the overall availability level for the IaaS Cloud resource, to the extent that this is affected by the failures and negative impact of the malicious computer viruses and other cyber threats. The study presents the results compared to benchmark steady state availability for the IaaS Cloud, failure rates and repair rates of the PMs that were obtained via observation. For the presented results, overall estimates of availability are obtained, considering the consequences of the activation of two types of malicious computer viruses by using the monolithic SMP model for an IaaS Cloud with three pools of PMs. Therefore, two additional branches of deliberate malicious impacts on PMs resources are required to be implemented by using proposed SMP availability model for an IaaS Cloud.

From the above considerations, it is concluded that the overall effort to promote survivability against cyber-threats, it is necessary to use AI to monitor the content of the emails, data in the databases, scripts, executable code that may contain malicious code. This widespread range of sources of possible cyber-threats should undergo a scanning and cleaning process before being used. Multiple approaches for detection methodologies have been proposed which demonstrates that the problem of data monitoring for potential dangers is a hard problem [8]. The proposed methodologies can be categorized as signature and non-signature based approaches [9]. The contribution of AI techniques is fundamental towards the aim of detecting all types of threats in data. Relevant datasets have been created with annotations if they belong to malicious code [10]. Machine learning algorithms include Supervised, unsupervised, reinforcement methods. The three schemes that were outlined above and are presented in this chapter promote the use of artificial intelligence in order to (i) produce early warning indicators of cyber threats in data, (ii) detect divergent system behavior that could be a sign of ongoing malicious activity within the system and (iii) monitor the availability of pooled cloud resources and the operational state of their individual physical machines during the period that they are suffering from the impact of computer viruses. In all these cases, the final goal is the assurance of increased survivability for systems.

3 Security Infringement Detection

In this section a technique for detecting security infringements in information systems is presented, that is based on a combination of learning techniques. More specifically, the techniques considered are Linear classifiers, Naïve Bayes classifiers, Decision trees with Random Forest technique and Convolutional neural networks employing deep learning.

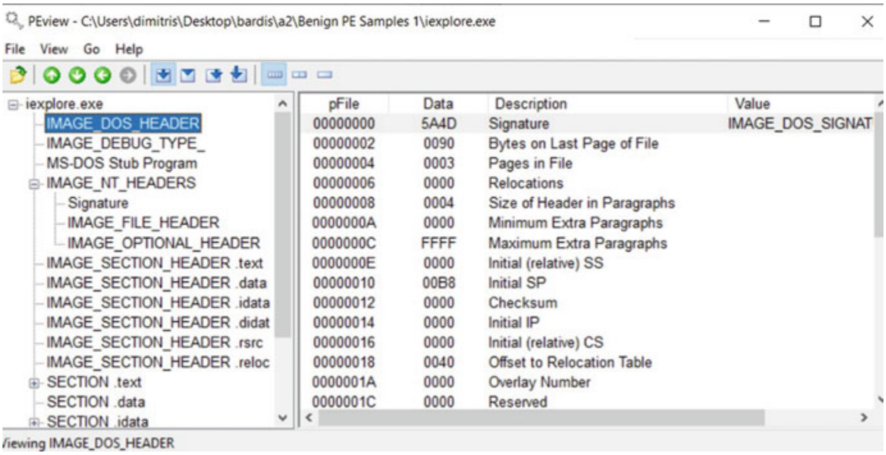
PE Format
From a practical point of view, part of the analysis is based on the examination of Portable Executable (PE) files, a common type of files in the Windows Operating System. They include .exe, .dll, and .sys files. All these files include a PE header, which is a set of instructions to the Windows OS about the analysis of the code that follows. The fields of PE header are usually used as features for the detection of malicious software [11]. Programming libraries in Python can be used to extract the values of PE header.

Many fields in PE files do not follow a strict organization. There exist redundant fields and spaces which can be replaced by malicious code.

3.1 Static Analysis of Code

When static analysis is used, the sample code is tested without being executed. The obtained information may be the PE of the file [12–14] or even more specialized like YARA signatures [15]. In this section, several features will be presented that can be extracted from the executable files via statistical analysis. These features are used in the experiments performed on the dataset to train and the classification algorithms.

It has been proposed in literature [10] that emails be classified using machine learning classifier in a cloud computing system. The security requirements of defense information systems and cloud computing infrastructure have been analyzed and benchmarks of the necessary performance have been determined. Information system users and security software are aware that malware is more likely to be embedded in files of certain types such as executable, shell script etc. Hence attackers are adjusting by masking the true nature of malicious code by hiding it inside normally harmless files or files of unknown type [13]. It is therefore necessary to develop techniques capable of determining the type of file given byte sequences correspond to, without depending on the standard identification criteria normally used by operating systems e.g. file extensions, file headers etc. This part of the study was focused on scanning files in order to definitively determine the type of the file by examining and recognizing the nature of its contents. A database of files was created for this purpose by scraping various internet sources such as GitHub and malware repositories, providing current samples of both benign and malicious files [13]. A base-64 encoding was used for binary data, when this was not already present. Script


PEview - C:\Users\dimitris\Desktop\bardis\a2\Benign PE Samples 1\iexplore.exe
File
View
Go
Help

```
iexplore.exe
  IMAGE_DOS_HEADER
  IMAGE_DEBUG_TYPE_
  MS-DOS Stub Program
  IMAGE_NT_HEADERS
    Signature
    IMAGE_FILE_HEADER
    IMAGE_OPTIONAL_HEADER
  IMAGE_SECTION_HEADER .text
  IMAGE_SECTION_HEADER .data
  IMAGE_SECTION_HEADER .idata
  IMAGE_SECTION_HEADER .didat
  IMAGE_SECTION_HEADER .rsrc
  IMAGE_SECTION_HEADER .reloc
  SECTION .text
  SECTION .data
  SECTION .idata
```

pFile	Data	Description	Value
00000000	5A4D	Signature	IMAGE_DOS_SIGNAT
00000002	0090	Bytes on Last Page of File	
00000004	0003	Pages in File	
00000006	0000	Relocations	
00000008	0004	Size of Header in Paragraphs	
0000000A	0000	Minimum Extra Paragraphs	
0000000C	FFFF	Maximum Extra Paragraphs	
0000000E	0000	Initial (relative) SS	
00000010	00B8	Initial SP	
00000012	0000	Checksum	
00000014	0000	Initial IP	
00000016	0000	Initial (relative) CS	
00000018	0040	Offset to Relocation Table	
0000001A	0000	Overlay Number	
0000001C	0000	Reserved	

/iewing IMAGE_DOS_HEADER

Fig. 1 Exploring the PE header of a benign executable using a hex editor

files, e.g. Javascript, were processed in plain text format. The obtained files are thoroughly filtered, adapted and selected for the testing environment.

In Fig. 1 the loading process of an executable file is visualized, and the code image is examined.

Various parts of the file can hence be examined, including the special field e_magic in DOS header which contains the MZ character series that corresponds to 0x4D 0x5A sequence, and the special header field PE defined as the IMAGE_NT_HEADERS structure. For the purposes of determining the file type, the fact that for static analysis the file is not loaded in memory is advantageous, since in this way the significant risk of malicious code concealment is avoided.

The malicious data is obtained from public databases which include real data and is available for comparison. This PE dataset contains 425 different samples, 378 benign and 47 malicious [13, 14].

Natural Language Processing (NLP) provides a wide range of techniques in text selection, analysis, and estimation. Since the content of the test and training databases is encoded as character sequences, that may be considered as text, the problem of classifying the file content as benign or malicious may be approached as an NLP problem [13]. It is noted that the data has been already annotated for the benign and malicious code fragments. Thus, supervised learning is used to design the classifiers.

Similarity between files can be measured using the hash functions used in cryptographic applications. The requirement in this case it to calculate quantitative similarity scores for file comparison, via similarity hashing algorithms. It hence becomes possible to detect modifications involving copying, insertions, deletions and tampering of the content. These scores can then by used a distance measure for clustering algorithms. Similarity hashing is more suitable in the current context, since it provides information about the nature and the extent of the intervention to the data, rather than a decision on whether two files are identical [13].

3.2 Methodology

The code sequences of the database are processed by means of static analysis of their respective files that are used for training and applying the following classification techniques.

- Linear classifier with Gaussian calculation of the parameters
- Naïve Bayes classifier
- Decision trees with Random Forest technique to solve regression and classification problem
- Convolutional neural networks employing deep learning for malware detection

The overall process is to create a dictionary of all the test cases. All data are created as arrays of words using suitable delimiters. Duplicate entries are removed resulting a dictionary of unique words. Data is transformed into vectors where the entries are used as input to the algorithms.

The header is analyzed by parsing into items or tokens that may be bytes, strings, or any other combination. A sequence of N such items is called N-Gram and this model has been used in many quantitative studies like computational linguistics, speech recognition, bioinformatics, etc. [16, 17]. The N-Gram model has been successfully used to create Markov chains for statistical prediction and text generation from a corpus [13]. In the context of malware detection, N-Gram counts and assortments are used as a basis of the local statistical analysis of the core data to provide a malicious code identification tool. In this study, data was analyzed in 4-Grams of characters and the 10 most frequent ones in each file were used as features. Once this process is completed, a global set of the most useful N-Grams is determined, since for any database of significant size, it is infeasible to consider the large number of all possible N-Grams. For this purpose, the N-Grams that should be chosen are the ones that present the highest discriminating capabilities i.e. ones that are more often observed in a specific type of data, based on the particular dataset. The resulting sequences are generally quite long scoring highly complicated search space. The above feature selection is utilized to enhance performance and reduce the dimensionality of the problem.

The filtering approach deals with the selection of the samples and the assignment of labels. The dataset is separated into learning and test subsets, created as the N-Grams from data and the feature vectors and normalized. The splitting is such that the proportion of each of the two types of data is equal in the global, the training and the test sets.

The classification algorithms receive as input both the data from the PE headers and the features extracted from the N-Gram analysis. Each item is mapped to a numerical vector x using hashing vectorizer, combined with a Term-Frequency

Times Inverse Document Frequency (TFIDF) transformer [18], the text data is converted into numeric form [18–20]. These phases correspond to the considerations and the processing already described in the current Section.

For this study, instead of optimizing a single classifier, a composite scheme utilizing several classifiers was used. A majority decision rule was then used in order to produce the final classification. The distance between points can be calculated with standard distances, like Euclidean, Manhattan or Chebyshev [21]. The Euclidean distance was used in the experiments in this study. A brief description of the classifiers used will be given in the following paragraphs, with detailed emphasis only on aspects that are different from their standard form described in literature. Further details about these schemes are widely available e.g., [21–24].

For a given classification scheme, decision thresholds and similar decision parameters obtained by training may require post adjustment in order to tune the different false positive/false negative detection probabilities required in a particular application.

Binary Classification Method

For a given item yielding the observations feature vector x, a discrimination function f is given as

$$f(w, x) \in \mathbb{R}$$

where w is a set of parameters selected so as to achieve the best separation of the data. In linear models for classification have the general form

$$f(w, x) = w^T x + w_0$$

which are inadequate in most situations since the classes are not linearly separable. In the multiple classes case, a k-rank discrimination function is used.

Naïve Bayes Algorithm

Data belong to two classes, i.e., benign and malicious. We have

$$P(\text{Benign}|k) = \frac{P(\text{Benign})P(k|\text{Benign})}{P(k)}$$

and

$$P(\text{Malicious}|k) = \frac{P(\text{Malicious})P(k|\text{Malicious})}{P(k)}$$

where k is the vector of N features.

Assuming that the features in k are mutually independent, the algorithm calculates the probabilities for new each sample case and compares the two probability values. The larger value is the winner. In the case where the two values are equal the algorithm cannot provide an answer.

Decision Trees

Using supervised learning a decision tree can be constructed to according to an if-logic. The data are classified using well defined questions like a calculated quantity is over a given value. Pro-pruning procedures can be used like minimum tree depth, the maximum leaf nodes, and the minimum samples for each leaf, are used to minimize the size of the evaluated tree. In general, decision trees offer low generalization.

A particular consideration in the case of the Decision Tree was the imbalance caused in the classification mechanism by the fact that benign files were more amply available than malicious ones. The reasons for this imbalance can be readily comprehended by considering the fact that users tend to promptly delete any file they perceive as a threat, e.g. based on the information from an antivirus program. The imbalance can be corrected using several schemes [21]. One scheme involves using class weights for each class that are inversely proportional to the frequency of this class in the data. Another approach is to restore balance by randomly repeating data samples of the least populated class within the iterations of the training epochs. Alternatively, samples from the most populated class can be discarded, for the purpose of equalizing the effect. All these methods were shown to improve the classification performance.

Random Forest

After the calculation of the N-Gram, the names of the units and their number that belong to the PE hear of each file, while the data that cannot be analyzed are omitted. Using a hashing vectorizer the text data are converted into numerical values. A random forest classifier [22] is used for the training, test, and validation data.

Convolutional Neural Network

The Convolutional Neural Network CNN is constructed following the steps [23, 24].

- Declaration of the required programming libraries
- Create a list of files and locations
- The bytes of the file are stored to an array
- Data are divided to train, validate and test
- Optimizer rate is defined
- The structure of the neural network is created
- The model is constructed

For a given number of epochs and batch size, the data are used to train the CNN.

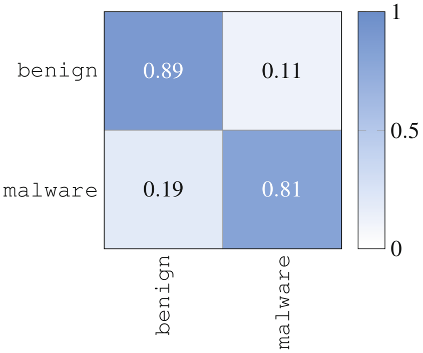
The Aggregated Model

Once trained all models an aggregation of all results takes place. Each method has its own strong aspects, and a soft voting procedure takes place to get the final classification decision. In Table 1 a proposed aggregation classifier is presented.

Table 1 Outline of the composite classification scheme used

Steps	Details
Data preparation	PE headers are being processed
Feature extraction	Dictionaries and hashing
Training base classifiers	Naïve Bayes
	Random Forest
	Convolution Neural Network
Feature identification	Is the pattern recorded in the database as known case
Soft voting engine	Validate the result, find a weighted average, insert in the quarantine, update the database

Fig. 2 Confusion matrix for binary classification



3.3 Results

Applying the combined detection algorithm to the provided dataset the confusion matrix in Fig. 2 is derived. True positive and true negative detection rates are both over 80%. This performance is considered satisfactory as a base for the malicious activity warning system application. With this starting point, the sensitivity of the scheme may be adaptively adjusted for the probability of a false alarm, in order to provide the required compromise between the frequency of alerts and the level of security required. Application of the scheme on the entire dataset produces a correct classification ratio of 87.76%. The efficiency of algorithm varies according to the size of training data.

For the particular case of the Deep Learning Malware Detector algorithm Table 2, the changes in performance with an increasing number of epochs is summarized. The loss function is the objective function that is used to rank and compare the candidate solutions. Epochs are full training cycles used it the training iteration. The results demonstrate that large increases in the number of epochs are not necessary in order to achieve the best possible accuracy of decisions.

Table 2 Deep learning testing accuracies

Epochs	Loss	Accuracy
1	0.3860	0.8604
5	0.3534	0.8906
10	0.3431	0.8813
60	0.3054	0.8996
200	0.4362	0.8709

3.4 Evaluation

The problem of using AI techniques in order to analyze the contents of files for the purpose of determining if potentially malicious code is present was analyzed and formulated as a NLP problem. Existing NLP software tools were used in order to design an end-to-end scheme for processing stored data and producing warning of the potential existence of malware, using machine leaning. The preprocessing require in order to extract the necessary features from the raw data was determined based on particular considerations arising from the nature of the problem. Datasets and data dictionaries were obtained from the performed training of the schemes that can be reused in order to repeat the detection task in other contexts. The datasets may be enriched with additional examples when these are available, in order to improve the classification performance. The framework designed is of particular importance for military applications due to the ability for tuning its sensitivity.

The application of the scheme in specific contexts may be easily tested and benchmarked before deployment to the production environment. This scheme simultaneously illustrates, both in theoretical and practical terms, the feasibility and benefits of using AI classifiers in order to produce indicators of cyber threats and malicious cyber activity. The scheme is essentially used to develop a model for the potential malicious activity. Using this model, it significantly improves security compared to customized rules and datasets. The data and the software tools and libraries used are open source. They can hence take advantage of improvements proposed by the community and continue building knowledge on the appearance of new threats. The composite classification approach used provided, combined with the NLP formulation of the problem provided superior results compared to those expected form applying the individual methods. Numerical data of the benchmark performance of this technique were produced that can be used in future enhancements. The scheme is suitable for demanding applications, such as military environments.

4 Digital Twin Cyber Resilience Decision Support

In the introduction, the notion of how using a duplicate system operating in a controlled environment for the prompt and early detection of cyber-attacks was presented. A formal model supporting the specification, implementation and deployment of digital twin (DT) systems using AI and Internet of Things (IoT) concepts was recently presented [9]. The formal model contributes to the design of DTs capable of detecting diversions from the expected behavior and hence supporting decisions and giving early warnings of cyber threats.

A DT is an exact copy of the actual system under modeling, that exhibits identical dynamic behavior in the environment, but under controlled conditions. The purpose of this system is to provide benchmark or reference behavior for the expected behavior of the actual system so as to enable automated anomaly detection. Ideally, the duplicate system has the same physical structure as the actual one. The overall system concept is illustrated in Fig. 3. However, since the actual system model may include non-replicable entities such as people or behaviors, some of the duplicate subsystems may have to be virtualized. This technology, together with simulation primitives, exhibits a wide variety of prospective applications for the purposes of predicting breakdowns, anomalies and cyber-attacks, optimizing service plans, and optimizing performance. The technology has been demonstrated to cooperate with IoT, AI and Virtual and Augmented Reality subsystems [9]. Thus, DT is a purely digital replica system that exhibits the same behavior as the real-world object, process or system in a controlled environment. DTs can also be considered at subsystem level to formulate components for creating twins of larger systems.

The DT concept arises from the Industry 4.0 movement and has been developed in order to facilitate the following [9].

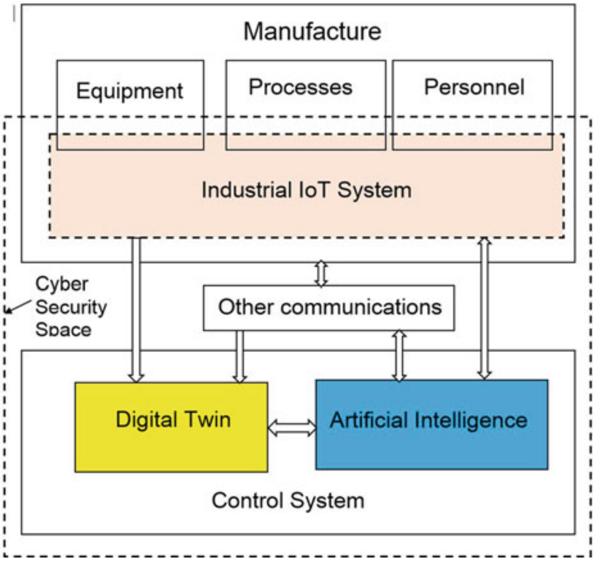


Fig. 3 Overall system concept

- The detection of divergence in behavior due to physical faults, adversarial intervention or erroneous input,
- Planning by means of the prediction of outcomes
- Global system optimization and decision making.

The concept of DT is related to previous concepts of computer aided design and to the notions of online customer profiles, but current DTs involve four significant differences:

- The model reliability with an emphasis on how they support specific performance aims;
- Communication with the real world, for monitoring and control in real time;
- The use of advanced big data analytics and artificial intelligence to open innovative deduction perspectives;
- The capability of evaluating *what-if* scenarios and conducting realistic exercises.

DT systems are combined with AI subsystems in order to promote survivability thereby improving performance and reducing downtime. Machine learning algorithms for manufacturing are shaped and tuned to the specific challenges of systems—such as reducing losses, improving process stability, limiting downtime and detecting anomalies [9].

Applications of AI in currently active operational systems include the predictive maintenance of production computer systems and machines, the use of image processing technologies for the automatic sorting of items such as consumer products, like batteries or food, or user communication using text-based dialog systems e.g., in chatbots. AI supported DT systems are critically dependent on the availability of databases containing real-world or high-quality, artificial performance data. Furthermore, a prerequisite for the successful deployment of AI enabled DT systems is advanced digital maturity of the organization. In physical terms, this implies e.g., the installation of suitable digital sensor systems in many applications along the field of operations. In terms of management know-how, the development of AI applications requires the availability of knowledge of data analysis and/or computer science. However, since this knowledge is not or only to a limited extent available to many organizations, external services are a solution [9]. Modular designs are feasible, in the context of which specific isolated cognitive AI-based services, such as image or face recognition or the conversion of speech into text can be delegated to external entities, like cloud processing resources, in the case where the size of the organization does not permit in-house support for all functionalities.

Additionally, due to the continuously increasing potentials of AI supported applications, the information processing environment paradigms are evolving. Applications involving pattern recognition tend for example to become more autonomous and cost efficient [9]. Simultaneously, AI systems bring improvements in the system's ability to forecast user and environment dependent parameters such as customer actions and wishes, adversarial activities and external interventions. As a result, innovative operational models arise that focus on adapting to the deployment environment, including user behavior. As the AI's acquaintance with the user, the equipment and the environment progresses, deviations from the expected norm become easier to be identified as potential cyber-attacks.

Further amplification of the benefits of the application of AI systems, data stores of limited scope need to be integrated with others in broad range data platforms that promote cross-application collaboration and the creation of digital ecosystems. This necessity is caused by the fact that AI is critically dependent on the volume of training data, including synthetic ones generated from digital models. The training data must include information regarding all possible states of the entities of interest. The quality, the representativeness and the robustness of the training data significantly affect the performance of the system and its overall effectiveness.

Even though AI does not demand the existence of analytical knowledge about the problem to be solved, such as the divergence from the expected behavior caused by the actions of a cyber intruder, data describing all occurrences that may be encountered during normal operations.

Possible benefits from the application of AI include the following [9].

i. Collection of statistics necessary for highlighting disruptions from normal behavior occurring over the entire dataset.
ii. Identification of critical situations encountered at operation time that are not provided for with the predictive analytics;
iii. Processing of large volumes of data produced at operation time and identifying and assessing inconsistencies.

Current research activities on Artificial Intelligence in operations data analysis focuses on the following topics [9].

i. Hierarchical and distributed neural networks-based system with combined relearning
ii. Big Data analytics for multi drone fleets-based monitoring adverse occurrences, such as accidents in remote locations
iii. Deep learning of neural networks for image recognition in space monitoring and manufacturing
iv. Machine vision of autonomous systems
v. Expert systems for logistics based on fuzzy logic
vi. Text recognition using deep learning neural networks
vii. Application of AI for development and implementation of IoT for industry domains.

4.1 Landscape Model Development

The concept of the Landscape has been proposed [9] that is an instrument for representing and analyzing the state of technological development in an entity. Entities may be systems of different sizes, from an information system to a country. It is significant regarding the Industry 4.0 movement which includes modern technologies such as AI, Digital Twins, IIoT etc. The landscape usually consists of a disorganized set of technologies, actors involved, development and implementation. Such a collection cannot serve as a model and be used in any analysis using formal techniques [9]. It has been proposed to describe the landscape in terms of a formal model as

$$LS = \{Reg, Tech, Ent, t, M_{tt}, M_{et}\}$$

where Reg is a region or location or set of regions, $\{Reg_i\}, i = 1, \dots, n$, $Tech$ is a technology or set of technologies $\{Tech_j\}$, $j = 1, \dots, m$, Ent is a technology or set of entities or enterprises $\{Ent_k\}, k = 1, \dots, p$, t is the time or time-slot of interest, M_{TT} is a mapping $M : T \to T$ connecting technologies, and M_{ET} is a mapping $M : E \to T$ describing technologies included in entities.

The model can be described by

- different technologies, locations etc.
- metrics obtained from n, m, p and the cardinalities of the sets $Reg, Tech, Ent$

The set-theoretical model of the landscape can be also represented as a connected graph with weighted nodes and links. More specifically,

- the set of nodes is defined by $Tech$
- the weight of each node is given by the number of entities including the technology
- the weight of each connection is given by the number of unique technologies included by the two entities.

For the case of enterprises, this model allows the evaluation of the most developed technologies and the description of the activities of each enterprise.

A variant of the set-theoretical model of the landscape has been presented [25] that is applicable for the case of Digital Twins. The model is given by

$$DT = \{PE, VM, Ss, DD, CN\}$$

where PE are physical entities, VM are virtual models, Ss are services, DT are data and CN are connections [25]. In this context, the physical entities need to be represented as virtual models in the digital twin, in order to reproduce their behavior. They can be units, systems or systems of systems.

The virtual models are representations of the PEs that maintain their physical and operational properties and present the same behavior for the same events. Additionally, they follow the same rules or logical abilities, such as reasoning, evaluation and decision making.

The data comes at different times, from different sources, are multidimensional and heterogeneous. Some of the data may be actual observations, some may be artificially generated and finally, some may be the product of knowledge of the functionality of the system. Data from multiple sources may be fused according to the needs of the application.

The services cover services offered and services received by the DT. The offered services can be simulation, verification, monitoring, optimization etc., while received services include data services, knowledge banks, computing services etc.

The DTs are connected with their real duplicates to perform complex operations and analyses. Each DT contains six connections

- physical entities to virtual model
- physical entities to data
- physical entities to services
- virtual models to data
- virtual models to services
- services to data

Digital twins designed using such models have been developed for production lines, training personnel, business process optimization, smart cities, construction, healthcare, shipping etc. In the context of cyber resilience, it is proposed that DTs are used for parallel and dynamic monitoring. This concept [8] employs continuous monitoring of the operational system under observation. Due to the diverse and unpredictable nature of possible cyber attacks, the proposed approach for detecting such events is by detecting divergence of the observed system behavior from expected system behavior. This detection needs to be early and prompt in order for the possible attack to be adequately deterred. It is hence proposed that the DT is operated in parallel to the operational system and an AI monitoring system is used to compare the performance of the two systems. The AI system receives input from the data connection and obtains monitoring and telemetry information. Additional training for recognizing divergent behavior is provided to the AI system by executing simulated attacks on the DT. The AI system hence produces security alerts regarding detected divergent behavior. The sensitivity of the alerts is configurable via suitable thresholds of the severity of the attack. Current research involves the additional development of the set-theoretical model so as to further formalize the design and implementation of the DT and its software and hardware components.

5 Semi-Markov Cloud Availability Model

The capability of assessment of the level of survivability achieved following the consequence of cyber-attacks before and after the introduction of the AI survivability promoting schemes is an indispensable tool for the successful development of such schemes.

Cloud Infrastructure–as–a–Service (IaaS) is an extensively used and appreciated cloud computing model with applications to a diverse variety of tasks in different operational environments, manufacturing installations, as well as in the scientific domain. The successful deployment of IaaS Cloud implementations critically depends on the existence of robust solutions to the problem of maintaining availability and guaranteeing cybersecurity for the cloud infrastructure components [10]. Therefore, the challenge of ensuring the availability level of the IaaS Cloud in an environment of diverse cybersecurity threats becomes a particularly significant component of the cybersecurity effort at national level. In order to address this problem, cloud service providers and users of cloud services require techniques capable of determining the effective cybersecurity level for IaaS Cloud, taking into account reliability characteristics of physical machines (PMs) in the process. Such resources typically include different types of servers based on virtual and real physical computer systems components.

Given that cybersecurity and reliability for cloud infrastructure components and availability of the IaaS Cloud are all elements of the overall system survivability, it has been proposed [10] to employ a monolithic Semi-Markov (SMP) model for the purpose of quantifying the overall availability level of the cloud infrastructure. As the global monitoring parameter, the steady state availability was used in order to derive state information [10].

In the research presented in [10] the monolithic SMP model was used for obtaining overall estimates of availability in the context where the effects of two types of malicious computer viruses were required to be studied for an IaaS Cloud instance encompassing three pools of PMs. For this purpose, the proposed SMP availability model for an IaaS Cloud [10] was used in order to implement two branches of deliberate malicious impacts on PMs resources.

Virtually all cloud service providers and users are currently appreciating the necessity to apply significant amounts of effort in order to maintain availability and promote cybersecurity of the IaaS Cloud. A variety of modeling approaches have been employed as instruments for the development of a toolkit necessary for preventing the adverse impact of deliberate malicious activity and ensuring increased availability level of the cloud services [10]. These approaches include models based on [10]

- stochastic non-state-space
- state-space,
- continuous-time Markov chains and
- discrete-time Markov chains (DTMCs).

Other schemes presented in literature employ non-Markovian approaches in order to solve identical tasks in preference to Markov models [10]. Alternative SMP models have also been proposed [10]. The research presented in [10] involves the development of two types of SMP models. The first proposed SMP model is one that can be solved through usage of Embedded Markov Chains for Cloud Systems [10]. The second proposed SMP model is also a model based on embedded DTMCs. The SMP based approach is then employed for the modeling and the determination of the steady-state availability of an IaaS Cloud with three pools of PMs. The modelling includes consideration of sudden failures and deliberate malicious impacts [10].

The model is not bound to a particular cloud architecture, but considers a generic simplified structure for the implementation of the PMs pools belonging to a single IaaS Cloud. According to this structure, an IaaS Cloud consists of hot, warm and cold pools of PMs. Hot pool PMs are powered on and operational. Warm pool PMs are also powered on, but are not active. Finally, cold pool PMs are turned off. Additionally, it assumes that all PMs in a pool belong to the same kind: a hot pool contains only fully operational PMs, a warm pool also includes normal working machines, but these PMs are not ready and a cold pool contains only turn off PMs. A specialized Technical and Information Monitoring System (TIMS) is used to monitor the sequence of states the system is following. Furthermore, TIMS is responsible for performing repair, remove or replacement operations for failed PMs. The entire range of deliberate malicious effects on software and hardware components of IaaS Cloud is also detected by TIMS [10]. The example considered involves the spreading of the WanaCry and Petya ransomware within the system.

The aim of the model is not to provide comprehensive screening and absolute protection from ransomware attacks; the operation of model is based on considering the stochastic process of the spreading of the impact of the malware on the cloud infrastructure. The ultimate purpose for using the Semi-Markov model is to obtain availability estimates of the timeframe for users to observe the effects of the ransomware to the system [10].

The model design is based on the assumption that there exists a relation between the ransomware attack and the reduction of the overall availability level and performance of the IaaS Cloud. Suppose that attack develops in accordance with familiar scenario, namely: first phase, when virus penetrates to physical machine and tries to impact information resource allocated by the PM (WannaCry pattern); second phase, when virus spreads by using of cryptography and ransomware techniques (Petya pattern). In Fig. 4, the finite graph is illustrated of the SMP availability model considering deliberate double insidious malicious impacts on information resources of IaaS Cloud. According to the model description given earlier on, the model also considers that the IaaS Cloud consists of three identical PMs, which are deployed as hot, warm and cold physical machine, respectively [10].

The proposed model additionally contains the TIMS component which, together with additional devices, are responsible for monitoring the system and detecting unauthorized intrusions. Vendors and users of cloud computing platforms are generally incorporating rigorous and effective monitoring systems, that comprise the Monitoring plane [10]. The Monitoring plane provides the functionality necessary to detect multiple instances of unauthorized penetrations employing different points of access. The model proposed can be used as an additional analytical toolkit to develop of anomaly detection technique based on considering different adverse effects, such as sudden failures of PMs and separate deliberate hacker attacks. The SMP availability model considering the adverse effects of deliberate malicious activities on information resources of IaaS Cloud consists of 20 states. Two branches for the activation and evolution of the viruses within the system are modeled. First branch is branch of activation of the WannaCry virus and second branch is the branch of the dispersion of the Petya virus. Table 3 separates the state transitions occurring in Fig. 4 for the case of each of the malwares that cause them in the context of the SMP availability model. The model is considering the impact of malicious activities internal to the information resources of the IaaS Cloud consisting of three

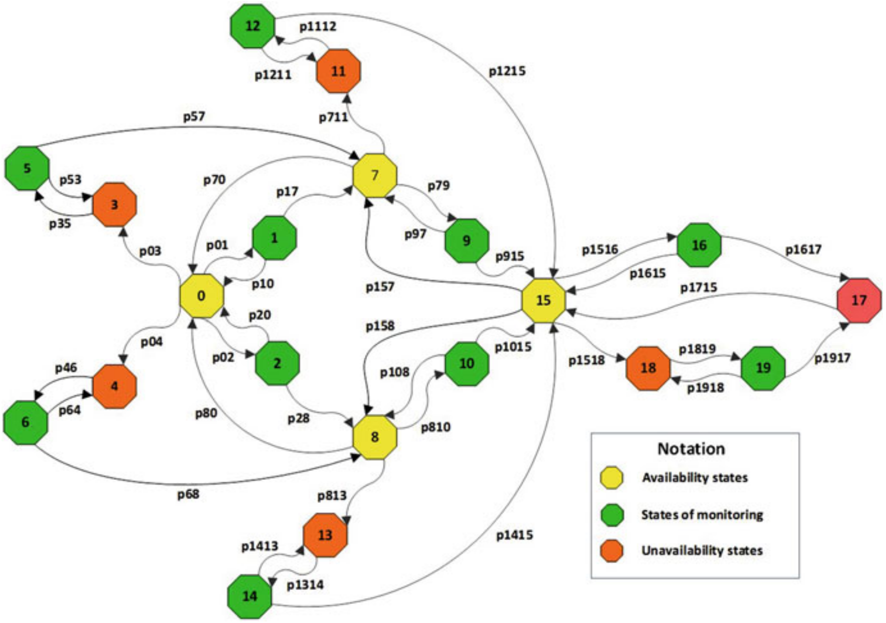


Fig. 4 Finite graph of the SMP availability model for the IaaS Cloud

Table 3 System state transitions caused by the two types of malware [10]

WannaCry virus	Petya ransomware
$0 \xrightarrow{p_{0,3}} 3 \xrightarrow{p_{3,5}} 5$	$0 \xrightarrow{p_{0,4}} 4 \xrightarrow{p_{6,4}} 6 \xrightarrow{p_{6,8}} 8$
$5 \xrightarrow{p_{5,7}} 7$	$7 \xrightarrow{p_{7,11}} 11 \xrightarrow{p_{11,12}} 12 \xrightarrow{p_{12,15}} 15$
	$8 \xrightarrow{p_{8,13}} 13 \xrightarrow{p_{13,14}} 14 \xrightarrow{p_{14,15}} 15$
	$15 \xrightarrow{p_{15,18}} 18 \xrightarrow{p_{18,19}} 19 \xrightarrow{p_{19,17}} 17$

PMs, as well as the impact of external malicious actions. Furthermore, according to previous experience [10] the monitoring system may be used, which this is a really effective means in order to achieve timely detection, but not prevention, of attacks on cloud assets and resources. In Fig. 4, the following conventions are observed for the presentation of the states of the second model: available states are in yellow color, unavailable states are in red color, control and monitoring states for TIMS system are green color.

As illustrated in Fig. 4, if three PMs fail the IaaS Cloud becomes unavailable. Consider the occasion where the system state is unavailable for the IaaS Cloud. Following that, the IaaS Cloud becomes available when the model has been in the states s_0, s_7, s_8, s_{15}. In state s_0, the IaaS Cloud is operational, because all three PMs are available. The opposite situation can only arise when system enters states $s_3, s_4, s_{11}, s_{13}, s_{18}$. These states may be described as states of viruses' attacks, when the system is unavailable due to hidden failures of PMs. The subset of states for TIMS involves all remaining states, namely $s_1, s_2, s_5, s_6, s_9, s_{12}, s_{14}, s_{16}, s_{19}$.

Indeed, the states $s_1, s_2, s_5, s_6, s_9, s_{12}, s_{14}, s_{16}, s_{19}$ are states, when system maintainers will have the ability to exploit the findings of TIMS in order to determine and solve control or monitoring tasks, including measures to implement the defensive features, regarding the viruses' activity.

In order to solve task pertaining to the SMP modeling for cloud infrastructure, it is proposed [10] that the control of the technical and information states the hot PMs perform over the deterministic period of time T, and transitions from states $s_i$to states s_j are given by

$$Q_{0,1}(t) = Q_{012}(t) = Q_{7,9}(t) = Q_{8,10}(t) = Q_{15,16}(t) = \begin{cases} 0 & \text{if } t < T \\ 1 & \text{if } t > T \end{cases}$$

It is also proposed [10] that transitions for the TIMS from state s_i to s_j state occur during period of time τ_c as

$$\begin{aligned} Q_{1,0}(t) &= Q_{2,0}(t) = Q_{5,3}(t) = Q_{6,4}(t) = Q_{9,7}(t) = Q_{10,8}(t) \\ &= Q_{12,11}(t) = Q_{14,13}(t) = Q_{16,15}(t) = Q_{19,18}(t) = \begin{cases} 0 & \text{if } t < \tau_c \\ 1 & \text{if } t > \tau_c \end{cases} \end{aligned}$$

The transitions for branch of activation of the WannaCry virus can be written based on hypoexponential distribution as [10]

$$Q_{10(t)} = Q_{10(t)} = \begin{cases} 1 - \alpha e^{-\lambda_1 t} + \beta e^{-\lambda_2 t} & \text{if } t < T \\ 0 & \text{otherwise} \end{cases}$$

where $a = \frac{\lambda_2}{\lambda_2 - \lambda_1}$, $\beta = \frac{\lambda_1}{\lambda_2 - \lambda_1}$.

Then, the transitions for branch of development of the Petya virus can be written based on hyperexponential distribution as [10]

$$\begin{aligned} Q_{0,4}(t) &= Q_{6,8}(t) = Q_{7,11}(t) = Q_{8,13}(t) \\ &= Q_{12,15}(t) = Q_{14,15}(t) = Q_{15,18}(t) \\ &= Q_{19,17}(t) = \begin{cases} \rho(1 - e^{-\lambda_3 t}) + \omega(1 - e^{-\lambda_4 t}) & \text{if } t < T \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where $\rho \in [0, 1], \omega = 1 - \rho$.

For other states, the exponential and Erlang-k, ($k = 2$) distributions are used in order to describe all times to sudden failures and recoveries of PMs respectively. The cumulative distribution functions for these states are stipulated in Table 4 [10].

Next, if by using the steady-state probability vector, all previous equations and total probability relation $\sum_{i=0}^{19} \pi_i = 1$, the required result is obtained as $A = \pi_0 + \pi_7 + \pi_8 + \pi_{15}$, where $\pi_0, \pi_7, \pi_8, \pi_{15}$ are steady states for states s_0, s_7, s_8, s_{15}.

Table 4 Transitions for failures and recoveries of PMs for the IaaS cloud

Transitions for PMs	CDFs for time transitions
1 → 7 2 → 8 9 → 15 10 → 15 16 → 17	exp(λ_s)
7 → 0 8 → 0	Erlang(2, μ_1)
15 → 7 15 → 8 17 → 15	Erlang(2, μ_2)

Simulation results presented in [10] demonstrate the capability of the SMP to model the expected behavior of the system, where the modelling prediction of the availability matches the observed values and reducing the spreading rate of the viruses, the availability increases.

The modeling results have several theoretical and practical implications [10]. Theoretical perspectives involve the development of Semi-Markov availability models with special states. This type of models may be solved using embedded DTMCs. Practical perspectives relate to the availability assessments of IaaS Cloud and possibility to optimize the architecture and diversification of specific services to be provided. AI techniques are an ideal technique to be employed for this optimization.

Future research could be dedicated to specifying numerical values of parameters for modeling availability assessments of IaaS Cloud with three pools of physical and virtual machines using AI for continuous tracking of such parameters.

6 Future Work

As it was explained in the introduction section of this chapter, cybersecurity is a problem concerning not solely the technical community, but society in general due to its dependence on information systems. AI techniques have been identified as a feasible means of processing large volumes of data for the purpose of identifying threats and divergent behavior. The technology of AI assisted cybersecurity has not yet reached the required level of maturity [26]. There exist several issues that need to be further studied. The success of any AI algorithm is highly dependent on the quantity and quality of the data used for its training.

Current techniques for AI based cybersecurity are primarily based on data originating internally from the organization that they concern. Internal data are naturally closer and more fitted to the organization's internal structure and may allow a quicker learning curve about detecting previous attacks and existing threats. However, the ability to exploit external data offers the prospective for better resilience and hence survivability, by considering the broader trends and developments in the Cyberspace. To this extent, data from GitHub was used in first scheme presented in this chapter, in order to support the recognition of benign files. The fact that a new dataset can be regularly rebuilt in an automatic way, gives the scheme the ability to adapt to emerging threats. Further data sources need to be exploited such SourceForge, search engines, feeds of threat intelligence data by industry and hacker forums that would provide more insight into software like exploit development kits, trends in threat design etc. Additionally, exploitation of data from public or private cyber threat reporting repositories and stores should be considered.

Adversarial Machine Learning (ML) is a growing class of techniques that aim to deceive algorithms by generating data that can pass as rel data. Malicious users are using such techniques for a variety of purposes, including generating AI driven system attacks [26]. Adversarial ML defense focuses on threat modeling, attack simulation, countermeasures, detection and evasion [26]. These schemes attain learning with small datasets and can hence quickly adapt to evolving environments, similarly to human actors. The second scheme presented in this chapter contributes in this direction by providing the infrastructure for realistic attack simulation detection and countermeasure exercises. Due to the dynamic nature of cyberattacks, it is proposed in literature that AI algorithms should not be allowed to take the relevant decisions, but rather support human operators in deciding. The second scheme presented in this paper should be used to become a fundamental part of the code of an AI based cybersecurity decision support system. To the same end, the third scheme presented, involving the SMP used for modeling system availability is suitable for providing such decision support systems with insight into the survivability prospects of cloud systems and the health of each one of their pooled resources.

7 Conclusions

Three techniques were presented, capable of producing indicators of malicious activity within information processing systems that could be associated with the presence of cyber-attacks. The first scheme concerned the detection of threats within data. Executable code, e-mail messages and network packets were processed via AI algorithms in order to model the threats and achieve effective detection of computer viruses and other dangerous content. Open databases and Natural Language processing were employed in order to train different types of classifiers and optimize the results. The second scheme was related to the detection of unexpected system behavior that can be associated to an ongoing cyber-attack. Such behavior can be identified by operating a clone system of the system under observation and using its behavior as a benchmark for the expected behavior for the original system. A technique has been proposed that enables the design and implementation of the clone systems in order to facilitate decision support. AI algorithms can be trained in order to detect diversions of the observed from the expected behavior and produce timely warnings. The final scheme focused on the cloud processing paradigm and was related to the examination of its availability and the detection failures of its constituent subsystems. The analysis was based on a Semi Markov model that enables monitoring of system states, analyzing availability and measuring the impact of the activation of computer viruses within the system. These results contribute to the prevention of the adverse effects of computer viruses and the assurance of high availability of computer systems.

References

1. M. A. Nacar B. Kasım, A. B. Çavdar and E. Çayırcı. Modeling and simulation as a service for joint military space operations simulation. *The Journal of Defense Modeling and Simulation*, 18(1):29–38, 2019.
2. M. J. North and C. M. Macal. *Agent-Based Modeling and Computer Languages*, pages 865–889. Springer Link, 2020.
3. J. van Oijen G. Poppinga M. Hou J. Roessingh, A. Toubman and L. Luotsinen. Machine learning techniques for autonomous agents in military simulations–multum in parvo. In *2017 IEEE International Conference on Systems, Man, and Cyber-netics (SMC)*, pages 3445–3450, October 2017.
4. Y. G. Kim J. Koo and S. H. Lee. Security requirements for cloud-based C4I security architecture. In *2019 International Conference on Platform Technology and Service (PlatCon)*, pages 1–4, January 2019.
5. N. Doukas O. P. Markovskyi P. Stavroulakis M. Kolisnyk V. Kharchenk and N. G. Bardis. *Reliability, Fault Tolerance and Other Critical Components for Survivability in Information Warfare*, volume 990, pages 346–370. Springer, Cham, 2017.
6. Peter Stavroulakis. *Reliability, survivability and quality of large-scale telecommunication systems: case study: Olympic games*. John Wiley and Sons, 2004.
7. Peter Stavroulakis Doukas, Nikolaos and Nikolaos Bardis. *Review of Artificial Intelligence Cyber Threat Assessment Tech-niques for Increased System Survivability.*, pages 207–222. Springer, Cham, 2021.
8. Taddeo M. McCutcheon T. and Floridi L. Trusting artificial intelligence in cybersecurity is a double-edged sword. *Nature Machine Intelligence*, 1(12):557–560, 2019.
9. Vyacheslav Kharchenko, Oleg Illiashenko, Olga Morozova, and Sergii Sokolov. Combination of digital twin and artificial intelligence in manufacturing using industrial IoT. In *2020 IEEE 11th international conference on dependable systems, services and technologies (DESSERT)*, pages 196–201. IEEE, 2020.
10. Oleg Ivanchenko, Vyacheslav Kharchenko, Borys Moroz, Leonid Kabak, and Kyrylo Smoktii. Semi-Markov availability model considering deliberate malicious impacts on an infrastructure-as-a-service cloud. In *2018 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, pages 570–573. IEEE, 2018.
11. Diomidis Spinellis. Reliable identification of bounded-length viruses is np-complete. *IEEE Transactions on Information Theory*, 49(1):280–284, 2003.
12. Yogesh Bharat Parmar. *Windows Portable Executor Malware detection using Deep learning approaches*. PhD thesis, Dublin, National College of Ireland, 2020.

13. Emmanuel Tsukerman. *Machine Learning for Cybersecurity Cookbook: Over 80 recipes on how to implement machine learning algorithms for building security systems using Python*. Packt Publishing Ltd, 2019.
14. S Lee, K Lee, et al. Packed PE file detection for mal ware forensics. *Computer Science and Its Applications*, 2009.
15. David N Palacio, Daniel McCrystal, Kevin Moran, Carlos Bernal-Cárdenas, Denys Poshyvanyk, and Chris Shenefiel. Learning to identify security-related issues using convolutional neural networks. In *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 140–144. IEEE, 2019.
16. Tina Rezaei, Farnoush Manavi, and Ali Hamzeh. A PE header-based method for malware detection using clustering and deep embedding techniques. *Journal of Information Security and Applications*, 60:102876, 2021.
17. Nitin Naik, Paul Jenkins, Roger Cooke, Jonathan Gillett, and Yaochu Jin. Evaluating automatically generated YARA rules and enhancing their effectiveness. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1146–1153. IEEE, 2020.
18. Shahzad Qaiser and Ramsha Ali. Text mining: use of TF-IDF to examine the relevance of words to documents. *International Journal of Computer Applications*, 181(1):25–29, 2018.
19. Nexus. Freeware hex editor. https://mh-nexus.de/en/hxd/. Accessed: 2021-09-30.
20. Mohd Zaki Mas' ud, Shahrin Sahib, Mohd Faizal Abdollah, Siti Rahayu Selamat, and Choo Yun Huoy. A comparative study on feature selection method for n-gram mobile malware detection. *Int. J. Netw. Secur.*, 19(5):727–733, 2017.
21. Abdullah Elen and Emre Avuçlu. Standardized variable distances: A distance-based machine learning method. *Applied Soft Computing*, 98:106855, 2021.
22. Zeinab Khorshidpour, Sattar Hashemi, and Ali Hamzeh. Evaluation of random forest classifier in security domain. *Applied Intelligence*, 47(2):558–569, 2017.
23. Zhiwei Gu, Shah Nazir, Cheng Hong, and Sulaiman Khan. Convolution neural network-based higher accurate intrusion identification system for the network security and communication. *Security and Communication Networks*, 2020, 2020.
24. Iraj Elyasi Komari, Mykola Fedorenko, Vyacheslav Kharchenko, Yevhenia Yehorova, Nikolaos Bardis, and Liudmyla Lutai. The neural modules network with collective relearning for the recognition of diseases: Fault-tolerant structures and reliability assessment. *Neural Networks*, 1:3, 2020.
25. Qinglin Qi, Fei Tao, Tianliang Hu, Nabil Anwer, Ang Liu, Yongli Wei, Lihui Wang, and AYC Nee. Enabling technologies and tools for digital twin. *Journal of Manufacturing Systems*, 2019.
26. Sagar Samtani, Murat Kantarcioglu, and Hsinchun Chen. Trailblazing the artificial intelligence for cybersecurity discipline: a multi-disciplinary research roadmap, 2020.