IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 January 2023, accepted 9 February 2023, date of publication 13 February 2023, date of current version 17 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3244490

RESEARCH ARTICLE

A Methodological Framework for AI-Assisted Security Assessments of Active Directory Environments

GIUSEPPE NEBBIONE AND MARIA CARLA CALZAROSSA, (Senior Member, IEEE)
Department of Electrical, Computer and Biomedical Engineering, University of Pavia, 27100 Pavia, Italy
Corresponding author: Maria Carla Calzarossa (mcc@unipv.it)

This work was supported in part by the University of Pavia through the CRUI-CARE Agreement.

ABSTRACT The pervasiveness of complex technological infrastructures and services coupled with the continuously evolving threat landscape poses new sophisticated security risks. These risks are mostly associated with many diverse vulnerabilities related to software or hardware security flaws, misconfigurations and operational weaknesses. In this scenario, a timely assessment and mitigation of the security risks affecting technological environments are of paramount importance. To cope with these compelling issues, we propose an AI-assisted methodological framework aimed at evaluating whether the target environment is vulnerable or safe. The framework is based on the combined application of graph-based and machine learning techniques. More precisely, the components of the target together with their vulnerabilities are represented by graphs whose analysis identifies the attack paths associated with potential security threats. Machine learning techniques classify these paths and provide the security assessment of the target. The experimental evaluation of the proposed framework was performed on 220 artificially generated Active Directory environments, half of which injected with vulnerabilities. The results of the classification process were generally good. For example, the F1-score obtained by the Random Forest classifier for the assessment of vulnerable networks was equal to 0.91. These results suggest that our approach could be applied for automating the security assessment procedures of complex networked environments.

INDEX TERMS Security assessment, graph theory, machine learning, Active Directory, security threats, vulnerabilities, networked environments.

I. INTRODUCTION

The size and complexity of the technological infrastructures and services being deployed nowadays pose security risks whose assessment is quite challenging. In fact, the layered structure of these environments is frequently characterized by unknown or unexplored dependencies. Similarly, the presence of outdated components and the co-existence of legacy solutions might lead to unpredictable behaviors. Moreover, unexpected events, such as the sudden shift to remote work due to the Covid-19 pandemic, make security risk assessment even more challenging. In fact, the use of remote devices significantly increases the risks and the attack surface of companies and organizations. In this ecosystem, vulnerabilities, due to software or hardware security flaws, misconfigurations and operational weaknesses, often remain undetected, thus allowing their exploitation for different malicious purposes [1], [2].

The number of Common Vulnerability and Exposures (CVE) publicly disclosed is increasing over the years [3]. For example, the number of CVEs disclosed in the first three quarters of 2022, i.e., 18,828, exceeds by about 500 the CVEs of the entire 2020.[1] Moreover, security attacks often involve the theft of personal or critical information, thus drastically increasing the financial and reputation impacts of these incidents. On average a data breach costs over

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

[1] https://nvd.nist.gov/

4 million USD,[2] while the average remediation costs for a ransomware attack are about 2 million USD.[3] To properly cope with this rapidly evolving threat landscape, regular and timely recognition and understanding of potential security risks should become an integral part of all security mechanisms. Risk assessments are generally very demanding since they are often based on time consuming and error prone manual procedures or on automated tools customized to specific infrastructures, such as power grids.

These issues are the main motivation of our work whose primary outcome is a methodological framework that addresses the compelling need of automating security assessment procedures of complex technological environments, such as Microsoft Active Directory (AD). The choice of these environments – that represent the most prolific technology deployed nowadays by enterprises and organizations – is mainly motivated by their complexity that makes them particularly vulnerable. The proposed framework is based on the combined application of graph-based and machine learning techniques. To the best of our knowledge, this is the first framework that combines these techniques for assessing the security of AD environments.

Graphs are particularly suitable to represent the target environment, that is, the individual entities together with their inter-dependencies, vulnerabilities and misconfigurations. In fact, the evaluation of the relationships between entities is more effective than the isolated evaluation of vulnerabilities of a single entity. From the analysis of the properties of these graphs, the attack paths representing the potential security threats affecting the target are identified. These paths are classified by means of machine learning techniques for obtaining the security assessments of the target. The proposed framework is tested on artificially generated Active Directory environments affected by vulnerabilities.

We outline that our framework plays an important role in the security domain in that it allows administrators of AD environments to identify potentially vulnerable attack paths and fix their vulnerabilities or misconfigurations ahead of the attacks, thus reducing the risks of potential disruptions to the technological infrastructures and services.

The main contributions of this work are summarized by the following items:

- Methodological framework for automating the security assessment of Active Directory environments;
- Combined application of graph-based and machine learning techniques;
- Identification of general features characterizing potential security threats of these environments;
- Extensive security assessments of artificially generated Active Directory environments.

The organization of the paper is as follows. Section II reviews the state of the art in the area of security risk assessment. Section III presents the AI-assisted methodological framework proposed for security assessments of Active Directory environments. The setup of the experiments performed to test this framework is covered in Section IV, while Section V focuses on the results of the assessments. Finally, some concluding remarks are presented in Section VI.

[2]https://www.ibm.com/security/data-breach
[3]https://secure2.sophos.com/en-us/content/state-of-ransomware.aspx

II. RELATED WORK

The assessment of security risks affecting technological infrastructures and services has been investigated in the literature under different perspectives (see, e.g., [4], [5] for detailed surveys). This problem is generally very challenging and the solutions are often customized to specific technological environments or tailored to specific security attacks. Some of these solutions exploit graphs, while some others are based on machine learning approaches.

Table 1 presents a comparison of our work with the state of the art. This comparison is based on some relevant parameters referring to the target of the security assessment as well as to the techniques applied and to the types of vulnerability considered. As can be seen, our framework is general and applicable to any type of security attack. In addition, it takes into account both misconfigurations and vulnerabilities, i.e., CVEs, affecting the target network.

In what follows, we present details of the state of the art and we outline our advancements.

A. VULNERABILITY ASSESSMENT

As already pointed out, many security risks are due to vulnerabilities. To reduce or mitigate these risks, several works focus on the development of methodological frameworks for vulnerability assessment of specific technological environments (see e.g., [8], [9], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22]).

In this context, to characterize the security risks in Industrial IoT environments, Figueroa-Lorenzo et al. [8] analyze the security of the main protocols, standards, and buses deployed by these environments and propose a vulnerability analysis methodological framework based on CVSSv3.1. Temporal and environmental metrics are complemented by external factors, such as exposure and threat, with the objective of assessing the impact of vulnerabilities on the three cybersecurity pillars, i.e., confidentiality, integrity and availability. Ten et al. [22] propose an analytical framework that provides a measure to systematically quantify the vulnerabilities of Supervisory Control And Data Acquisition (SCADA) systems. The methodology covers three levels, namely, system, scenarios, and access points.

In the area of cloud computing, Saripalli and Walters [21] devise a quantitative impact and risk assessment methodology where risks are defined as a combination of the probability of a security threat event and its severity. Kamongi et al. [9] offer a vulnerability assessment framework that uses an ontology to create a knowledge base populated with a wide range of vulnerabilities, e.g., Common Vulnerabilities and Exposures (CVE), Common Weakness Enumeration (CWE),

TABLE 1. Summary of the state of the art.

Paper	Target		Technique		Assessment	
	Domain	Attacks	Graphs	ML	CVE	Misconfig
Bi et al. [6]	Smart Grid	Data Injection	✓			
Fellner et al. [7]	Smart Grid	Any		✓		✓
Figueroa-Lorenzo et al. [8]	Industrial IoT	Any			✓	
Kamongi et al. [9]	Cloud	Any			✓	
Kotlaba et al. [10]	Active Directory	Kerberoast		✓		
Tang et al. [11]	Web	SQL Injection		✓		
Wang et al. [12]	Industrial IoT	Any	✓		✓	
This work	**Active Directory**	**Any**	✓	✓	✓	✓

Common Vulnerability Scoring System (CVSS), stored in the National Vulnerability Database (NVD). To obtain a preliminary evaluation of the security level provided by cloud applications, Casola et al. [14] propose a methodology that takes into account the architecture of the applications and their potential security issues, such as threats, attacks, vulnerabilities and weaknesses.

Unlike these works, our framework is general-purpose and can be applied to assess the vulnerabilities of any complex networked environment consisting of diverse devices based on heterogeneous technologies. This also means that our approach can easily cope with the continuously evolving technological landscape.

The problem of IoT-based smart home security risks is investigated in [13]. In particular, this assessment relies on the operationally critical threat, asset, and vulnerability evaluation methodology. Scores are associated with the potential impacts of security risks. Similarly to this work, we define scores for quantifying to what extent the individual components of the target network are vulnerable. Nevertheless, the scope of our approach is not limited to IoT devices.

B. GRAPH-BASED APPROACHES

Approaches based on graphs represent a common solution in security assessment research (see, e.g., [6], [12], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40]). Many papers customize these approaches to the specific technologies, such as smart grids [6], medical devices [29], smart infrastructures [30], web technologies [31], cloud environments [40].

Let us remark that most papers model the components of the target network together with their relationships and vulnerabilities by means of the so called "attack graphs" (see, e.g., [32] for a detailed taxonomy for attack graph generation and usage).

In the framework of network security assessment, an integrated application of attack graph and Hidden Markov models is proposed in [34], whereas Wu et al. [41] focus on an ontology and graph-based approach. In detail, the ontology represents security knowledge concerning, for example, assets, vulnerabilities, attacks, relationships, and the inference rules for identifying possible attacks.

Several metrics have been defined in the literature for assessing security risks. For example, in [27] the evaluation of the relative security levels of various network configurations is based on two metrics, namely, probabilistic security metric and attack resistance metric. In general, these metrics are obtained by exploring the properties of the graphs and identifying the paths that might allow attackers to compromise an individual resource of the network or even the entire network. A common denominator in the definition of the metrics is the CVSS score.[4] In [12] a combination of the CVSS score associated with a CVE, the attack cost and the attack profit is used to characterize Industrial IoT security scenarios. Similarly, in [26] the CVSS score is the basis of a probabilistic metric that estimates the threat of each path of the graph. Gallon and Bascou [25] define damage metrics associated with hosts and networks. These metrics take into account the characteristics and consequences of the attacks constituting an attack scenario.

We outline that the CVSS score captures the principal characteristics of a vulnerability and produces a numerical score that reflects its severity. Nevertheless, this method often fails to take account of misconfigurations that might be abused by attackers whenever these misconfigurations are not classified as vulnerabilities. Our methodological framework copes with this issue and among the characteristics of the graphs it considers both misconfigurations and vulnerabilities affecting the target network.

C. MACHINE LEARNING APPROACHES

Machine Learning techniques are very popular for assessing the security risks of technological infrastructures and services (see, e.g., [42], [43], [44], [45], [46], [47] for detailed surveys of specific technological areas).

For example, Kotlaba et al. [10] focus on the detection of Kerberoasting attacks, a common type of attacks performed in Active Directory environments. The detection of this attack starts from a feature engineering phase based on Microsoft

[4] https://nvd.nist.gov/vuln-metrics/cvss

event logs. Data originating from the logs is analyzed and passed as input to a machine learning classifier able to discern regular noise events from actual Kerberoasting attempts.

Artificial neural networks are adopted in [11] for the detection of SQL injection attacks. In particular, the identification of these attacks is based on a combination of different types of neural networks, i.e., LSTMs and MLPs, whose input is represented by URLs. Another interesting neural network model focusing on the web domain is presented in [48]. This work addresses the detection of a family of XSS vulnerabilities known as DOM XSS. More precisely, a bag of words representation derived from Javascript functions is the input of the deep neural network. A deep learning approach is also applied in [7] to detect misconfigured grid devices using operational data of power distribution grids.

In the framework of Software Defined Networks, Cheng et al. [49] propose a machine learning model for deep packet inspection of encrypted and unencrypted traffic. In particular, a binary logistic regression model is applied for identifying malicious payloads in unencrypted packets, whereas decision trees are applied for encrypted packets.

In the context of machine learning based penetration testing, Valea and Oprişa [50] devise an automated platform to assess the security of a host on a network. In detail, the choice of the exploit to be used on the target host is based on the application of decision trees. This approach focuses on vulnerabilities belonging to a single host, whereas it does not consider vulnerabilities associated with the presence of multiple hosts interconnected in a network. Unlike this work, our approach considers both sources of vulnerabilities and it is not customized to any specific type of attack.

III. METHODOLOGICAL FRAMEWORK

The overall architecture of the methodological framework proposed for the security assessment of complex technological environments, such as Active Directory, is shown in Figure 1. This AI-assisted approach is based on the combined application of graph theory and machine learning techniques. As can be seen, the target, consisting of a large variety of heterogeneous devices, is represented as a graph whose nodes correspond to these devices and whose edges represent their relationships. Nodes and edges are in turn characterized by their properties. Moreover, within the graph multiple attack paths, i.e., sequences of adjacent nodes that are potentially vulnerable, are identified. These paths are described by features and classified to obtain the final assessment of the target as safe or vulnerable. We outline that a network is considered vulnerable whenever security loopholes that might need the attention of network administrators have been identified.

We outline that the nodes of the graphs are colored differently to denote the various types of network entities. We also emphasize that the simple graphs presented in this section are examples aimed at supporting the illustration of the proposed methodological framework and as such they do not fully represent the complexity typically found in the networked environments deployed nowadays.

The proposed methodology is based on the following workflow:

- Data acquisition: dealing with the collection of data about the target network;
- Graph construction: dealing with the encoding of the collected data into a graph;
- Attack paths extraction: dealing with the identification of the sequences of adjacent nodes that represent potential threats;
- Feature engineering: dealing with the identification and selection of the features that characterize the attack paths;
- Classification: dealing with the final assessment of the target network.

Details of the various stages are provided in what follows.

A. DATA ACQUISITION

Data acquisition consists in making the inventory of the entities (e.g., users, groups, computers, printers, routers) belonging to the target network. This inventory, typically built using automated enumeration tools, includes the list of entities together with their properties and physical and logical relationships as well as the vulnerabilities and misconfigurations affecting individual entities. Examples of properties enumerated for network devices refer to the type and version of the operating system, the processor architecture, the firmware version, the open port numbers with the associated services. Similarly, the users being enumerated are described by properties such as personal details, privileges, last login date and time.

Enumeration tools are also useful for extracting the relationships between entities. In particular, these relationships refer to the physical connections between devices and to the logical connections derived from the properties of the various entities. For example, a user with an account on a specific device has a logical relationship with that device. A network printer shared by several computers has a physical relationship with them.

B. GRAPH CONSTRUCTION

Graph construction consists in encoding the enumerated target network into a directed graph model. Graphs are very useful for threat modeling since they highlight non-obvious relationships between network entities. Let us recall that a directed graph G is a pair $(V(G), E(G))$ where $V(G)$ is the set of vertices (or nodes) of the graph and $E(G)$ is the set of edges, with $E(G) \subseteq V(G) \times V(G)$.

In our framework, the nodes of the graph correspond to the network entities previously enumerated, while their physical and logical relationships are represented as edges between nodes. Each node is described by the properties of the network entity it represents and by topological properties such as centrality measures. Moreover, edges between nodes

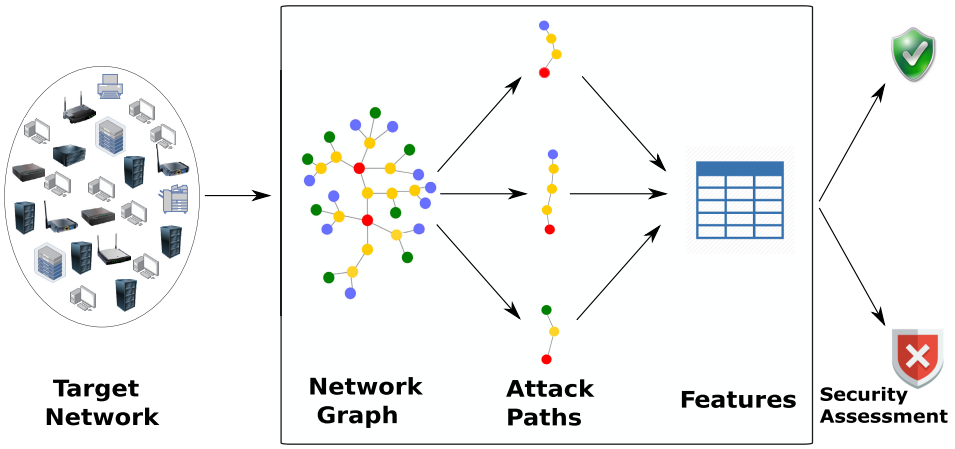


FIGURE 1. Architecture of the proposed methodological framework.

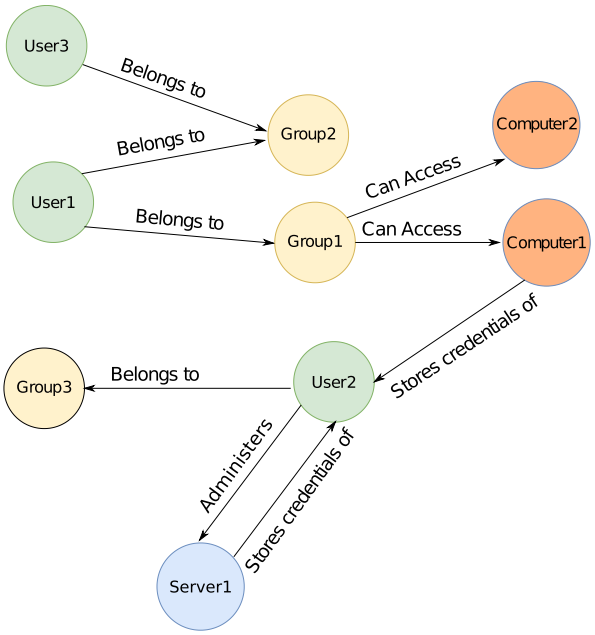


FIGURE 2. Example of a graph modeling a target network.

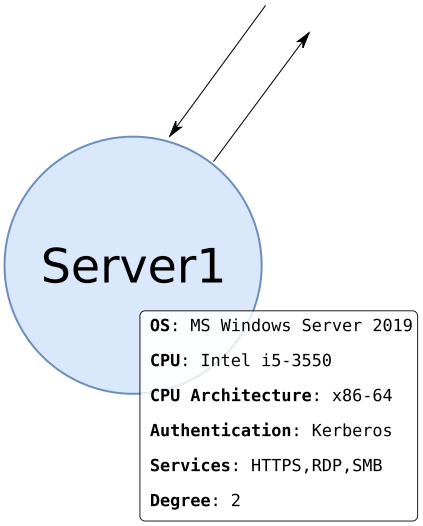


FIGURE 3. Example of a node with its properties.

can be characterized by weights obtained by combining the properties of the nodes and the types of relationships.

Figure 2 shows an example of a graph modeling a target network with nine entities and various types of relationships. The entities refer to users, groups, computers and a server. The relationships between these entities are modeled by the edges. More precisely, the figure shows relationships of users belonging to groups, a group that can access computers, a computer and a server that store user credentials and a user who is the server administrator.

Figure 3 shows a little sample of possible properties associated with the server of Fig. 2. We outline that servers are generally described by many diverse properties related, for example, to their basic characteristics and security services as well as to the services being offered and the relationships with neighbor network entities. The server properties listed in the figure refer to its operating system, i.e., Microsoft Windows Server 2019, its processor architecture, i.e., Intel x86-64, and the supported authentication mechanism, i.e., Kerberos. In addition, the server exposes three services, i.e., `https`, `rdp` and `smb`, and is characterized by one incoming and one outgoing edge, that is, the node has a degree equal to two.

C. ATTACK PATHS EXTRACTION

The analysis of nodes and edges of the graph and of the corresponding properties is the basis for identifying the potential security threats of the target network. This analysis aims at extracting the attack paths, that is, potentially vulnerable paths that could be exploited by attackers to compromise some specific network entities.

Let us recall that a path X on a graph G is a non-empty graph consisting of a sequence of non-repeating adjacent

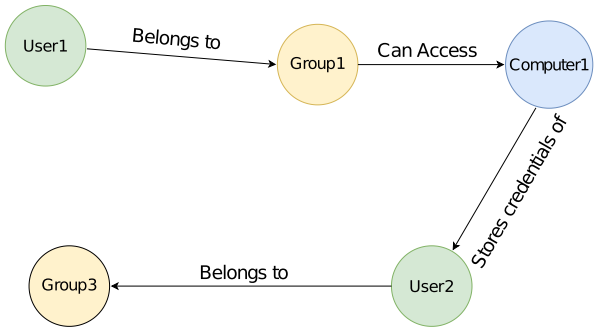


FIGURE 4. Example of attack path from `User1` to `Group3`.

nodes and edges such that $V(X) \subseteq V(G)$ and $E(X) \subseteq E(G)$. The path length corresponds to the number of edges in X.

Given an origin node, e.g., a compromised node, and a destination node, e.g., the target of the attacker, different approaches can be applied to extract attack paths. For example, attack paths could correspond to the shortest paths between the nodes, that is, the minimum number of nodes to be traversed, thus prioritizing how close nodes are. For weighted graphs, attack paths could corresponds to the paths of the minimum weight between the nodes, thus prioritizing specific characteristics of the graph. Attack paths could also be identified by applying heuristics that take advantage of the domain knowledge of the technological environment under investigation. For example, paths could include nodes and edges characterized by specific relationships that make them particularly vulnerable.

Figure 4 shows the path of length four between `User1` and `Group3` extracted from the graph of Figure 2. This attack path is potentially vulnerable because of the relationships between nodes. In fact, the membership of `User1` to `Group1` grants the access to `Computer1`. In addition, `Computer1` stores the credentials of `User2`. Hence, `User1` could retrieve these credentials and impersonate `User2`, thus reaching `Group3` and performing potential privilege escalation. This example has shown that the security risks associated with the extracted paths mainly depend on the properties of the nodes and on their relationships. Of course, the consequences and impacts of this attack depend on the privileges associated with `Group3`.

In what follows we denote the set of attack paths from a given origin node to the destination node as attack graph.

D. FEATURE ENGINEERING

The security assessment of the target network requires the identification and engineering of features describing the properties of the nodes and edges of individual attack paths and of the corresponding attack graph.

More precisely, these features should capture the potential vulnerabilities of nodes and edges within attack paths (e.g., presence and number of network entities running obsolete operating systems, storing passwords in clear-text or enabling remote access).

Other features could be related to the structural properties of individual attack paths (e.g., path length, total weight of the path) or of the corresponding attack graph (e.g., number of attack paths, clustering coefficients, transitivity).

Features significantly affect the classification process and the outcome of the security assessment. Hence, feature engineering, e.g., selection, scaling and aggregation of the identified features, is a crucial task that requires a solid technological and security background.

E. CLASSIFICATION

The classification of the identified attack graphs is the basis of the security assessment of the target network. For this purpose, a classical machine learning approach consisting of three phases, namely, training, validation and testing, is applied. In particular, training deals with learning how to distinguish between vulnerable and safe attack graphs, whereas validation and testing deal with the tuning and evaluation of the classification process.

This process is based on different classification algorithms, such as Logistic Regression, Support Vector Machines, Decision Trees, K-Nearest Neighbors, that differ for their learning strategy and computational complexity [51].

Since the performance of the classification process heavily depends on the identified features, features might be re-engineered multiple times to obtain an accurate security assessment.

IV. EXPERIMENTAL SETUP

This section presents the setup of the experiments performed to test the proposed methodological framework on Active Directory environments. As already pointed out, the complexity of these environments makes them highly vulnerable, thus requiring accurate and timely security assessments [52]. In addition, we believe that Active Directory environments are good representative of complex networked scenarios that might benefit of automated security assessment procedures.

In what follows, we describe the main characteristics of the environments considered in our investigation and discuss the choices made in the various steps of the methodology as well as the corresponding implementation details.

A. GENERATION OF ACTIVE DIRECTORY ENVIRONMENTS

Before presenting the characteristics of the Active Directory environments being tested, we briefly introduce the main components and services offered by these environments.

Active Directory is a set of technologies developed by Microsoft to implement directory services aimed at managing complex computer networks [53], [54]. A directory is a hierarchical structure that stores information about objects on the network. According to Active Directory terminology, objects refer to users, computers, groups, organizational units, services and even to network policies. Moreover, the hierarchical organization of objects includes domains, i.e., collections of objects, and forests, i.e., collections of domains.

Active Directory services offer the methods for storing and retrieving directory data. This data is essential for the proper functioning of the entire network and in particular for many fundamental services, such as authentication and authorization.

The implementation of Active Directory services is based on specialized servers, known as Domain Controllers, often used for the centralized configuration and management of the network.

The methodological framework is tested on artificially generated Active Directory environments since, to the best of our knowledge, no datasets of Active Directory environments are publicly available. In fact, companies and organizations are not willing to disclose any detail about their technological infrastructure and internal organization due to confidentiality and security issues as well as to competitive reasons [44]. In addition, Active Directory environments typically consist of hundreds of network entities characterized by a complex hierarchical organization, thus the setup of technologies representative of realistic scenarios is very expensive and cumbersome.

For the generation of realistic AD environments, we consider the different types of network entities that are typically part of these environments, e.g., users, groups, computers, organizational units, domain controllers, and we describe each entity by properties and relationships, e.g., authentication and delegation mechanisms, membership, access control and trust relationships, group policy management. Some of the properties might refer to various types of misconfigurations inadvertently caused by system administrators, such as incorrect access control list settings.

To promote the diversity of the environments being generated, we associate a probability distribution with each property and relationship. For example, to choose the access control right of a given entity, we assign a probability to each possible right, e.g., `GenericAll`, `AddMember`, `WriteDacl`, and we sample the corresponding distribution. Note that some properties and relationships might lead to the generation of entities affected by vulnerabilities (e.g., zerologon) or misconfigurations (e.g., unconstrained delegation, non-expiring passwords) that might allow attackers to compromise the network. For these reasons, the generated AD environments are particularly suitable for testing our methodological framework.

It is also important to mention that some specific characteristics of the generated AD environments, such as the number of Domain Controllers, are chosen according to heuristics derived from the best practices suggested by Microsoft [55], [56]. For example, for reliability reasons it is recommended to include multiple Domain Controllers in a domain. For customizing the configuration policies, authorizations granted to users should take into account their role and responsibilities. Similarly, users and computers should belong to different organizational units.

TABLE 2. Scores assigned to some of the relationships associated with the edges of the graphs.

Relationship	Score	Severity
Administers	10	HIGH
Belongs to	5	MEDIUM
Can Execute	5	MEDIUM
Can RDP	5	MEDIUM
Has Session	3	LOW

For our experiments we generate in total 220 artificial AD environments. On average an environment consists of eight Domain Controllers managing 103 users and 120 computers subdivided into 59 groups and belonging to 41 organizational units. Moreover, an average of 31 configuration policies are associated with Domain Controllers.

B. ATTACK GRAPH CONSTRUCTION

The generated Active Directory environments are modeled through the use of graphs. Within these graphs, three main categories of relationships have been identified according to their purpose, namely:

- Access relationships describing access capabilities of network entities;
- Authorization relationships describing the permissions of network entities;
- Hierarchical relationships describing the hierarchical organization of the network entities.

Under specific circumstances these relationships might be exploited, thus becoming critical for the security of the network.

To quantify the degree of vulnerability of the network entities, we assign scores to the properties associated with nodes and to their relationships. These assignments require solid knowledge of Active Directory environments as well as deep experience in the security domain. The choice of the values of these scores is driven by the characteristics of the nodes and the potential interest of an attacker towards the specific node, e.g., domain administrator, unprivileged user. In particular, higher scores are associated with a higher degree of vulnerability. In general, scores are customized to the networked environment being tested. Scores might change as a consequence of newly discovered vulnerabilities affecting network entities or existing vulnerabilities being patched.

Not to clutter the presentation, Tables 2 and 3 present some examples of scores for node properties and relationships, respectively. Note that the choice of the values assigned to these scores takes into account our experience in the Active Directory domain. As can be seen, the `Belongs to` relationship, referring to users being members of groups, has a lower score with respect to the `Administers` relationship referring to users with the role of server administrators (see Table 2).

Moreover, the scores assigned to computers running old operating systems, such as `Fedora 16` or `Microsoft Windows XP`, differ significantly because of the different degrees of vulnerability affecting these operating systems

TABLE 3. Scores assigned to some properties associated with the nodes of the graphs.

Entity	Property	Value	Score	Severity
Computer	OS	MS Windows XP	30	CRITICAL
		MS Windows 7	10	HIGH
		MS Windows 10	0	NULL
		Fedora 16	5	MEDIUM
		Ubuntu 22.10	0	NULL
User	Expiring Password	False	3	LOW
		True	0	NULL
Group	Privilege	High	50	CRITICAL
		Medium	30	CRITICAL
		Low	20	HIGH

(see Table 3). On the contrary, computers running modern operating systems, such as `Microsoft Windows 10` or `Ubuntu 22.10`, are less vulnerable because of the regular updates and patches released by the developers. Hence, their score is set to the minimum value, i.e., zero.

The potential vulnerability of the node N_i is summarized in terms of its overall score $v(N_i)$, that is, the sum of the scores assigned to the properties of the node and to the relationships of its outgoing edges.

The overall score is used as the basis for assigning weights to edges. In detail, the non-negative weight $w(e_{ij})$ associated with edge e_{ij} connecting node N_i with node N_j is defined as follows:

$$w(e_{ij}) = \log\left(1 + \frac{1}{v(N_i) + 1}\right)$$

To better identify the shortest paths, the weights are defined as inversely proportional to the overall score. Moreover, the logarithmic transformation is applied to reduce skewness, while one is added to $v(N_i)$ to avoid a division by zero. Note that weights are also interpreted as costs associated with edges.

C. IMPLEMENTATION

The technology used to store and analyze these graphs is a graph-based database platform, namely, Neo4j,[5] that relies on a de-facto standard language for graph querying, i.e., `cypher`.[6] This kind of database is very efficient for storing data related to graphs. In fact, storage and computation requirements are very limited. For example, a node of a graph can be stored in as little as 15B, while an edge requires 34B. Moreover, the encoding of the generated environment into a graph is very fast. In fact, a laptop with a Intel Core i7-6600U CPU running at 2.6GHz with 8GB of RAM, takes less than three minutes to construct a graph consisting of about 400 nodes and 4, 000 relationships.

To extract attack paths from each graph we create simple `cypher` queries where we select as origin node a non-privileged user and as destination nodes privileged users, such as domain administrators. Note that a fast bidirectional breadth-first search algorithm is used if the predicates can be evaluated whilst searching for the path, if not, the slower exhaustive depth-first search algorithm is used.

```
MATCH paths=allShortestPaths(
    (User1)-[*1..]->
    (Group3:Group
        {name: 'DOMAIN ADMINS@B.COM'}))
WHERE NOT User1=Group3
RETURN paths
```

FIGURE 5. `Cypher` **query to compute shortest path.**

An example of a query that computes the shortest paths from `User1` to `Group3` of the domain administrators is reported in Figure 5. These paths refer to the minimum number of nodes from the origin to the destination. A similar query is created to extract the path with the minimum weight identified by using the Dijkstra's algorithm.

As a result of these queries, we identify 220 attack graphs consisting on average of six attack paths with seven nodes each. We recall that an attack graph is the set of all the attack paths extracted from the graph constructed for the generated AD environment (see Fig. 4 for an example). The paths correspond to the nodes to be traversed, starting from a given origin node, to compromise a target destination node, We recall that these graphs are particularly important since they highlight non-obvious chained misconfigurations that could be exploited by attackers.

The average weight associated with the weighted attack paths is rather small, that is equal to 0.13. In fact, according to our formulation, the weights are strictly positive and do not exceed 0.69 corresponding to $log(2)$ and our objective is to find the path of minimum weight, i.e., cost.

Note that each path has been labeled manually as vulnerable or safe depending on whether it could be exploited by an attacker to compromise the network. This labeling process is based on the analysis and visual inspection of the corresponding sub-graphs. In detail, paths are labeled by analyzing the properties associated with the nodes of the paths and by looking at the relationships between nodes. For example, the relationships between the nodes of the attack path shown in Figure 4 make it vulnerable. In fact, by leveraging the credentials of `User2`, `User1` could reach `Group3` and perform potential privilege escalation. It is important to emphasize that the labeling process requires a solid knowledge in the security and Active Directory domains.

D. MACHINE LEARNING CLASSIFIER SETUP

As already pointed out, to discriminate between vulnerable and safe attack graphs, the extracted features should summarize the characteristics of these graphs and capture the potential vulnerabilities of nodes and relationships. In particular, features are associated with the frequency of each type of node and each type of relationship within a path as well as with the frequency of nodes and relationships considered critical from a security perspective.

[5] https://neo4j.com/
[6] https://opencypher.org/

Of these features, 26 refer to the shortest paths and 26 to the weighted paths. In summary, these features belong to the following categories:

- Type of node: 6 features;
- Nodes with critical properties: 6 features;
- Non-Critical Access Relationships: 12 features;
- Non-Critical Authorization Relationships: 14 features;
- Non-Critical Hierarchical Relationships: 8 features;
- Critical Relationships: 6 features.

We also consider features referring to the total number of nodes and relationships in the identified paths, to the number of paths and to the corresponding assessment metric. In total we obtain 60 features.

In what follows, we present examples of the types of nodes and relationships considered to extract the features. Additional details can be found in [57].

Some features refer to the various types of nodes included in the attack graphs, such as users, computers and groups, as well as to the nodes critical from a security point of view, such as domain controllers, privileged users and groups, computers running obsolete operating systems or with unconstrained delegation enabled. In the context of relationships, features are associated with access control relationships, such as `GenericWrite` and `GenericAll`, remote access relationships, such as `CanRDP` and `CanPSRemote`, and critical relationships, such as `AddMember`, `HasSession` and `AdminTo`.

In general, the features related to critical properties and relationships are particularly relevant to differentiate safe and vulnerable attack graphs. For example, the feature related to the number of computers running old operating systems is good for this purpose. In fact, a path that includes a large number of these computers is potentially more vulnerable than a path that does not include any or includes very few.

We outline that the some of the features selected for describing the attack graphs are not specifically customized to AD environments, thus they could be used to characterize the potential security risks of other networked environments.

To reduce the problem dimensionality, we create for each group of relationships a new feature obtained by counting the number of relationships within the group. Moreover, we compute the coefficient of correlation between features and we discard highly correlated ones, that is, features whose coefficient of correlation is above 0.9. As a result, we reduce the 60 extracted features to 12 features only, namely, two features referring to the shortest path, that is:

- Number of computers running old operating systems;
- Overall cost associated with the shortest path;

and ten features referring to the weighted shortest path, that is:

- Number of relationships related to access control mechanisms;
- Number of nodes representing hierarchical grouping;
- Number of relationships related to remote access capabilities;
- Number of relationships considered critical from a security point of view;
- Number of computers;
- Number of computers running old operating systems;
- Number of users;
- Number of administrative sessions;
- Overall score of the weighted shortest path;
- Overall cost associated with the weighted shortest path.

Popular algorithms, i.e., Logistic Regression, Support Vector Machine and Random Forest, are applied for the classification of the attack paths. Note that these classifiers and the corresponding machine learning models are used in these experiments as a proof of concept of the proposed framework. In facts, our main objective is to assess their ability to classify attack graphs as either vulnerable or safe, rather than identifying the best classifier for the data at hand.

The main choices associated with the classification process are summarized as follows:

- 12 features;
- Classical 80/20 split of the dataset;
- k-fold technique with $k = 10$ for the validation of the classifiers;
- Grid search for hyper-parameter tuning.

The implementation of the entire classification process is based on `Python3` and in particular on the `pandas` and `scikit-learn` modules.

TABLE 4. Hyper-parameters used for tuning the Logistic Regression classifier.

Parameter	Range	Best Value
C	$10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^{2}, 10^{3}$	1
Penalty	L1, L2	L1

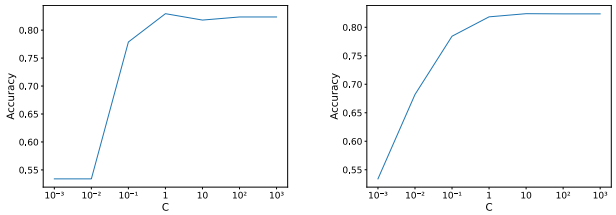
V. EXPERIMENTAL RESULTS

The application of the classification process provides the security assessments of the Active Directory environments described in Section IV-A. Not to clutter the presentation, this section discusses the main results of this process. Additional details are provided in [57]. Let us recall that the dataset used in the experiments is balanced and consists of 110 safe attack graphs and 110 vulnerable attack graphs. Each graph is described by 12 features.

A. HYPER-PARAMETER TUNING

The tuning of hyper-parameters relies on a grid search customized to each classifier. For the Logistic Regression (LR) classifier, the grid search is based on two hyper-parameters, i.e., regularization C and penalty (see Table 4). The accuracy corresponding to the tested hyper-parameters is shown in Figure 6. As can be seen, the results of the validation suggest that the best values of the regularization parameter C and of the penalty are 1 and $L1$, respectively. In fact, the resulting accuracy on the testing dataset is equal to 84.1%.

Similarly, the grid search of the Support Vector Machine (SVM) classifier is based on two hyper-parameters, namely,

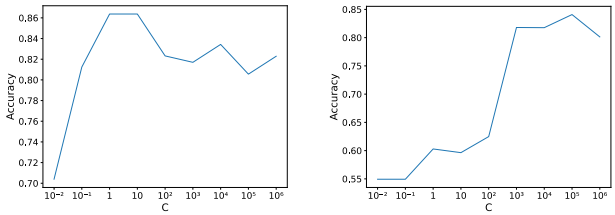


(a) $L1$ Penalty (b) $L2$ Penalty

FIGURE 6. Accuracy of the Logistic Regression classifier on the Cross-Validation dataset.

TABLE 5. Hyper-parameters used for tuning the Support Vector Machine classifier.

Parameter	Range	Best Value
Decision Trees	10, 20, 50, 100, 150, 200, 300	100
Max Tree Depth	None, 2, 3, 4	3



(a) Using $Linear$ Kernel (b) Using RBF Kernel

FIGURE 7. Accuracy of the Support-Vector Machine classifier on the Cross-Validation dataset.

TABLE 6. Hyper-parameters used for tuning the Random Forest classifier.

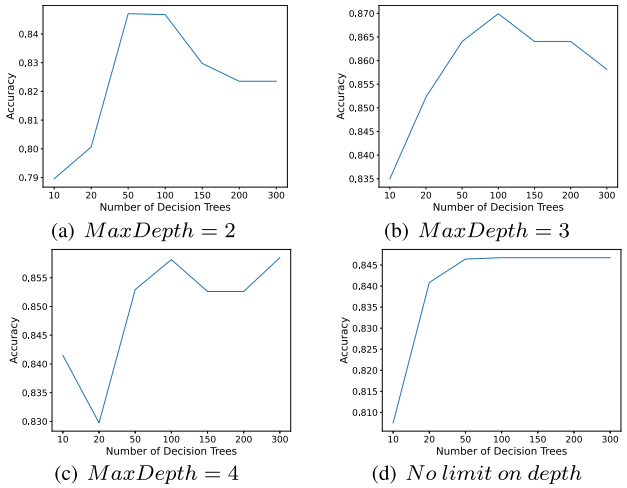
Parameter	Range	Best Value
C	$10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^4, 10^5, 10^6$	1
Kernel	Linear, RBF	Linear

regularization C and kernel type (see Table 5). The accuracy corresponding to these hyper-parameters is shown in Figure 7. The best performance using SVM during validation is obtained by setting $C = 1$ and using a *linear* kernel. The resulting accuracy on the testing dataset is 86.3%. Finally, for the Random Forest (RF) classifier, the grid search is based on two tuning parameters, namely, number of decision trees used within the forest and maximum tree depth (see Table 6). The accuracy corresponding to the tested hyper-parameters is shown in Figure 8. The grid search suggests that the best performance is obtained with a number of decision trees equal to 100 and a maximum tree depth equal to 3. The resulting accuracy reaches 91% on the testing dataset.

B. PERFORMANCE COMPARISON

Table 7 summarizes the performance of the three classifiers applied for the security assessment, that is, to assess whether the generated Active Directory environments are safe or vulnerable. The performance, expressed in terms of precision, recall and F1-score, refers to the testing dataset.

We notice that RF classifier consistently outperforms the other classifiers, although the performance of all classifiers



(a) $MaxDepth = 2$ (b) $MaxDepth = 3$

(c) $MaxDepth = 4$ (d) $No\ limit\ on\ depth$

FIGURE 8. Accuracy of the Random Forest classifier on the Cross-Validation dataset.

TABLE 7. Comparison of the performance of the three classifiers.

Classifier	Label	Precision	Recall	F1
LR	Safe	0.74	0.88	0.80
	Vulnerable	0.92	0.82	0.87
SVM	Safe	0.90	0.82	0.86
	Vulnerable	0.83	0.91	0.87
RF	Safe	0.95	0.86	0.90
	Vulnerable	0.88	0.95	0.91

is generally good. For example, the F1-score ranges between 0.80 obtained by the LR classifier for the assessment of safe networks and 0.91 obtained by the RF classifier for the assessment of vulnerable networks. Similarly, the precision and the recall are always greater or equal to 0.74 and 0.82, respectively.

In summary, these results suggest that the combined application of graph models and machine learning techniques is very useful for automating the security assessment procedures of complex technological environments, such as Active Directory.

VI. CONCLUSION

In this paper we addressed the problem of security assessment of Active Directory environments because their complexity makes them particularly vulnerable. In particular, to evaluate whether a target is vulnerable or safe, we proposed an AI-assisted methodological framework based on the combined application of graph models and machine learning techniques. More precisely, from the graphs describing the target, attack paths representing its potential security threats are extracted. The classification of these paths by means of machine learning techniques provides the security assessment of the target. For the experimental evaluation of the proposed framework we focused on 220 artificially generated Active Directory environments, half of which affected by vulnerabilities. The results of the classification process were generally good. For example, the F1-score obtained by the

Random Forest classifier for the assessment of vulnerable networks was equal to 0.91. These experiments suggested that our framework could be applied for automating the security assessment procedures although it might require an initial human intervention in the definition of the scores associated with nodes and relationships of the target being tested.

Future works will be dedicated to analyze the sensitivity of the proposed approach with respect to the scores assigned to nodes and relationships and to the evolving threat landscape. Moreover, we plan to characterize the attack graphs in terms of additional features and investigate their impact in the security assessment process.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions that improved the overall quality and clarity of the manuscript.

REFERENCES

[1] M. A. Bishop, *The Art and Science of Computer Security*. Reading, MA, USA: Addison-Wesley, 2002.

[2] M. Stamp, *Information Security: Principles and Practice*. Hoboken, NJ, USA: Wiley, 2011.

[3] R. Byers, C. Turner, and T. Brewer, "National vulnerability database," Distributed by NIST, Gaithersburg, MD, USA, doi: 10.18434/M3436.

[4] D. D. Bertoglio and A. F. Zorzo, "Overview and open issues on penetration test," *J. Brazilian Comput. Soc.*, vol. 23, no. 1, pp. 1–16, Dec. 2017.

[5] S. Shah and B. M. Mehtre, "An overview of vulnerability assessment and penetration testing techniques," *J. Comput. Virol. Hacking Techn.*, vol. 11, no. 1, pp. 27–49, Feb. 2015.

[6] S. Bi and Y. J. A. Zhang, "Graph-based cyber security analysis of state estimation in smart power grid," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 176–183, Apr. 2017.

[7] D. Fellner, T. I. Strasser, and W. Kastner, "Detection of misconfigurations in power distribution grids using deep learning," in *Proc. Int. Conf. Smart Energy Syst. Technol. (SEST)*, Sep. 2021, pp. 1–6.

[8] S. Figueroa-Lorenzo, J. Añorga, and S. Arrizabalaga, "A survey of IIoT protocols: A measure of vulnerability risk analysis based on CVSS," *ACM Comput. Surv.*, vol. 53, no. 2, pp. 1–53, Mar. 2021.

[9] P. Kamongi, S. Kotikela, K. Kavi, M. Gomathisankaran, and A. Singhal, "VULCAN: Vulnerability assessment framework for cloud computing," in *Proc. IEEE 7th Int. Conf. Softw. Secur. Rel.*, Jun. 2013, pp. 218–226.

[10] L. Kotlaba, S. Buchovecká, and R. Lórencz, "Active Directory Kerberoasting attack: Detection using machine learning techniques," in *Proc. 7th Int. Conf. Inf. Syst. Secur. Privacy*, 2021, pp. 376–383.

[11] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL injection based on artificial neural network," *Knowl.-Based Syst.*, vol. 190, Feb. 2020, Art. no. 105528.

[12] H. Wang, Z. Chen, J. Zhao, X. Di, and D. Liu, "A vulnerability assessment method in Industrial Internet of Things based on attack graph and maximum flow," *IEEE Access*, vol. 6, pp. 8599–8609, 2018.

[13] B. Ali and A. Awad, "Cyber and physical security vulnerability assessment for IoT-based smart homes," *Sensors*, vol. 18, no. 3, p. 817, Mar. 2018.

[14] V. Casola, A. de Benedictis, M. Rak, and U. Villano, "A methodology for automated penetration testing of cloud applications," *Int. J. Grid Utility Comput.*, vol. 11, no. 2, pp. 267–277, 2020.

[15] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," *J. Internet Services Appl.*, vol. 4, no. 5, pp. 1–13, 2013.

[16] S. A. P. Kumar and B. Xu, "Vulnerability assessment for security in aviation cyber-physical systems," in *Proc. IEEE 4th Int. Conf. Cyber Secur. Cloud Comput. (CSCloud)*, Jun. 2017, pp. 145–150.

[17] G. Nebbione and M. C. Calzarossa, "Security of IoT application layer protocols: Challenges and findings," *Future Internet*, vol. 12, no. 3, p. 55, Mar. 2020.

[18] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2702–2733, 3rd Quart., 2019.

[19] K. C. Park and D.-H. Shin, "Security assessment framework for IoT service," *Telecommun. Syst.*, vol. 64, no. 1, pp. 193–209, Jan. 2017.

[20] S. Ristov, M. Gusev, and A. Donevski, "Security vulnerability assessment of OpenStack cloud," in *Proc. 6th Int. Conf. Comput. Intell., Commun. Syst. Netw.*, May 2014, pp. 95–100.

[21] P. Saripalli and B. Walters, "QUIRC: A quantitative impact and risk assessment framework for cloud security," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, Jul. 2010, pp. 280–288.

[22] C.-W. Ten, C.-C. Liu, and G. Manimaran, "Vulnerability assessment of cybersecurity for SCADA systems," *IEEE Trans. Power Syst.*, vol. 23, no. 4, pp. 1836–1846, Nov. 2008.

[23] M. U. Aksu, M. H. Dilek, E. I. Tatli, K. Bicakci, H. I. Dirik, M. U. Demirezen, and T. Aykir, "A quantitative CVSS-based cyber security risk assessment methodology for IT systems," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2017, pp. 1–8.

[24] K. Beckers, M. Heisel, L. Krautsevich, F. Martinelli, R. Meis, and A. Yautsiukhin, "Determining the probability of smart grid attacks by combining attack tree and attack graph analysis," in *Proc. Int. Workshop Smart Grid Secur.* Cham, Switzerland: Springer, 2014, pp. 30–47.

[25] L. Gallon and J. J. Bascou, "Using CVSS in attack graphs," in *Proc. 6th Int. Conf. Availability, Rel. Secur.*, Aug. 2011, pp. 59–66.

[26] G. George and S. M. Thampi, "A graph-based security framework for securing industrial IoT networks from vulnerability exploitations," *IEEE Access*, vol. 6, pp. 43586–43601, 2018.

[27] N. Ghosh and S. K. Ghosh, "An approach for security assessment of network configurations using attack graph," in *Proc. 1st Int. Conf. Netw. Commun.*, 2009, pp. 283–288.

[28] N. Ghosh, I. Chokshi, M. Sarkar, S. K. Ghosh, A. K. Kaushik, and S. K. Das, "NetSecuritas: An integrated attack graph-based security assessment tool for enterprise networks," in *Proc. 16th Int. Conf. Distrib. Comput. Netw.*, Jan. 2015, pp. 1–10.

[29] M. Ibrahim, A. Alsheikh, and A. Matar, "Attack graph modeling for implantable pacemaker," *Biosensors*, vol. 10, no. 2, p. 14, Feb. 2020.

[30] D. Ivanov, M. Kalinin, V. Krundyshev, and E. Orel, "Automatic security management of smart infrastructures using attack graph and risk analysis," in *Proc. 4th World Conf. Smart Trends Syst., Secur. Sustainability*, Jul. 2020, pp. 295–300.

[31] J. Jia, Z. Dong, J. Li, and J. W. Stokes, "Detection of malicious DNS and web servers using graph-based approaches," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 2625–2629.

[32] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *J. Inf. Secur. Appl.*, vol. 29, no. 3, pp. 27–56, 2016.

[33] P. Lin and Y. Chen, "Dynamic network security situation prediction based on Bayesian attack graph and big data," in *Proc. IEEE 4th Inf. Technol. Mechatronics Eng. Conf. (ITOEC)*, Dec. 2018, pp. 992–998.

[34] S.-C. Liu and Y. Liu, "Network security risk assessment method based on HMM and attack graph model," in *Proc. 17th IEEE/ACIS Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distrib. Comput. (SNPD)*, May 2016, pp. 517–522.

[35] A. Nadeem, S. Verwer, S. Moskal, and S. J. Yang, "Alert-driven attack graph generation using S-PDFA," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 731–746, Mar. 2021.

[36] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proc. Workshop New Secur. Paradigms*, Jan. 1998, pp. 71–79.

[37] Q. Jia, S. Wang, C. Xia, and L. Lv, "Automatic generation algorithm of penetration graph in penetration testing," in *Proc. 9th Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput.*, Nov. 2014, pp. 531–537.

[38] R. W. Ritchey and P. Ammann, "Using model checking to analyze network vulnerabilities," in *Proc. IEEE Symp. Secur. Privacy*, May 2000, pp. 156–165.

[39] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing, "Automated generation and analysis of attack graphs," in *Proc. IEEE Symp. Secur. Privacy*, May 2002, pp. 273–284.

[40] P. D. Zegzhda, D. P. Zegzhda, and A. V. Nikolskiy, "Using graph theory for cloud system security modeling," in *Computer Network Security* (Lecture Notes in Computer Science), vol. 7531, I. Kotenko and V. Skormin, Eds. Cham, Switzerland: Springer, 2012, pp. 309–318.

[41] S. Wu, Y. Zhang, and W. Cao, "Network security assessment using a semantic reasoning and graph based approach," *Comput. Electr. Eng.*, vol. 64, pp. 96–109, Nov. 2017.
[42] A. Afaq, N. Haider, M. Z. Baig, K. S. Khan, M. Imran, and I. Razzak, "Machine learning for 5G security: Architecture, recent advances, and challenges," *Ad Hoc Netw.*, vol. 123, Dec. 2021, Art. no. 102667.
[43] U. A. Butt, M. Mehmood, S. B. H. Shah, R. Amin, M. W. Shaukat, S. M. Raza, D. Y. Suh, and M. J. Piran, "A review of machine learning algorithms for cloud computing security," *Electronics*, vol. 9, no. 9, p. 1379, Aug. 2020.
[44] H. Hanif, M. H. N. M. Nasir, M. F. A. Razak, A. Firdaus, and N. B. Anuar, "The rise of software vulnerability: Taxonomy of software vulnerabilities detection and machine learning approaches," *J. Netw. Comput. Appl.*, vol. 179, Apr. 2021, Art. no. 103009.
[45] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in IoT security: Current solutions and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1686–1721, 3rd Quart., 2020.
[46] H. Jiang, J. Nagra, and P. Ahammad, "SoK: Applying machine learning in security—A survey," 2016, *arXiv:1611.03186*.
[47] S. M. Tahsien, H. Karimipour, and P. Spachos, "Machine learning based solutions for security of Internet of Things (IoT): A survey," *J. Netw. Comput. Appl.*, vol. 161, Jul. 2020, Art. no. 102630.
[48] W. Melicher, C. Fung, L. Bauer, and L. Jia, "Towards a lightweight, hybrid approach for detecting DOM XSS vulnerabilities with machine learning," in *Proc. Web Conf.*, Apr. 2021, pp. 2684–2695.
[49] Q. Cheng, C. Wu, H. Zhou, D. Kong, D. Zhang, J. Xing, and W. Ruan, "Machine learning based malicious payload identification in software-defined networking," *J. Netw. Comput. Appl.*, vol. 192, Oct. 2021, Art. no. 103186.
[50] O. Valea and C. Oprisa, "Towards pentesting automation using the Metasploit framework," in *Proc. IEEE 16th Int. Conf. Intell. Comput. Commun. Process. (ICCP)*, Sep. 2020, pp. 171–178.
[51] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. Cham, Switzerland: Springer, 2013.
[52] G. Grillenmeier, "Now's the time to rethink Active Directory security," *Netw. Secur.*, vol. 2021, no. 7, pp. 13–16, Jul. 2021.
[53] S. Berkouwer, *Active Directory Administration Cookbook*. Birmingham, U.K.: Packt, 2019.
[54] B. Sheresh and D. Sheresh, *Understanding Directory Services*. Carmel, IN, USA: Sams, 2002.
[55] B. Desmond, J. Richards, R. Allen, and A. G. Lowe-Norris, *Active Directory: Designing, Deploying, and Running Active Directory*. Sebastopol, CA, USA: O'Reilly Media, 2008.
[56] J. Dias, "A guide to Microsoft Active Directory (AD) design," Dept. Energy Lawrence Livermore Nat. Lab., Livermore, CA, USA, Tech. Rep., UCRL-MA-148650, 2002.
[57] G. Nebbione, "AI-aware network security assessment: A graph based approach," Ph.D. thesis, Dept. Elect., Comput. Biomed. Eng., Univ. Pavia, Pavia, Italy, 2021.

GIUSEPPE NEBBIONE received the Ph.D. degree in computer engineering from the Department of Electrical, Computer and Biomedical Engineering, University of Pavia, Italy. He is currently a Postdoctoral Researcher with the Royal Institute of Technology (KTH), Sweden. His research interests include computer security, graph theory, and machine learning.

MARIA CARLA CALZAROSSA (Senior Member, IEEE) is currently a Professor of computer engineering with the Department of Electrical, Computer and Biomedical Engineering, University of Pavia, Italy. Her research interests include performance evaluation and workload characterization of complex systems and services, social networks, cloud computing, and benchmarking.

Open Access funding provided by 'Università degli Studi di Pavia' within the CRUI CARE Agreement