electronics

Article

Unsupervised Learning for Lateral-Movement-Based Threat Mitigation in Active Directory Attack Graphs

David Herranz-Oliveros [1], Marino Tejedor-Romero [1], Jose Manuel Gimenez-Guzman [2,*] and Luis Cruz-Piris [1]

1 Departamento de Automática, Universidad de Alcalá, 33,600, 28805 Madrid, Spain; david.herranz@uah.es (D.H.-O.); marino.tejedor@uah.es (M.T.-R.); luis.cruz@uah.es (L.C.-P.)
2 Departamento de Comunicaciones, Universitat Politècnica de València, 46022 Valencia, Spain
* Correspondence: jmgimenez@upv.es

Abstract: Cybersecurity threats, particularly those involving lateral movement within networks, pose significant risks to critical infrastructures such as Microsoft Active Directory. This study addresses the need for effective defense mechanisms that minimize network disruption while preventing attackers from reaching key assets. Modeling Active Directory networks as a graph in which the nodes represent the network components and the edges represent the logical interactions between them, we use centrality metrics to derive the impact of hardening nodes in terms of constraining the progression of attacks. We propose using Unsupervised Learning techniques, specifically density-based clustering algorithms, to identify those nodes given the information provided by their metrics. Our approach includes simulating attack paths using a snowball model, enabling us to analytically evaluate the impact of hardening on delaying Domain Administration compromise. We tested our methodology on both real and synthetic Active Directory graphs, demonstrating that it can significantly slow down the propagation of threats from reaching the Domain Administration across the studied scenarios. Additionally, we explore the potential of these techniques to enable flexible selection of the number of nodes to secure. Our findings suggest that the proposed methods significantly enhance the resilience of Active Directory environments against targeted cyber-attacks.

Keywords: cybersecurity; lateral movement; threat mitigation; unsupervised learning; attack graphs; active directory; hardening placement

check for updates

Citation: Herranz-Oliveros, D.; Tejedor-Romero, M.; Gimenez-Guzman, J.M.; Cruz-Piris, L. Unsupervised Learning for Lateral-Movement-Based Threat Mitigation in Active Directory Attack Graphs. *Electronics* **2024**, *13*, 3944. https://doi.org/10.3390/electronics13193944

Academic Editors: Abdussalam Elhanashi and Pierpaolo Dini

Received: 4 September 2024
Revised: 30 September 2024
Accepted: 4 October 2024
Published: 6 October 2024

1. Introduction

The digital security landscape is increasingly threatened by sophisticated attackers who navigate and manipulate enterprise networks to compromise critical assets. In this context, preventing or significantly impeding lateral movement within networks is a crucial defense strategy for cybersecurity professionals. The goal is to limit the reach of such incursions and slow down adversaries as they attempt to breach the targeted infrastructure. This task is particularly challenging in complex network environments, where maintaining a balance between effective security protocols and optimal network performance is essential.

Networks incorporating Microsoft Active Directory (AD) are especially vulnerable to these challenges. AD, predominantly used in Windows-based environments to manage user permissions and resources, is a prime target for cybercriminals due to its central role in network administration [1]. A breach in AD infrastructure can lead to severe disruptions, as evidenced by several high-profile cyber incidents in recent years.

Examples include operations by the self-named Ransomware as a Service (RaaS) group, Black Basta, between 2022 and 2024, which compromised over 500 organizations. They exploited AD environments to perform domain enumeration and expand their attack surface within compromised networks. Another recent case is the ransomware attack on Ascension Health by the Akira group in May 2024, disrupting critical healthcare operations across the United States. These incidents underscore the significance of such threats that can compromise AD, as well as the serious consequences that follow. Both of these threats have recently been highlighted by the U.S. Cybersecurity and Infrastructure Security Agency (CISA) in their #*StopRansomware* communications series [2,3].

Given this context, there is an urgent need for innovative defense mechanisms that are both effective and minimally disruptive to network performance. Our research proposes the use of Unsupervised Learning (UL) techniques, specifically density-based clustering algorithms based on centrality metrics, to enhance the security of AD infrastructures. This approach leverages concepts from graph theory to model the logical architectures that govern permissions and trusts in these environments, as well as from epidemiological models to simulate threat propagation within them [4].

As shown in Figure 1, our proposed methodology utilizes graph-based models of AD infrastructure, where network actors (computers, users, groups, domains, etc.) are represented as nodes connected by edges that denote different relationships between them.

We model the progression of attackers from their initial access point to critical organizational assets using an Attack-Path-Based (APB) approach [5]. This involves simulating a snowball attack [6,7], where attackers advance by chaining credentials and permissions from the network's entry point towards their targets, progressively compromising more AD components. In network graph terms, attackers compromise more nodes as they traverse the connecting edges. The resulting paths from the network's periphery to its central critical elements form the attack paths, which are the focus of this research.

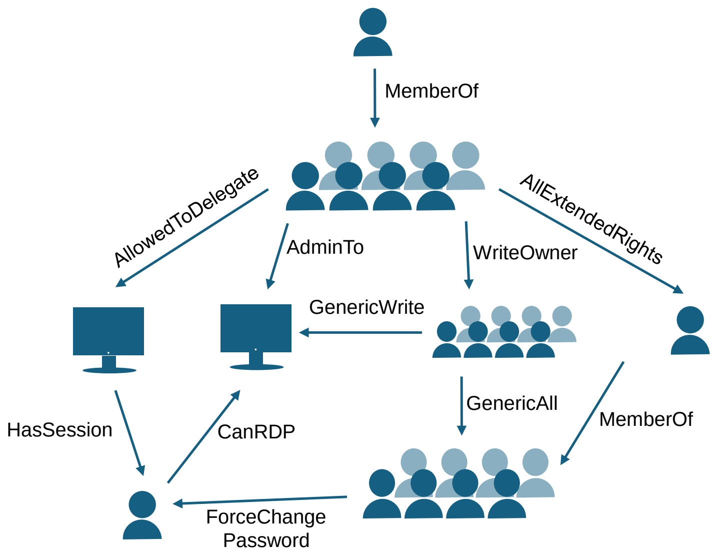

Figure 1. Example of an AD graph model illustrating some typical relationships between network actors.

We apply centrality metrics to assess the strategic importance of each node in potential attack paths to domain management. By employing unsupervised density-based clustering on these metrics, we identify and prioritize critical points where hardening (i.e., applying safeguards) can significantly mitigate the risk of lateral movement, thus protecting the organization's critical assets while minimizing operational disruption.

This strategic immunization approach mimics epidemiological interventions, targeting nodes whose security enhancements yield significant network-wide benefits. Through extensive simulations of potential attack scenarios, we assess the effectiveness of our immunization strategies in curbing the success of cyber threats. These scenarios include evaluations on both a real network graph and three more generated using various widely referenced tools (BadBlood, BloodHoundDBCreator, AD Simulator).

In parallel, we evaluate two well-known clustering algorithms (DBSCAN [8] and HDBSCAN-GLOSH [9]) to determine their respective benefits based on the nature of the analyzed networks.

We adopt a Time to Compromise (TTC)-based evaluation framework [10], which highlights the impact of our measures on the time it takes for attackers to compromise the network, specifically targeting the Domain Administration (DA). In summary, the novel contributions of our research that advance this goal are as follows:

- Network graph modeling is applied to AD infrastructures, focusing on generating subgraphs that capture the dynamics of attack paths that lateral-movement-based threats might follow from the network periphery to the DA. This approach optimizes the subsequent application of UL techniques. Unlike other studies, our research analyzes four varied AD graphs, including one from a real network infrastructure. Additionally, related studies limit their scope to a few specific types of edges in the AD attack graph (typically *MemberOf*, *AdminTo*, and *HasSession*), whereas our model encompasses all possible enumerated edges.
- UL techniques combined with centrality metrics commonly applied in epidemiology are used to identify network elements whose hardening can delay the time it takes for threats to reach the DA in the AD, thereby reducing the likelihood of full network compromise. In contrast to most related studies, we do not consider these metrics in isolation but instead employ clustering to detect global anomalies throughout the attack graph.
- HDBSCAN-GLOSH is analyzed and evaluated as a method for identifying candidate network points for hardening, which does not depend on clustering groupings. This includes a comparative accuracy analysis against DBSCAN for the same purpose. This method enables the flexible determination and prioritization of the desired number of candidates based on a continuous scoring system.
- Inspired by epidemiology, we assess the impact of hardening identified network points through a tailored compartmental Susceptible–Infected (SI) propagation model, which realistically simulates all possible lateral movement dynamics from the network's periphery (where threats typically originate) to the DA. Our model stands out by considering multidirectional propagation, allowing us to analyze the global impact of intrusions on the attack graph without requiring the attacker to iteratively choose a specific path.
- Yielding on the stochastic nature of the aforementioned propagation model, we construct a Continuous-Time Markov Chain (CTMC) to reflect, probabilistically, the impact of countermeasures on slowing down attack progress (i.e., a higher success rate per edge decreases the time it takes for an attacker to traverse it).
- We address the mitigation of targeted threats in AD environments as a node-hardening–placement problem rather than one of node-blocking or edge-blocking. We achieve this by modeling hardening a node as a delay factor α on the average time required to compromise it rather than completely preventing the attacker from being able to achieve it (i.e., $\alpha = \infty$). This enables us to propose a set of nodes for hardening (node budget) that are not restricted to distinct paths towards the DA but are instead distributed across the entire graph.
- Regarding the temporal nature of our proposed model, we define a novel TTC-based metric that quantifies the delay imposed on attackers reaching the DA after applying countermeasures. This is measured throughout extensive simulations as the Median Time to Compromise DA (MTCDA).

Our approach aims to enhance the resilience of AD environments by providing a defense mechanism primarily intended for rapid and effective protection following the detection or suspicion of an ongoing intrusion. Beyond this, our ultimate goal is to offer a scalable framework for securing all types of complex and hierarchical network infrastructures against lateral movement in targeted cyber-attacks. Our experimental results demonstrate that the use of UL-based hardening–placement techniques can delay threats by 2 to 7 times in reaching the central DA of the AD, depending on the analyzed graph and the delay factor α associated with the application of countermeasures. Additionally, the findings suggest that depending on the topology of the graphs, in most cases, comparable or even superior benefits can be obtained by employing techniques that provide greater flexibility in selecting the number of nodes to harden.

This paper is structured as follows: Section 2 discusses the most relevant works related to our research. Section 3 presents the system model, including how we model lateral movement processes on AD graphs, the graphs we analyze, and the considerations regarding the origin and targets of the threats we study. Section 4 introduces the UL-based algorithms that underpin our proposed node identification strategies for threat mitigation, the centrality metrics on which they are applied, and the subgraphs on which these metrics are calculated. Section 5 presents our study's results. Finally, Section 6 summarizes our contributions and outlines further research avenues.

2. Related Work

Recent advancements in cybersecurity have leveraged graph-based models, Machine Learning (ML), and Reinforcement Learning (RL) to enhance threat detection and mitigation in multiple environments. Modeling networks as graphs to study lateral movement has been extensively utilized in previous research [4,11,12]. These models help visualize and analyze the complex relationships within a network, allowing for effective detection and mitigation of lateral movement threats. For instance, Ref. [13] employed graph theory to propose a centrality-based methodology for analyzing and reducing potential attack vectors in Microsoft cloud environments. Unlike our approach, their proposal focuses exclusively on Microsoft Azure capabilities and uses centrality metrics isolated from each other. In [14], a methodology to build a threat model based on multilayer graphs is proposed, alongside a set of techniques to reconfigure the network in order to mitigate the risk over assets.

At the same time, ML and RL have emerged as powerful tools for enhancing cybersecurity and network resilience. Ref. [15] developed ML-driven anomaly detection systems to protect against zero-day attacks, and Ref. [16] explored deep learning techniques for threat detection and anomaly analysis. Refs. [17,18] proposed the use of Graph Neural Networks (GNNs) for detecting Structural Hole Spanners (SHS) in generic dynamic networks. These studies highlight the crucial role of advanced learning algorithms in improving the detection and response capabilities against sophisticated cyber threats.

Deepening in graph-based models, the concept of APB analysis is fundamental to understanding and mitigating lateral movement in networks. Ref. [5] introduced APB analysis, where network relationships are viewed as "hops" forming attack paths within a graph. Ref. [7] developed Heat-ray, a system combining ML and combinatorial optimization to reduce the potential of identity snowball attacks within large networks. Ref. [19] further contributed to this field by modeling hops between machines in parallel graphs, utilizing centrality metrics to rank nodes for immunization (once again, unlike us, using these metrics in isolation from one another).

Building on the concept of immunization, epidemiological approaches have been adapted from public health to various fields, including cybersecurity, to model and mitigate the spread of threats. Refs. [20,21] discussed the application of these epidemiological models in public health, which have also been applied to social networks [22,23] and financial analysis [24]. The authors of [25] conducted an in-depth study on the dynamics of epidemiological infection processes in generic networks, leading to the development of a framework for simulating these processes [26], which enables the evaluation of the impact of mitigation strategies in real infrastructures. In the area of epidemiology and cybersecurity, the works of [4,10] stand out by modeling the spread of cyber threats as analogous to epidemiological infections. Notably, Ref. [4] proposed the use of clustering for identifying key nodes in lateral-movement-based threats spreading within AD networks. However, unlike our approach, their methodology did not experimentally employ a tailored epidemiological model and relied solely on DBSCAN applied to authentication graphs between computers. In [10], the authors further extended these methodologies to whole AD network graphs, integrating centrality metrics with k-shell decomposition to enhance node selection for immunization and network reconfiguration. Unlike our current approach, their proposal focuses on mitigating non-targeted threats (i.e., without a defined target) and does not explore the use of more flexible node-ranking strategies for hardening–placement, such as those based on HDBSCAN-GLOSH.

Focusing on the defense against threats in AD attack graphs, several recent works are relevant to our study, employing both edge-blocking strategies [6,27–31] and node-blocking strategies [32,33]. Refs. [27–30] primarily focus on using RL-based strategies to generate attack policies, which are then used to apply Evolutionary Diversity Optimization (EDO) techniques to identify optimal edge budgets for defense. In contrast, Refs. [6,31] adopt Fixed-Parameter Tractable (FPT)-based approaches to achieve optimal defense. However, scalability issues lead them to tightly link their proposals to the ideal characteristics of AD networks (assuming high tree-likeness and absence of cycles) or resort to GNN-based heuristics. All of these proposals are inapplicable to the problem we address, as they focus on identifying edges rather than nodes. Interestingly, although these studies use probabilistic models as we do, they distinguish between the failure rate, detection rate, and success rate for each edge, except for the authors of [31]. Similar to this work, and since we do not tackle threat detection, we focus exclusively on the success rate.

In [32,33], while the focus is indeed on node identification, their approach focuses on detection via decoy placement rather than threat mitigation. Specifically, Ref. [33] seeks to ensure intrusion detection by positioning decoys that, within a given budget, cover the maximum number of shortest paths (SPs) between the DA and a defined set of entry points to the network. Ref. [32] extends this approach by adopting a temporal-based approach, similar to our methodology (TTC-based), but with the aim of maximizing the time between threat detection and the attacker reaching the target.

These edge-blocking and node-blocking strategies also differ from our work in their use of a Stackelberg Game model, where the attacker and defender make iterative decisions based on each other's actions. This contrasts with our focus on hardening–placement strategies, as these approaches rely on blocking strategies, either by removing links (i.e., failure rate = 1) or assuming detection as the primary means of mitigation. Additionally, they limit their scope to a few specific types of edges in the AD attack graph (typically *MemberOf*, *AdminTo*, and *HasSession*), whereas our model encompasses all possible enumerated edges. Finally, none of these studies incorporate real AD attack graphs into their experiments, further distinguishing our approach.

In conclusion, the integration of graph-based models, ML, RL, and epidemiological techniques provides a comprehensive framework for enhancing cybersecurity in AD environments. The synergy between these approaches facilitates effective threat detection, mitigation, and overall network security, underscoring the importance of continuous advancements and interdisciplinary methodologies in this critical field.

3. System Model

Our proposal relies on a network model using directed graphs designed to efficiently and accessibly aggregate all information regarding an organization's various assets and the interdependencies among them. We specifically target Microsoft AD, the predominant tool for network management at the enterprise or organizational level [34]. During our research, we evaluated our proposal using four distinct AD graphs—one derived from a real, anonymized network infrastructure (referred to as RS) and three generated using different synthetic tools: BadBlood https://github.com/davidprowe/BadBlood (accessed on 1 January 2024), AD Simulator https://github.com/nicolas-carolo/adsimulator (accessed on 1 January 2024), and BloodHound DB Creator https://github.com/BloodHoundAD/BloodHound-Tools/tree/master/DBCreator (accessed on 1 January 2024) (referred to as BB, AS, and BH, respectively). RS and BB graphs were collected using SharpHound https://github.com/BloodHoundAD/SharpHound (accessed on 1 January 2024). AD

Simulator and BloodHound DB Creator are widely used in AD security research [27–30,32,33] due to the lack of studies using real graphs because of their sensitive data. To enrich our analysis, we also incorporate BadBlood, a well-regarded tool within the AD community. Unlike typical synthetic graph generators, it automates the deployment of a whole simulated AD domain, which we subsequently enumerate as a real graph. As we will discuss later, the structure of the BB graph closely resembles that of the RS graph, especially concerning the low exposure of the DA. See Table 1 for detailed information on the nodes and edges of all these graphs.

Table 1. An overview of node and edge categories in the studied graphs based on AD elements. * AD Simulator generated a graph with four domains: one primary and three empty ones. All analyses in this paper focus solely on the primary domain.

Graph		**RS**	**BB**	**AS**	**BH**
Node Type	User	102,589	99,957	10,004	99,957
	Computer	3236	3236	341	3237
	Group/Container	984	1026	154	1005
	OU	399	223	41	21
	GPO	159	2	18	22
	Domain	1	1	4 *	1
		107,368	104,445	10,562	104,243
Edge Type	AddKeyCredentialLink	0	206,382	0	0
	AddMember	959	0	3	1
	AdminTo	85	5	1467	11,181
	AllExtendedRights	734	0	10,070	0
	AllowedToDelegate	0	0	64	646
	CanPSRemote	0	3	64	0
	CanRDP	18	3	64	645
	Contains	106,180	104,390	10,386	1021
	DCSync	0	11	0	0
	ExecuteDCOM	0	2	64	646
	ForceChangePassword	204,728	0	3	1
	GenericAll	535,998	598,000	31,416	104,191
	GenericWrite	1471	104,219	10,528	2
	GetChanges	4	1	3	2
	GetChangesAll	2	1	2	3
	GetChangesInFilteredSet	0	1	0	0
	GpLink	198	2	33	42
	HasSession	449	0	15,086	99,827
	MemberOf	636,600	118,128	125,419	453,667
	Owns	105,936	158	10,527	0
	ReadLAPSPassword	0	0	2	0
	SQLAdmin	1	0	0	0
	TrustedBy	0	0	4	0
	WriteDACL	108,271	0	10,636	4
	WriteOwner	53	261	10,632	4
		1,701,687	1,131,567	236,473	671,883

Our primary objective is to assess the risk exposure of critical network elements. One prevalent risk factor is lateral movement, which enables attackers to compromise one or more low-value network elements and subsequently navigate through the network by exploiting multiple security vulnerabilities until critical assets are reached. To represent this behavior in our network graphs, we model a snowball attack [6,7] on AD. This entails the attacker's initial access to various elements of the AD network (e.g., users, computers) and the progressive expansion of the attack, element by element, through the chained use of stolen credentials and the exploitation of existing privileges among the network's entities. For cybersecurity purposes, this type of attack involving the spread of malware is categorized as a lateral-movement-based attack.

3.1. Lateral Movement through Active Directory as an Epidemiological Process

Most proposals related to AD defense cited in Section 2 utilize a Stackelberg Game model. In this framework, the attacker, starting from a defined set of entry points, makes decisions based on the defensive strategy while considering various conditions (such as prior knowledge of the network, the risk of detection, the ability to reattempt exploitation of edges, etc.).

In contrast, our approach evaluates hardening–placement in a much broader and demanding context. We simulate an attacker who moves indiscriminately throughout the network, similar to an epidemiological process, where any prior knowledge of the network structure is irrelevant and any technique for exploiting an edge in the attack graph is applicable. This model enables us to accurately capture all possible lateral-movement-based attack dynamics on AD.

We utilize an SI epidemiological model [25], which is extensively employed across various research fields, including cybersecurity. We model lateral movement as a probabilistic infection mechanism within a compartmental framework. Each simulation begins with a single initially infected node and concludes once the DA is compromised. This approach allows us to evaluate the effectiveness of threat mitigation by strategically hardening specific nodes, thereby slowing the progression toward the DA takeover.

We utilize an SI model rather than a Susceptible–Infected–Recovered (SIR) model, as the SI model allows for both preventive and reactive countermeasures. This is because the countermeasures considered in our research can be applied preventively or reactively without altering the dynamics of the SI model. These countermeasures influence the infection transmission rate between nodes in the graph rather than the state transitions (i.e., recovery) of already infected nodes. Here, S represents the set of susceptible nodes and I the infected nodes. At any time t, $S(t)$ and $I(t)$ denote their respective counts. The infection rate is β, with N being the total number of nodes. The dynamics of infection are captured by the following equations:

$$\frac{dS}{dt} = -\beta I(t)\frac{S(t)}{N}; \frac{dI}{dt} = \beta I(t)\frac{S(t)}{N} \tag{1}$$

This model assumes nodes transition from susceptible to infected. The infection rate β is expressed as τN, where τ is the transmission rate per edge. Thus, the equation for $\frac{dI}{dt}$ becomes:

$$\frac{dI}{dt} = \tau I(t)(N - I(t)) \tag{2}$$

The success rate per edge inherent in node-blocking and edge-blocking models is analogous to this transmission rate τ, carrying over its meaning to the hardening–placement problem we address. This factor allows us to model the difficulty an attacker faces in compromising a specific relationship in the graph. The higher the transmission rate of an edge, the shorter the expected time for the attacker to compromise it. The impact of hardening on this transmission rate will be discussed further in Section 5.1.

To incorporate randomness, we model the process as a CTMC using Kolmogorov equations to describe the probability of different numbers of infected nodes over time. Building on this model, we created a tailored Discrete Event Simulator (DES) to study the infection process based on the models implemented in [26]. The simulation begins with an initially infected node located as far as possible from the DA and proceeds to compromise it. Initially, all nodes are marked as susceptible, and infection events for the initial compromised node are scheduled at $t = 0$. The simulator processes the event queue iteratively, with infection events governed by an exponentially distributed ($X \sim \text{Exp}(\lambda)$) time interval Δt, determined by the rate $\lambda = \tau^{-1}$. At each iteration of the simulator at a given time t, the node(s) scheduled for infection will transition to an infected state (i.e., $S \to I$). New infection events will then be queued for all newly reachable nodes from those just infected, provided they remain susceptible. Notably, if a node has an infection event scheduled in the queue, but a new event for the same node is queued for an earlier time, the original event will be replaced by the incoming one.

This behavior of our DES is key to understanding the multidirectional nature of the attacker we model. It allows us to assess the effectiveness of our approach under attack conditions that go far beyond an attacker simply choosing the most optimal available path into the network while remaining undetected. Consider, for instance, an Advanced Persistent Threat (APT) whose presence in the network is suspected or even detected, but the full extent of the intrusion remains unknown.

Our main goal would be to avoid network collapse by minimizing the time between threat detection and complete remediation or, conversely, by delaying DA compromise as long as possible. In this scenario, we would not know from what distance to which the DA countermeasures should be applied without severely affecting the network's functionality. Additionally, the attacker might pivot through suboptimal paths to ensure persistence. Therefore, a hardening–placement strategy that proposes nodes distributed throughout the network (regardless of their distance from the DA, and even including those along the same attack paths) by prioritizing hardening over blocking (i.e., reinforcing paths rather than splitting them) emerges as a critical defense mechanism since maximizes network coverage by securing the largest possible segment [35,36].

Our DES is designed to evaluate different hardening–placement strategies by simulating a multidirectional attacker as described, subjecting them to all potential lateral movement dynamics, including the aforementioned scenarios.

3.2. Domain Administration as the Target of the Attackers

While our previous research focused on the analysis of lateral movement spreading horizontally in AD networks [10], our current contribution aims to conduct an analogous analysis regarding the vertical spreading of attackers (i.e., privilege escalation). But what does this horizontal–vertical duality mean in terms of lateral movement propagation?

To understand this differentiation, it is essential to recall the network model we are working with. In this model, we find numerous elements of the organization's network, as well as all the trust relationships, privilege grants, and memberships among them. It is no secret that the ultimate goal of attackers is to compromise the element with the highest functional capacity over the network (i.e., the highest level of privilege). Since we are dealing with environments orchestrated by AD, this critical asset is the main server or Domain Controller (DC), which is responsible for the overall DA. Based on the data model inherent to the graphs we analyze, we observe that, in all cases, administrative users or groups, as well as the computer acting as DC (if enumerated in the graph), ultimately have a direct outgoing link to the Domain-typed node. From now on, we will refer to this single node as the DA. Notice that, for simplicity, our study does not consider multi-domain AD networks.

Furthermore, keeping in mind Microsoft's privilege grant pyramid for AD-based environments (AD Tier Model), we observe a well-defined vertical hierarchy with several levels or tiers. According to this hierarchy, the DC and other assets capable of compromising the entire network are located in tier 0. These critical assets are logically the least numerous. Conversely, non-administrator user accounts and common workstations are the most numerous elements at the base of this pyramid (i.e., tier 2). Their lower value within the organization typically confers a lower degree of cybersecurity hardening. When considering that these elements are often operated by non-IT users or individuals without security knowledge, it becomes evident that one or more elements from this tier are likely to be initially compromised during an attack. This is usually where lateral movement attempts to progress from. In real-world scenarios, this often happens through social engineering techniques, phishing or spear phishing, mail spoofing, etc.

With this hierarchical pyramid in mind, it becomes much easier to understand what we mean by horizontal or vertical progression of lateral movement. When discussing points in the network that, if compromised, enable attackers to access a broad segment of the network (regardless of the value or privilege level of those exposed elements), we refer to the horizontal expansion of the lateral movement. Conversely, when discussing points in the network that allow attackers to access higher privilege elements, whether few or many, we refer to vertical progression. This is analogous to malware spreading across an AD environment horizontally along the general privilege pyramid or escalating from the bottom to the top.

In [4], the author explains how the spread of cyberattacks is fundamentally due to three types of nodes in the affected organization, based on the role they play in lateral movement progression: spreaders, gatekeepers, and escalators. Considering the network as a directed graph to model a snowball attack, spreaders are nodes that allow access to many others, i.e., horizontal propagation. On the other hand, we have nodes involved in the vertical propagation of threats, known as escalators and gatekeepers.

Escalators are nodes that enable access to higher privilege nodes, i.e., performing cross-tier logins. A clear example would be the exposure of an administrator's credentials (tier 0 or 1) on a machine regularly used by non-administrator accounts (tier 2). This could occur due to the residual presence of privileged credentials from a previous session, such as one initiated for maintenance tasks. In the attack graph, this scenario would manifest as a *HasSession* edge between the two nodes, allowing an attacker, after compromising the machine, to retrieve those credentials and, therefore, escalate.

Gatekeepers, on the other hand, are topologically close nodes or immediate ancestors of the escalators within the attack graph. This means that an attacker must pass through them to reach the escalators. Their role is specifically defined to designate them as detection agents where monitoring measures can be applied. However, since our work focuses on a post-detection scenario (or one where the risk of detection does not influence the attacker's behavior), we can operationally consider them as escalators since they also contribute indirectly to the attacker's vertical progression through the network.

Our objective is to identify this latter group (escalators and gatekeepers) related to the attackers' progression from the most mundane elements in tier 2 to the top of the pyramid in tier 0, the DA.

3.3. Identifying Entry Points to the Network

In AD environments, attackers typically aim to compromise the DA. To model a directional snowball attack, we must consider not only the targets of intrusions but also their origins. As previously mentioned, attackers usually gain initial access to the network by compromising one or more non-administrative elements with low privilege levels and, consequently, low value to the organization.

Following this premise, to evaluate the effects of threat mitigation in AD during our research, we define the set of entry points P as the network elements (nodes in the network graph) from which the attacker begins lateral movement during simulations. Inspired by existing research [6,31], we select these nodes based on their distance from the DA in the network graph. These nodes must satisfy the following conditions:

- Each entry point $p \in P$ must be part of the node set V in the analyzed graph G.
- Each entry point $p \in P$ must have at least one path leading to the DA (i.e., $p \to \text{DA}$).
- The SP from any p to the DA ($\text{SP}(p, \text{DA})$) must have the maximum path length (i.e., the number of hops) among all nodes $v \in V$ in the graph G that have a path to the DA. Note that nodes v without any path to the DA (i.e., $v \nrightarrow \text{DA}$) are excluded from this calculation.

Considering these conditions, we formally define our selection of the entry points for attackers into the network as follows:

$$P = \{p \in V \mid (p \to \text{DA}) \wedge \text{SP}(p, \text{DA}) = \max(\{\text{SP}(v, \text{DA}) \mid v \in V \wedge (v \to \text{DA})\})\} \quad (3)$$

Additionally, we define the set R of non-reaching nodes as those nodes from which there is no path to the DA in G.

$$R = \{r \in V \mid (r \nrightarrow \mathrm{DA})\} \tag{4}$$

4. Unsupervised Learning for Active Directory Threat Mitigation

4.1. Domain Reachability Graphs

In the literature on attack graph studies, both generally and specifically for AD, attack graphs typically represent the paths an attacker might traverse to achieve their objectives, given one or more initial starting points [6]. In our study, we begin with network graphs representing all interactions related to privilege grants and memberships among network elements indiscriminately. We use the previously defined entry points P as access points for attackers, and the node representing the DA as their ultimate target. The subgraph extracted from the original network graph—considering the starting point, tentative attack destination, and the routes in between—can already be considered an attack graph. However, the method for extracting this attack graph, as well as the volume of data or routes it contains, is not arbitrary.

Initially, we considered composing an attack graph that strictly contains all SP($p \in P \rightarrow$ DA). This graph would exhibit a Directed Acyclic Graph (DAG) nature, ensuring that all optimal paths an attacker would take to achieve their goal are reflected. If we also ensure a tree-like structure by guaranteeing only one path $i \rightarrow j$ between each pair of vertices (i, j), the resulting attack graph could accurately reflect the attack dynamics leading attackers optimally to the DA. But what should be the nature of the attack graph?

To answer this question, we must consider the analytical purpose of our desired attack graph. Our goal is to analyze the centrality of the attack graph nodes via UL (specifically using density-based clustering algorithms) to determine nodes where applying security safeguards would drastically reduce attackers' success chances. Therefore, referencing an attack graph that only reflects the optimal route from each node p to the DA ensures threat mitigation for those specific routes, without considering all other possible routes. These include other routes of equal length if we have taken only one SP($p \rightarrow$ DA) between each pair (p, DA), as well as longer routes (e.g., only one hop longer), even if we calculated all SP($p \rightarrow$ DA) of equal length between each pair. Additionally, the tree-like nature of the optimal SP-based attack graph conceals another possibility for attackers during analysis: the ease with which attackers pivot from one route to another.

Thus, we conclude that we are interested in the attack graph capturing the maximum number of possible routes and their interactions without cycles. Logically, this graph would contain not only the SPs but also all paths mediating between P and the DA (i.e., $\{(p \rightarrow \mathrm{DA}) \mid p \in P, (p \rightarrow \mathrm{DA}) \subseteq G\}$). The granularity of the solution we seek will depend on this. If we aim to mitigate the most critical network routes effectively and in isolation, the attack graph will contain only one or a subset of SPs between each pair (p, DA). If we seek a solution to apply safeguards in a general way throughout the network, the subset of SPs will grow (first taking, one by one, all the SPs of equal length, then starting to take, one by one, the shorter ones of greater length than the first ones, and so on) until finally resulting in the graph of all paths between each pair (p, DA).

Unfortunately, calculating the attack graph based on all routes or a subset of SPs between P and the DA is computationally impractical. Therefore, our proposed approach introduces an attack graph we call the Domain Reachability Graph (DRG). Our interest is to cover as much of the network as possible when determining where to apply safeguards to evaluate our proposal under the most demanding scenario. Thus, the DRG theoretically seeks to approximate that graph encompassing all paths of any length between each pair (p, DA) by finding the graph that contains all nodes involved in at least one route to the DA.

Recalling the previously defined set R as non-reaching nodes, the graph we seek contains all nodes $v \in V$ in the network graph G that are not part of the set R. Additionally, the set E of edges in the original graph is reduced to E', containing only the edges mediating between nodes also included in the resulting DRG. Thus:

$$\mathrm{DRG}(V',E'):\begin{cases} V' = \{v \in V \mid v \notin R\} \\ E' = \{(u,v) \in E \mid u \in V' \wedge v \in V'\} \end{cases} \tag{5}$$

Finally, it is worth noting that this DRG generation mechanism does not exempt us from encountering cycles among different routes to the DA. While this might compromise the precision of the subsequent centrality-based analysis, we must consider the inherent tree-like nature of logical AD graphs like those we work with [6,31]. Since we generate the DRG through a mere node elimination process, the original graph's tree-like nature will remain. This property ensures the minimal presence of cycles along the routes between different node pairs in the graph and, thus, the impact of ignoring them when calculating node centrality. Notably, given the characteristics of the simulation process carried out to evaluate the proposal's effectiveness, detailed in Section 5.1, the presence of cycles in the graph will not affect the obtained results.

4.2. Centrality Analysis

Below, we present the centrality metrics calculated prior to applying UL-based techniques to them. We draw on multiple previous works, particularly those focused on cybersecurity and attack graphs [4,10,13,19], to select a set of six widely used centrality metrics that, in our view, provide valuable insights from sufficiently orthogonal perspectives when evaluating the role nodes play in the overall connectivity of the aforementioned DRGs. We decided to use a set of well-known metrics that, although widely used in the security domain, are as general-purpose as possible. A deeper exploration remains to assess the performance of more specialized metrics tailored to specific contexts.

It is worth noting that [4,10] have already demonstrated the effectiveness of using these metrics, among others, to identify key nodes relevant to lateral movement on AD through the application of density-based clustering. Ref. [13] also addresses AD security using some of the metrics we propose. Additionally, Ref. [19] highlights the benefits of adapting the calculation of these metrics to specific attack subgraphs when dealing with threats targeting a defined objective, rather than considering the entire network. In our case, we achieve this by calculating these metrics on our DRGs. Let V be the set of nodes, and E the set of edges of a given DRG:

- The betweenness centrality B_i of a node i is a measure of the node's influence over the flow of information in the network:

$$B_i = \sum_{\substack{a \neq i \neq b \\ a \neq b}} \frac{\delta_{ab}(i)}{\delta_{ab}}, \tag{6}$$

  Here, δ_{ab} denotes the total number of SPs from node a to node b, and $\delta_{ab}(i)$ represents the count of those paths that pass through node i, normalized by dividing by $(V-1)(V-2)$.
- Closeness centrality C_i for a node i indicates how close the node is to all other nodes in the network:

$$C_i = \sum_{j \neq i} \frac{1}{d_{ij}}, \tag{7}$$

  where d_{ij} represents the SP distance between i and j. If no path exists, $d_{ij} \to \infty$. The values are normalized by dividing by $(V-1)$.
- The eigenvector centrality e_i of a node i is determined by the principal eigenvector of the adjacency matrix:

$$\mathbf{Me} = \mu \mathbf{e}, \tag{8}$$

where $\mathbf{M}$ is the adjacency matrix, and μ is the largest eigenvalue. Normalization is achieved by dividing by the Euclidean norm $\|e\|_2$:

$$\|e\|_2 = \sqrt{\sum_{i=1}^{V} e_i^2}. \tag{9}$$

- Katz centrality z_i for node i incorporates both the number and the quality of connections:

$$z_i = \beta \mathbf{M} y_i + \gamma, \tag{10}$$

  Here, $\mathbf{M}$ is the adjacency matrix, and γ is a vector representing initial centrality values, usually set to $\mathbf{1}$. The attenuation factor β adjusts the influence of nearby versus distant nodes and must be less than the inverse of the largest eigenvalue of $\mathbf{M}$. Normalization uses the Euclidean norm $\|y\|_2$.
- The degree centrality g_i of a node i is its degree, defined as:

$$g_i = |\{(i,j) \in E : i \neq j\}| + |\{(j,i) \in E : i \neq j\}|, \tag{11}$$

  where (i,j) is a directed edge from i to j. Values are normalized by dividing by $2(V-1)$.
- The number of descendants D_i of node i represents the count of reachable nodes:

$$D_i = |\{j \in V \setminus \{i\} : i \to j\}|, \tag{12}$$

  Path $i \to j$ is calculated using the Breadth First Search (BFS) algorithm [37]. The values are normalized by dividing by $(V-1)$.

4.3. Density-Based Clustering Techniques

The use of centrality metrics to evaluate the significance of elements within a network (i.e., nodes in graph-based models) is a well-established method in both general epidemiology [38,39] and cybersecurity research applied to ICT networks [4,10,13,19]. Many of these studies [13,19,38,39] explore various strategies for achieving this through ranking based on individual centrality metrics, which often limits the scope of the analysis. Ref. [19] emphasizes the importance of combining multiple metrics into a unified ranking system to better prioritize safeguards in attack graphs. However, it stops short of exploring more advanced methods for effectively aggregating these metrics beyond simple weighting and combining individual rankings.

We based our proposal on the works in [4,10]. Identifying key nodes whose hardening can significantly delay an attacker's progress toward their objectives (i.e., the DA in the case of an AD attack graph) is feasible through the use of UL-based techniques, specifically clustering. Ref. [4] introduces a novel approach by applying density-based clustering on centrality metrics in AD authentication graphs between computers, allowing the aggregation of information from multiple metrics simultaneously. The proposal is grounded in the observation that centrality values, especially when considering multiple metrics per node, tend to show high concentrations in a small subset of nodes. In Section 5.1, we explore how this phenomenon also occurs in the metrics we calculate. This leads to the idea of identifying these potentially critical nodes as outliers (i.e., nodes that remain unclustered after applying density-based clustering) in an n-dimensional metric space. This aligns with the concept highlighted in [19] (p. 3), "highly ranked hosts are more likely to be used in an attack", but offers a more comprehensive perspective centered on identifying global maxima within the examined metric space. Our previous work on detecting superspreaders in non-targeted AD reachability graphs [10], which differs from our current focus on targeted attacks as explained in Section 3.2 further supports the validity of applying UL techniques to the problem at hand. Additionally, in this work, we explore performance not only in a global anomaly detection framework but also in a combined global and local context.

We have considered two different UL algorithms based on clustering techniques: Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [8] and a variant of DBSCAN called Hierarchical DBSCAN (HDBSCAN) [40]. Next, we briefly describe both clustering techniques.

The foundations of DBSCAN are based on the principle that for each node in a cluster, the neighborhood with radius ϵ must contain at least a minimum number of nodes $minsamples$. This is equivalent to stating that the cardinality of the neighborhood must exceed a certain threshold. Specifically, the concept of a cluster corresponds to the set of all nodes that are density-reachable from a core node within the cluster. DBSCAN examines the ϵ-neighborhood of each node in the graph, and if it contains more than $minsamples$ nodes, a new cluster is formed. It then checks the ϵ-neighborhood of all nodes that have not yet been processed. In our study, the neighborhood is defined within the 6-dimensional space determined by the metrics specified in Section 4.2. It is important to note that both ϵ and $minsamples$ are tunable hyperparameters. Given that our goal is to identify nodes with atypical centrality as candidates for safeguard application (i.e., outliers in the clustering), the $minsamples$ parameter will be the one we modify. The higher the desired number of outliers, the greater the value of the $minsamples$ should be.

On the other hand, HDBSCAN follows Hartigan's model of density–contour clusters and is capable of producing a complete clustering hierarchy that encompasses all possible DBSCAN-like clusterings for an infinite range of density thresholds. From this result, a simplified cluster tree can be easily extracted using Hartigan's concept of rigid clusters [41].

The hyperparameters of HDBSCAN are also tunable. In this case, the two main parameters are $minsamples$ and $minclustersize$. Once again, adjusting $minsamples$ will govern the number of outliers resulting from the clustering process, while $minclustersize$ determines the granularity of the clustering (i.e., the minimum size a cluster must reach to be recognized as such, rather than being merged with another existing bigger cluster).

Moreover, and crucial to our work, the density-based hierarchy produced by HDBSCAN can be utilized for outlier detection using Global–Local Outlier Scores from Hierarchies (GLOSH) [40]. In our research, we focused on using this algorithm specifically for outlier detection rather than relying on clustering results in HDBSCAN. Nevertheless, from now on, we refer to this approach as HDBSCAN-GLOSH, as its execution depends on HDBSCAN. Its distinctive features are as follows:

- The ability to consider both local and global outliers during detection. This allows for the identification of anomalous values in the 6-dimensional metrics space both unidimensionally and multidimensionally.
- The result of outlier detection is independent of the number and size of the resulting clusters. HDBSCAN-GLOSH thus provides a general scoring for all nodes in the graph based on their degree of anomaly (i.e., a score indicating how anomalous each node is).

This second aspect is what specifically motivates us to include HDBSCAN-GLOSH in our study as a UL-based alternative to DBSCAN. The resulting scoring offers a much more flexible framework for outlier detection since, when selecting candidate nodes for hardening, we can choose the desired number of nodes without being constrained by the number of nodes left unassigned to any cluster after detection. Once a specific node budget has been determined, this approach also allows for the optimal application or planning of countermeasures. Within the identified set of nodes, we retain an outlier score that enables us to rank them based on the priority of action, ensuring efficient resource allocation.

5. Experiments

Below, we detail the outcomes derived from the application of UL analysis on the DRGs extracted from the different graphs whose results we compared: RS, BB, BH, and AS (see Table 2). Finally, we present and compare the simulation results obtained after applying security safeguards based on the findings from this analysis.

Table 2. A summary of the number of nodes and edges per original graph and its associated DRG.

Graph	Original		DRG			
	#Nodes	#Edges	#Nodes	%	#Edges	%
RS	107,368	1,701,687	150	0.14%	1001	0.06%
BB	104,445	1,131,567	374	0.36%	3251	0.29%
AS	10,562	236,473	2474	23.42%	49,674	21.01%
BH	104,243	671,883	10,011	9.60%	53,737	8.00%

5.1. Experimental Settings

The first step involves calculating the centrality metrics mentioned in Section 4.2 for all nodes in each of the DRGs under study. To analyze the obtained values, we refer to Figure 2. This figure presents the descending progression of values for each metric across all nodes in each graph. It is important to note that, for better data visualization, the values shown for each metric are min-max normalized, and the progression of nodes on the X-axis is logarithmic. We observe that in all graphs, to a greater or lesser extent, a relatively small subset of nodes exhibits centrality values that significantly deviate from the overall trend in at least one or more of the calculated metrics.

This distribution of values is precisely what ensures that identifying nodes with atypical centrality (i.e., outliers in the clustering) will allow us to pinpoint network elements that play a critical role in network connectivity, as discussed in Section 4.3. Here, we have to demonstrate that the identified elements play a role, such that, when security safeguards are applied to them, we achieve a significant impact on the potential intrusions based on lateral movement, both in terms of delaying the expected time to reach the DA and, therefore, the risk to which it is subjected.

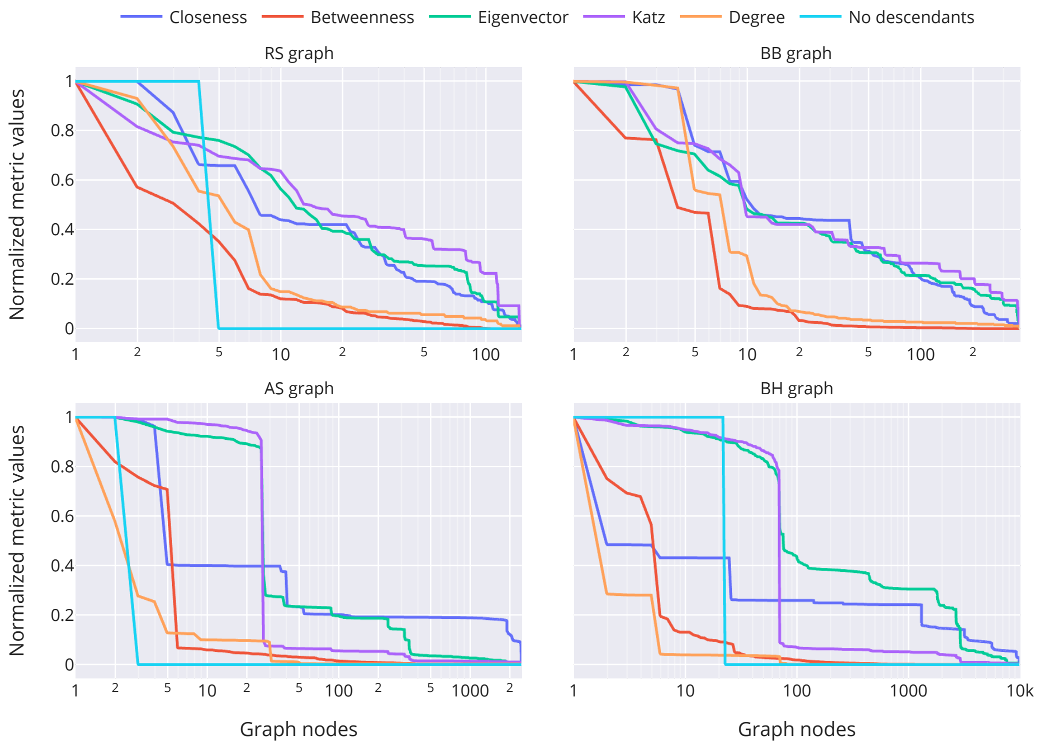

Figure 2. The distribution of centrality metrics along the DRG nodes.

Within the six-dimensional space of centrality metrics generated for each graph, we employed UL methods to identify anomalies. This approach involved the use of DBSCAN and HDBSCAN-GLOSH algorithms. To simplify the hyperparameter tuning and considering that we do not seek an excessively large number of candidate nodes for safeguard application (relative to the total number of nodes), we varied the possible values for the *minsamples* parameter in DBSCAN. For each graph, we selected the value that best suited our analysis based on the number of candidate nodes identified.

For HDBSCAN, the goal was to perform detection under conditions as similar as possible to those used in DBSCAN to allow for a fair comparison of the results obtained from both algorithms. Therefore, we determined the *minsamples* value analogously to the method used for DBSCAN in each case. As for the *minclustersize* parameter, we set it by default to be equal to the *minsamples* value.

As shown in Table 3, given the varying sizes of the DRGs associated with each network graph, and to ensure that we can analyze the results across a broader range of scenarios, we selected different *minsamples* values for each graph. This approach allowed us to generate cases with varying rates of candidate nodes for hardening while consistently maintaining a relatively small number of candidates in relation to the DRG node count.

Table 3. Overview of candidate node number per graph for applying security countermeasures.

Graph	#DRG Nodes	*Minsamples*	#Candidate Nodes	Candidates Rate
RS	150	5	13	8.67%
BB	374	5	19	5.08%
AS	2474	20	60	2.43%
BH	10,011	45	128	1.28%

With this in mind, the following scenarios were considered during the experimental phase for hardening–placement techniques aimed at mitigating lateral-movement-based threats:

- ***Baseline***: This scenario involves no hardening techniques. The infection process will proceed under the normal conditions inherent to the network graph being simulated. This scenario serves as a baseline to observe the impact of the various strategies applied in the subsequent scenarios on the infection process.
- ***ULdb***: In this scenario, hardening is applied to the nodes identified as candidates through the execution of the DBSCAN algorithm. Here, the candidate nodes are those not belonging to any of the identified clusters (i.e., those considered outliers by the clustering algorithm).
- ***ULgl***: In this scenario, hardening is applied to the nodes identified as candidates through the execution of the HDBSCAN-GLOSH algorithm. For each graph, if DBSCAN identifies n candidate nodes, HDBSCAN-GLOSH will identify the n nodes with the highest anomaly scores, ensuring that the candidate sets have equal cardinality in both cases.
- ***RNDngb***: In this scenario, hardening is applied to the nodes identified as candidates using the immunization algorithm proposed in [42], which favors the random selection of nodes with high degree centrality. As in the previous scenario, the number of candidate nodes selected using this strategy will match the number identified by DBSCAN for each graph. This scenario serves as a reference to evaluate whether the effect achieved by UL-based strategies represents a significant improvement. To ensure a fair comparison of results, random node selection will be performed on the DRGs, ensuring at least one path exists from any proposed node to the DA.

Given the characteristics of our simulator, as detailed in Section 3.1, it is important to note that throughout the entire simulation phase, there will be no practical difference between using the original network graphs or their corresponding DRGs. Since the simulations will start from the various entry points P identified for each graph and converge upon reaching the DA, any nodes in the original graphs that are not included in the associated DRG will not play a role in the simulated infection process. To optimize resource usage, such as memory during simulations, we will conduct the simulations on the DRGs extracted from each original graph. However, whether we use the original graphs or the DRGs will not affect the outcomes of the simulations.

To measure the effectiveness of the safeguards applied to the identified nodes, we introduce a TTC-based metric called MTCDA. It is important to remember that the simulations involve a sequence of infection events that unfold over a certain period until the DA is compromised, meaning the attacker has traversed the DRG from start to finish. The simulation ends at this point. The critical data point is when the DA is compromised or, in other words, when the simulation concludes. Naturally, our objective is to delay this moment as much as possible by implementing countermeasures. Thus, the MTCDA represents the median time it takes for the attacker to reach the DA across all iterations of the simulation. The number of iterations performed by the simulator will be determined by three factors:

- **Entry Points (P)**: We conducted multiple simulations, initiating the infection process from each of the entry points $p \in P$ that were initially identified for each graph. The count of these entry points can be found in Table 4.

Table 4. Number of nodes acting as entry points P for each graph.

Graph	RS	BB	AS	BH
$\lvert P\rvert$	9	1	3	6

- **Hardening–placement strategy**: We carried out various simulations where hardening measures were applied to mitigate lateral movement on different sets of nodes identified through the strategies described earlier (*ULdb*, *ULgl*, and *RNDngb*), as well as a scenario where no mitigation strategy was applied (*Baseline*).
- **Equal setup simulations**: Given specific initial conditions (i.e., a particular infection source node and a selected mitigation strategy, if any), and to account for the inherent randomness of the stochastic process governing the simulations, we performed 500 iterations of the simulator under these conditions. This approach ensures that the confidence intervals (CIs) for the MTCDA are precise enough.

Taking all of this into account, the total number of simulator runs I performed for each graph is given by the following expression:

$$I = \lvert P\rvert \times 4 \times 500 \tag{13}$$

where $\lvert P\rvert$ represents the cardinality of the set of entry points, the factor 4 corresponds to the number of hardening–placement strategies applied, and the factor 500 accounts for the number of iterations with the same initial setup in each case. To execute all these simulations, we used multi-threaded Python 3.10 code (optimized with Numpy and Numba), running on an Intel Xeon server with 48 cores and 128 GB of RAM.

Finally, we need to clarify how we model the application of security countermeasures (i.e., hardening) to the various network nodes identified through the aforementioned strategies. We model this by introducing a mitigation factor α that takes effect whenever the infection attempts to reach a node designated for hardening. This factor α implies that each time the infection process tries to reach one of these nodes, the expected average time for that transition to occur (derived from the mean λ^{-1} of the probability distribution $X \sim \mathrm{Exp}(\lambda)$ governing the time between infections) will be increased by a factor of α (i.e, redefining the distribution mean as $\lambda = \alpha\tau^{-1}$). Without loss of generality, we set a uniform value of $\tau = 50\%$ for all edges in the graph and a value of $\alpha = 10$ for the incoming edges to hardened nodes. It is important to emphasize that our model aims to establish a framework for estimating the impact on the compromise time of the hardened nodes. This will enable us to assess the area of the network that we can secure through the application of countermeasures. However, a quantitative analysis of the specific impact of implementing these countermeasures remains to be conducted.

5.2. Experimental Evaluation

In this section, we present the experimental results obtained from applying the proposed methodology developed throughout this work, following the preliminary considerations detailed in Section 5.1.

We focus on the results derived from the TTC-based evaluation we propose, which aims to determine the impact of hardening by implementing security countermeasures at specific points in the network identified via UL techniques (i.e., DBSCAN and HDBSCAN-GLOSH) to mitigate lateral-movement-based threats. This mitigation is reflected in the delay imposed on potential attackers in reaching the organization's critical assets (i.e., the DA or Domain node), assuming that they initiate the compromise by lateral movement from the farthest point in the network from which this objective can be achieved.

The objective is to make it as difficult as possible for attackers to traverse the network when attempting to compromise the entire affected AD environment. This approach not only reduces the likelihood of a successful attack but also extends the available reaction time for incident response teams to take reactive measures, if necessary, before the network is fully compromised.

Figure 3 illustrates the distribution of time values (TTC-DA) that attackers, modeled as an infection process, took in each of the proposed scenarios to reach the DA, across all iterations conducted by the simulator. The figure also highlights the median value of these times (MTCDA) in each scenario. Note that the time in these figures is dimensionless, as we are focused on a consistent comparison among techniques.

At first glance, it is evident that the *ULdb* strategy yields outstanding results, not only significantly outperforming the baseline scenario but also surpassing the *RNDngb* strategy, which serves as a reference. A deeper analysis of these results can be found in Table 5, where the median values (MTCDA) for each scenario are presented along with the corresponding CI-95%. Additionally, the gain indicator $\mathcal{G}_{ULdb}$ is defined as the ratio between the MTCDA using the *ULdb* strategy and the MTCDA for the baseline scenario where no hardening has been applied to the network. For simplicity, this gain is not calculated for the other scenarios since *ULdb* clearly stands out as the most advantageous.

Table 5. MTCDA and CI-95% (lower and upper bounds) for the different graphs and hardening–placement strategies.

	Graph											
	RS			**BB**			**AS**			**BH**		
	Med	**CI$_{lo}$**	**CI$_{up}$**	**Med**	**CI$_{lo}$**	**CI$_{up}$**	**Med**	**CI$_{lo}$**	**CI$_{up}$**	**Med**	**CI$_{lo}$**	**CI$_{up}$**
Baseline	10.47	0.144	0.178	20.10	0.671	0.909	2.43	0.060	0.057	8.39	0.122	0.127
ULdb	41.87	1.214	1.195	144.60	6.751	5.116	4.65	0.095	0.111	21.91	0.326	0.234
ULgl	21.04	0.475	0.449	69.93	3.149	3.432	4.04	0.092	0.076	11.63	0.138	0.140
RNDngb	24.98	0.884	0.882	81.66	4.442	3.344	4.32	0.092	0.085	15.24	0.296	0.260
$\mathcal{G}_{ULdb}$	3.9970	-	-	7.1925	-	-	1.9168	-	-	2.6101	-	-

The results show that the expected time for attackers to reach the DA, after applying countermeasures to the nodes identified through DBSCAN, increases by approximately 2 to 7 times, depending on the graph. Furthermore, a clear trend is observed: graphs where attackers would naturally take longer to reach the DA due to the network's inherent topological characteristics are the cases where the gain is even greater. This makes logical sense, as the more of the network the attacker must traverse through lateral movement, the more likely they are to encounter the points where we have implemented the appropriate countermeasures (i.e., the application of safeguards according to our proposal has a broader effective action area).

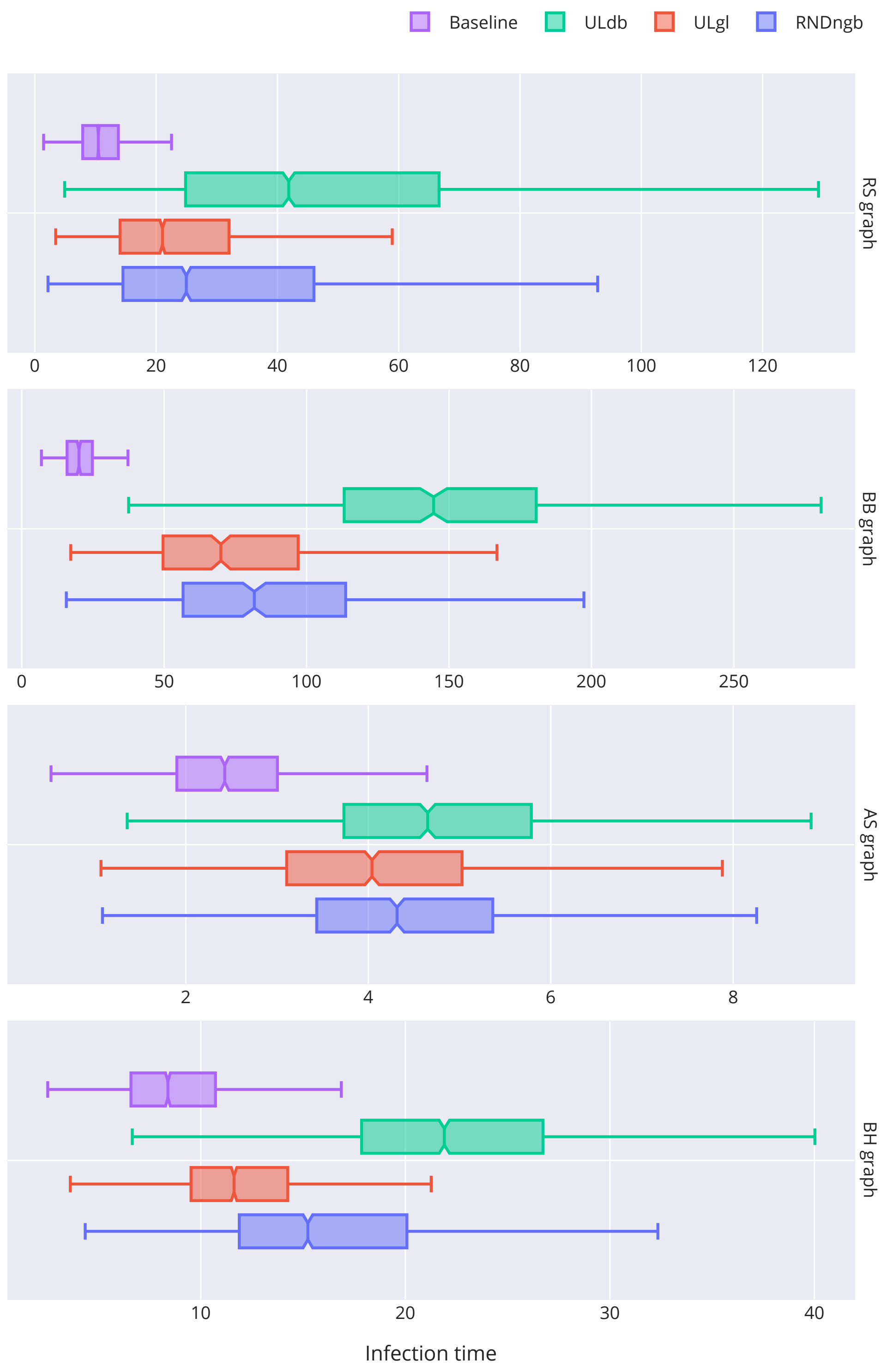

Figure 3. TTC-DA for different hardening–placement techniques.

On the other hand, we observed less favorable results with our secondary proposal, which used the *ULgl* strategy for hardening–placement. While the MTCDA did increase compared to the scenario without countermeasures, the improvement was not significant. In fact, the *RNDngb* strategy showed greater benefits in all study cases. The goal of applying the *ULgl* strategy, based on HDBSCAN-GLOSH, was to explore an alternative to *ULdb* that could provide a general anomaly score for all nodes in the graph. This would allow for flexible selection of the number of candidate nodes for hardening. Although the initial results did not meet expectations, we hypothesize that the issue may stem from incorrect hyperparameter tuning. This would imply that although HDBSCAN-GLOSH accounts for both global and local outliers, this distinction from DBSCAN is not the cause of the problem. This makes sense, as DBSCAN effectively identifies nodes crucial to network connectivity across multiple factors in the six-dimensional centrality metric space. However, a node's local importance in just one of these factors might also indicate its relevance.

We then assessed whether varying HDBSCAN-GLOSH hyperparameters could improve the performance of the *ULgl* strategy on each graph. The basic hyperparameters for HDBSCAN-GLOSH were *minsamples* and *minclustersize*. By default, we set both values to be equal, but this time we decoupled them from the *minsamples* parameter in DBSCAN.

We kept the other hardening–placement strategies (*baseline*, *ULdb*, and *RNDngb*) unchanged. We then redefined a new GLOSH-based strategy called *ULgl′*. For this strategy, the execution of HDBSCAN-GLOSH remained the same; however, we adjusted the reference values of its hyperparameters and recalculated the outlier scores. Since our goal was to match the results of applying *ULgl′* with those of *ULdb*, we selected the parameter values to identify a set of nodes for each graph with the highest possible overlap with the set previously identified by DBSCAN. Table 6 shows the parameter values used for this configuration, as well as the number and proportion of nodes identified that coincide between both strategies (i.e., *ULdb* and *ULgl′*).

Table 6. Overview of #nodes per graph identified by both DBSCAN and HDBSCAN-GLOSH, and their rate relative to DBSCAN's original candidates.

Graph	#Candidate Nodes	*Ulgl′* Hyperparameters	#Coincident Nodes	Overlap Rate
RS	13	10	12	92.31%
BB	19	25	18	94.74%
AS	60	500	30	50%
BH	128	1750	29	22.66%

In Figure 4 and Table 7, we present the extended results for each graph, including the new *ULgl′* strategy. Since the new *ULgl′* parameters were obtained by seeking to maximize node overlap with *ULdb*, it is expected that cases with higher overlap rates (i.e., RS and BB) show closer results between both strategies. Conversely, cases with lower overlap (i.e., AS and BH) are more likely to produce differing results.

Table 7. MTCDA and CI-95% (lower and upper bounds) for the different graphs and hardening–placement strategies, including *ULgl* hyperparameter variation (*ULgl′*).

	Graph											
	RS			BB			AS			BH		
	Med	CI_{lo}	CI_{up}	Med	CI_{lo}	CI_{up}	Med	CI_{lo}	CI_{up}	Med	CI_{lo}	CI_{up}
Baseline	10.47	0.144	0.178	20.10	0.671	0.909	2.43	0.060	0.057	8.39	0.122	0.127
ULdb	41.87	1.214	1.195	144.60	6.751	5.116	4.65	0.095	0.111	21.91	0.326	0.234
ULgl	21.04	0.475	0.449	69.93	3.149	3.432	4.04	0.092	0.076	11.63	0.138	0.140
ULgl′	41.39	1.260	1.122	136.07	4.913	6.859	9.93	0.234	0.300	14.90	0.224	0.204
RNDngb	24.98	0.884	0.882	81.66	4.442	3.344	4.32	0.092	0.085	15.24	0.296	0.260
$\mathcal{G}_{ULdb}$	3.9970	-	-	7.1925	-	-	1.9168	-	-	2.6101	-	-
$\mathcal{G}_{ULgl'}$	3.9513	-	-	6.7679	-	-	4.0911	-	-	1.7753	-	-

Overall, we observed a positive effect. With a simple adjustment to HDBSCAN-GLOSH hyperparameters, the TTC-DA distribution for *ULgl′* and the median value MTCDA increased significantly in three out of the four scenarios studied. This effect can be summarized by noting that the gain from this new strategy ($\mathcal{G}_{ULgl'}$) is nearly identical to that originally achieved by DBSCAN ($\mathcal{G}_{ULdb}$) for both the RS and BB graphs, with MTCDA increasing nearly 4 and 7 times, respectively, compared to the scenario without countermeasures. This was expected since the set of nodes to harden under both strategies overlapped nearly 100% for these graphs (see Table 6).

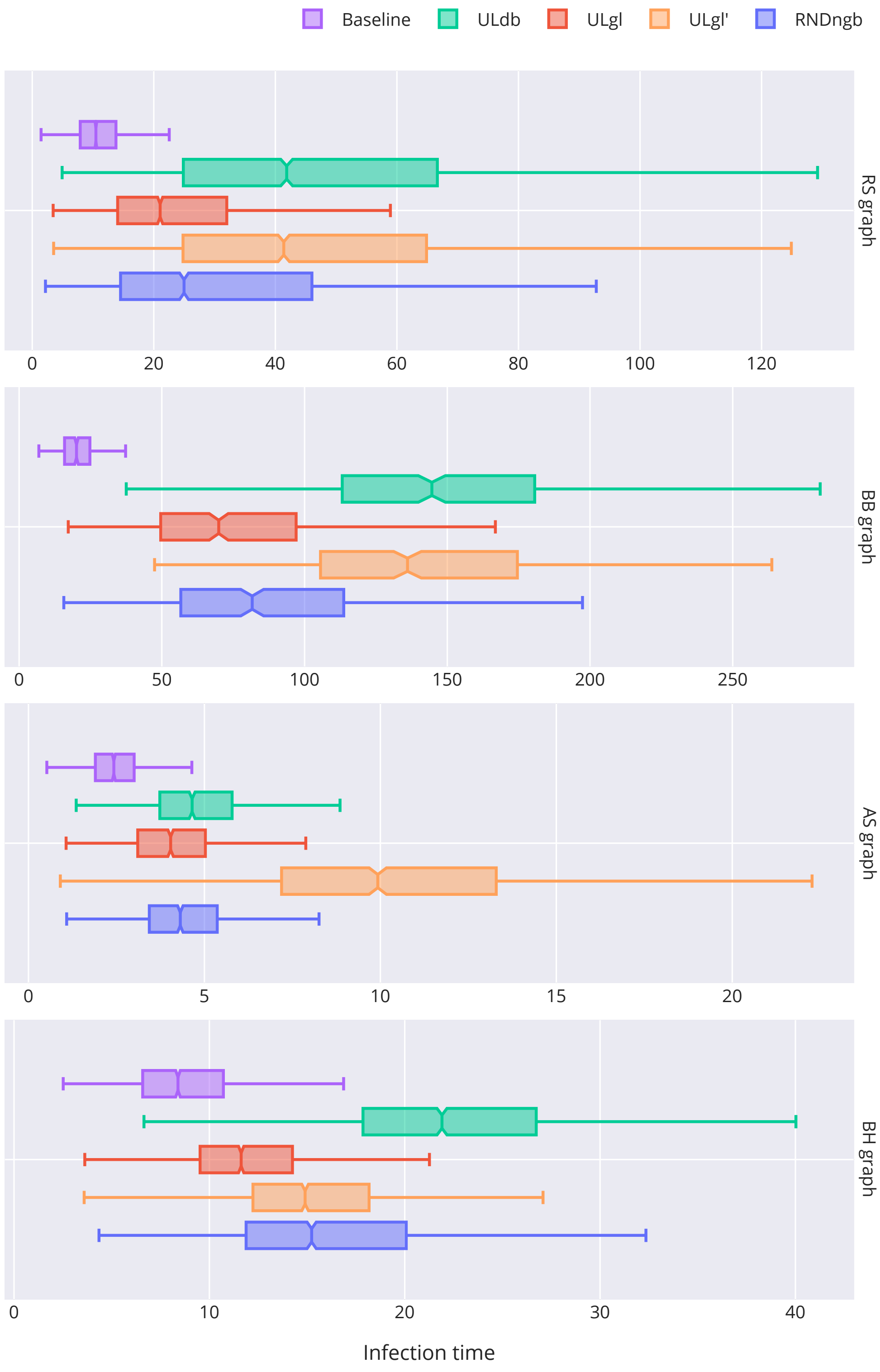

Figure 4. TTC-DA for different hardening–placement techniques, including *ULgl* hyperparameter variation (*ULgl′*).

Additionally, we explore these results further in Figure 5, which shows, for all graphs, the histogram of outlier scores given by HDBSCAN-GLOSH segmented into 20 equal-sized bins, alongside a complementary CDF plot (i.e., CCDF or $1 - \text{CDF}$) illustrating the ratio of nodes whose outlier score is equal to or higher than a given value. We observe that for the RS graph, the number of nodes is relatively small at high scores, gradually increasing as lower scores are considered. This suggests that there are no significant concentrations of nodes in any high sub-range of outlier scores. In a less regular and gradual manner, we observe a similar trend in the BB graph, too. Thus, by applying the *ULgl′* strategy and selecting the top percentage of nodes (i.e., *Candidates Rate*) with the highest outlier scores, we effectively capture a homogeneous set of the most relevant nodes for hardening, which is almost identical to that obtained by *ULdb*.

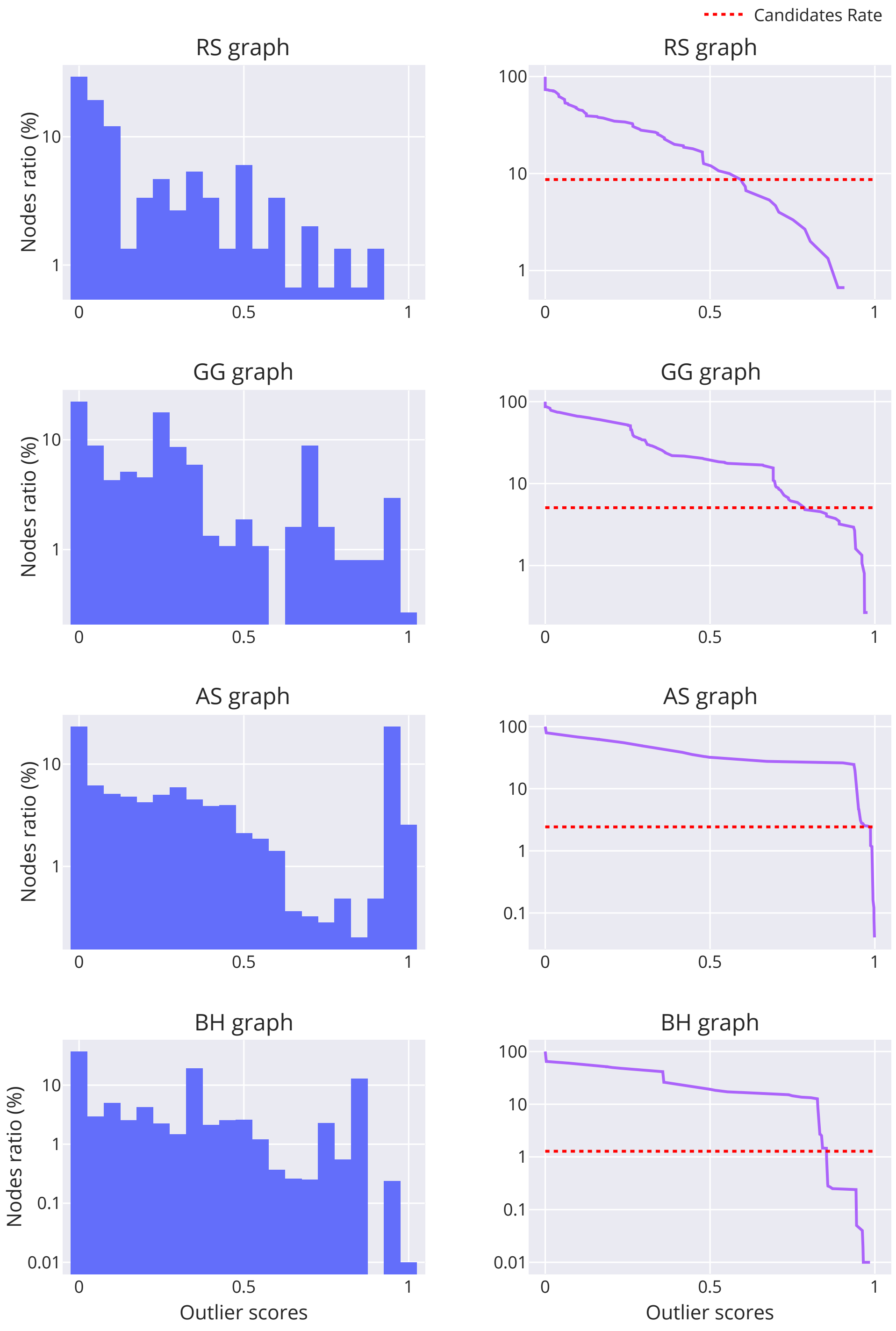

Figure 5. Histogram (**left**) and CCDF (**right**) of outlier scores given by HDBSCAN-GLOSH for each graph.

In the case of the AS graph, the overlap between the two strategies was only 50%, resulting in more unequal outcomes. Notably, the *ULgl′* strategy not only matched but significantly exceeded the results obtained by *ULdb*, increasing the MTCDA from approximately 2% to over 4%. Upon examining Figure 5, we observe a distinct trend compared to previous findings. A considerable proportion of the outlier scores given by HDBSCAN-GLOSH are concentrated in the higher ranges. This is particularly evident when considering the top decile (D_9) of the score values, which stands at 0.946 for the AS graph, whereas for the RS, BB, and BH graphs, its value is 0.56, 0.698, and 0.827, respectively. This suggests that by employing the *ULgl′* strategy, we successfully captured a subset of nodes exhibiting markedly anomalous centrality, thus highlighting their critical importance for being hardened. Furthermore, focusing on the differences between the sets of nodes identified by *ULdb* and *ULgl′*, we note that when using DBSCAN, a particularly high *minsamples* value was not configured (i.e., 20, as indicated in Table 3). This may have prevented it from considering "doubtful" outliers. Returning to Figure 5, we observe that the percentage of nodes designated as candidates (i.e., *Candidates Rate*) excludes many other nodes with similarly high scores that are close to those selected. This might explain why *ULdb* identified a distinct subset of nodes whose hardening also offers significant benefits, despite not being as pronounced as those achieved by *ULgl′*.

In the case of the BH graph, we observe that the hyperparameter adjustments applied to *ULgl′* have only a marginal impact, resulting in a slight improvement that merely brings its performance on par with the *RNDngb* strategy, which remains significantly inferior to that of *ULdb*. Notably, the node set identified by DBSCAN in this scenario required a substantially higher *minsamples* parameter to reach the desired cardinality (1.28% of DRG nodes), as compared to the other graphs (i.e., 45, as shown in Table 3). This suggests that there may be a relatively low proportion (relative to the large size of the DRG, as seen in Table 2) of strongly anomalous nodes within the clustered metric space, complicating DBSCAN's ability to detect significant outliers when *minsamples* is low. Figure 5 further supports this, as the difficulty in detecting pronounced anomalies is also evident for HDBSCAN-GLOSH. There are few nodes with high scores (i.e., 0.9 or above), and only below the threshold of 0.85 does the node count increase considerably. Furthermore, when evaluating the *Candidates Rate* threshold cutoff point, we observe that it excludes in the score range $(0.8, 0.85)$ a significant number of nodes with scores very close to those selected. Given the large size of the BH DRG graph, which increases the need for accurate node selection, these factors force *ULgl′* to identify a set of less strongly outlier nodes, making the selection less reliable. Consequently, *ULdb* proves to be a more consistent and effective strategy for selecting nodes to harden in this case.

In summary, we observe that tuning the HDBSCAN-GLOSH hyperparameters does not necessarily provide a viable alternative to DBSCAN. Our analysis shows that in cases like the RS and BB graphs, where the subgraph of nodes capable of reaching the domain (i.e., DRG) is smaller and centrality distribution results in a progressively lower presence of outlier nodes as they become more anomalous, both hardening–placement strategies yield similar results. In cases where there is a significant presence of strongly outlier nodes, such as the AS graph, HDBSCAN-GLOSH has demonstrated its ability to capture them more extensively and reliably than DBSCAN, significantly surpassing its outcomes. Conversely, in cases like the BH graph, which exhibits a very low presence of strongly outlier nodes, HDBSCAN-GLOSH's results fall short of those achieved by DBSCAN.

Given these findings, it becomes clear that the morphology and centrality distribution of the graph play a crucial role in determining which algorithm is more effective. Further exploration across a broader range of AD attack graphs with varying structures is essential to fully understand the conditions under which DBSCAN is suitable, as well as under which HDBSCAN-GLOSH can reach or even outperform it. However, it is evident that while tuning DBSCAN's *minsamples* parameter adjusts the number of identified nodes in a somewhat uncontrolled manner, the proper tuning of HDBSCAN-GLOSH hyperparameters (without needing to control the number of identified nodes) remains essential for accurate anomaly detection.

6. Conclusions

Over time, industries and other organizations have increasingly been affected not only by the growing amount of cyber threats globally but also by the continuous advancement in the complexity and professionalization of these threats. One of the most prevalent techniques employed by these cyber threats, and whose mitigation is crucial, is lateral movement. In this study, we explore threat mitigation possibilities through the application of ML (specifically UL), focusing on network infrastructures orchestrated by Microsoft AD, one of the most widely used technologies in the world, for this purpose.

We defined DRGs that isolate the dynamics of lateral movement propagation, targeting the most critical asset of AD networks, the DA. Various widely used centrality metrics were calculated on these subgraphs, allowing us to apply density-based clustering (DBSCAN) on the resulting sets. This approach identified anomalous nodes where applying security countermeasures could significantly slow lateral movement before the domain is compromised. Additionally, we applied the HDBSCAN-GLOSH algorithm to attempt to match, or even surpass, the results obtained with DBSCAN by providing a continuous anomaly score for all nodes, enabling the flexible determination of the number of network points to harden.

Our analysis was conducted on four AD graphs: one extracted from real, anonymized infrastructure and three others generated synthetically utilizing widely used tools (BadBlood, ADSimulator, and BloodHoundDBCreator). The application of DBSCAN yielded significantly positive results. Depending on the graph and a given delay factor α (used to model node hardening), we were able to delay the compromise of DA by up to seven times compared to scenarios without countermeasures. We also found that HDBSCAN-GLOSH, depending closely on the characteristics of the analyzed graph and its proper hyperparameter tuning, can achieve results similar to or even surpassing those of DBSCAN while also providing the advantage of specifying the desired number of nodes to harden.

Despite these positive outcomes, our research remains open to further advancements. The main ones are outlined below:

- We aim to explore the optimal hyperparameter tuning characteristics for DBSCAN and HDBSCAN-GLOSH to achieve the best possible results, as well as to include a broader and more diverse range of AD graphs within the scope of our study to validate them. This includes the addition of more real attack graphs whenever possible, as well as the use of state-of-the-art synthetic generation tools like ADSynth [43].
- We plan to analyze the performance of these proposals as we adjust the budget for the number of nodes to be hardened, as well as the selection of centrality metrics used, considering both their quantity and complexity.
- We aim to conduct a quantitative analysis of the exploitability of each type of edge within the AD graph. This will enable us to estimate the exploitation difficulty associated with each edge, as well as the effort and impact involved in implementing specific countermeasures for each case.
- We will explore other UL-based clustering algorithms, including classical methods like OPTICS (Ordering Points to Identify the Clustering Structure) [44], as well as more recent ones as proposed in [45]. Additionally, we will examine other anomaly detection techniques, such as Isolation Forest [46], which is particularly suited for outlier detection.
- In addition to exploring a broader range of anomaly detection techniques (both clustering-based and otherwise), we aim to enhance our research by establishing new baselines for evaluating future proposals. To achieve this, we will explore the applicability of untapped solutions in AD security from other fields of study, such as those based on SHS [17,18].

- We are also considering the possibility of developing a general risk reduction framework that integrates our study on identifying key nodes for attackers to reach AD administration (escalators and gatekeepers) with the work in [10] on identifying generic superspreaders in AD infrastructures.

Additionally, this research could extend even further by generalizing analyses to apply to more types of logical graphs from not only AD-based infrastructures [14]. Moreover, we are considering including dynamic graphs [33,47] in our research scope, which can represent the reaction dynamics of incident response teams when a threat is being mitigated.

Author Contributions: Conceptualization, D.H.-O. and M.T.-R.; data curation, D.H.-O.; formal analysis, D.H.-O., M.T.-R. and J.M.G.-G.; funding acquisition, J.M.G.-G. and L.C.-P.; investigation, D.H.-O., M.T.-R., J.M.G.-G. and L.C.-P.; methodology, D.H.-O., M.T.-R., J.M.G.-G. and L.C.-P.; project administration, J.M.G.-G.; resources, D.H.-O., M.T.-R. and L.C.-P.; software, D.H.-O. and L.C.-P.; supervision, M.T.-R. and J.M.G.-G.; validation, D.H.-O., M.T.-R., J.M.G.-G. and L.C.-P.; visualization, D.H.-O., M.T.-R. and J.M.G.-G.; writing—original draft, D.H.-O., M.T.-R. and J.M.G.-G.; writing—review and editing, D.H.-O., M.T.-R. and J.M.G.-G. All authors have read and agreed to the published version of the manuscript.

Funding: This publication is part of project TED2021-131387B-I00 funded by MCIN/AEI/10.13039/501100011033 and by the European Union "NextGenerationEU"/PRTR and of project PID2021-123168NB-I00 funded by MCIN/AEI/10.13039/501100011033/FEDER, UE. Finally, this work is a part of the research project SBPLY/23/180225/000160, which is funded by the EU through FEDER, Spain, and by the JCCM through INNOCAM. David Herranz is also funded by both an FPU grant and a Mobility Grant for Research Staff in Training from the University of Alcalá.

Data Availability Statement: The datasets presented in this article are not readily available because data from real network infrastructures are confidential. Requests to access the other datasets should be directed to the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AD	Active Directory
APB	Attack-Path-Based
APT	Advanced Persistent Threat
BFS	Breadth First Search
(C)CDF	(Complementary) Cumulative Distribution Function
CI	Confidence Interval
CTMC	Continuous-Time Markov Chain
DAG	Directed Acyclic Graph
DA	Domain Administration
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DC	Domain Controller
DES	Discrete Event Simulator
DRG	Domain Reachability Graph
EDO	Evolutionary Diversity Optimization
FTP	Fixed-Parameter Tractable
GLOSH	Global–Local Outlier Score from Hierarchies
GNN	Graph Neural Network
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise
ICT	Information and Communication Technology
ML	Machine Learning
MTCDA	Median Time to Compromise Domain Administration
OPTICS	Ordering Points to Identify the Clustering Structure
RaaS	Ransomware as a service
RL	Reinforcement Learning
SHS	Structural Hole Spanners

SI	Susceptible–Infected
SIR	Susceptible–Infected–Recovered
SP	Shortest Path
TTC	Time to Compromise
UL	Unsupervised Learning

References

1. Grillenmeier, G. Now's the time to rethink Active Directory security. *Netw. Secur.* **2021**, *2021*, 13–16. [CrossRef]
2. Cybersecurity and Infrastructure Security Agency. #StopRansomware: Black Basta (AA24-131A). 2024. Available online: https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-131a (accessed on 29 August 2024).
3. Cybersecurity and Infrastructure Security Agency. #StopRansomware: Akira Ransomware (AA24-109A). 2024. Available online: https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-109a (accessed on 29 August 2024).
4. Powell, B.A. The epidemiology of lateral movement: Exposures and countermeasures with network contagion models. *J. Cyber Secur. Technol.* **2020**, *4*, 67–105. [CrossRef]
5. Lambert, J. Defenders Think in Lists. Attackers Think in Graphs. As Long as This Is True, Attackers Win. 2015. Available online: https://perma.cc/6NZ2-A2HY (accessed on 19 June 2023).
6. Guo, M.; Li, J.; Neumann, A.; Neumann, F.; Nguyen, H. Practical fixed-parameter algorithms for defending active directory style attack graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, pp. 9360–9367.
7. Dunagan, J.; Zheng, A.X.; Simon, D.R. Heat-ray: Combating identity snowball attacks using machinelearning, combinatorial optimization and attack graphs. In Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles, Big Sky, MT, USA, 11–14 October 2009; pp. 305–320.
8. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the KDD, Portland, ON, USA, 2–4 August 1996; Volume 96, pp. 226–231.
9. Campello, R.J.; Moulavi, D.; Sander, J. Density-based clustering based on hierarchical density estimates. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Gold Coast, Australia, 14–17 April 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 160–172.
10. Herranz-Oliveros, D.; Marsa-Maestre, I.; Gimenez-Guzman, J.M.; Tejedor-Romero, M.; de la Hoz, E. Surgical immunization strategies against lateral movement in Active Directory environments. *J. Netw. Comput. Appl.* **2024**, *222*, 103810. [CrossRef]
11. Powell, B.A. Role-based lateral movement detection with unsupervised learning. *Intell. Syst. Appl.* **2022**, *16*, 200106. [CrossRef]
12. Bowman, B.; Laprade, C.; Ji, Y.; Huang, H.H. Detecting Lateral Movement in Enterprise Computer Networks with Unsupervised Graph AI. In Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020), San Sebastian, Spain, 14–15 October 2020; USENIX Association: Berkeley, CA, USA, 2020; pp. 257–268.
13. Elmiger, M.; Lemoudden, M.; Pitropakis, N.; Buchanan, W.J. Start thinking in graphs: Using graphs to address critical attack paths in a Microsoft cloud tenant. *Int. J. Inf. Secur.* **2024**, *23*, 467–485. [CrossRef]
14. Marsa-Maestre, I.; Gimenez-Guzman, J.M.; Orden, D.; de la Hoz, E.; Klein, M. REACT: Reactive resilience for critical infrastructures using graph-coloring techniques. *J. Netw. Comput. Appl.* **2019**, *145*, 102402. [CrossRef]
15. Chen, E.; Lockey, S.; Khosravi, H.; Baghaei, N. Enhancing Cybersecurity through Machine Learning-Driven Anomaly Detection Systems. *J. Artif. Intell. Res. Appl.* **2024**, *4*, 123–135.
16. Sarker, I.H. Learning Technologies: Toward Machine Learning and Deep Learning for Cybersecurity. In *AI-Driven Cybersecurity and Threat Intelligence: Cyber Automation, Intelligent Decision-Making and Explainability*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 43–59.
17. Goel, D.; Shen, H.; Tian, H.; Guo, M. Discovering Top-k Structural Hole Spanners in Dynamic Networks. *arXiv* **2023**, arXiv:2302.13292.
18. Goel, D.; Shen, H.; Tian, H.; Guo, M. Effective graph-neural-network based models for discovering Structural Hole Spanners in large-scale and diverse networks. *Expert Syst. Appl.* **2024**, *249*, 123636. [CrossRef]
19. Hong, J.B.; Kim, D.S. Scalable security analysis in hierarchical attack representation model using centrality measures. In Proceedings of the 2013 43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop (DSN-W), Budapest, Hungary, 24–27 June 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1–8.
20. He, Z.Y.; Abbes, A.; Jahanshahi, H.; Alotaibi, N.D.; Wang, Y. Fractional-order discrete-time SIR epidemic model with vaccination: Chaos and complexity. *Mathematics* **2022**, *10*, 165. [CrossRef]
21. Thomas, D.M.; Sturdivant, R.; Dhurandhar, N.V.; Debroy, S.; Clark, N. A Primer on COVID-19 Mathematical Models. *Obesity* **2020**, *28*, 1375. [CrossRef]
22. Raponi, S.; Khalifa, Z.; Oligeri, G.; Di Pietro, R. Fake news propagation: A review of epidemic models, datasets, and insights. *ACM Trans. Web (TWEB)* **2022**, *16*, 1–34. [CrossRef]
23. Hosseini, S.; Zandvakili, A. Information dissemination modeling based on rumor propagation in online social networks with fuzzy logic. *Soc. Netw. Anal. Min.* **2022**, *12*, 34. [CrossRef]
24. Bucci, A.; La Torre, D.; Liuzzi, D.; Marsiglio, S. Financial contagion and economic development: An epidemiological approach. *J. Econ. Behav. Organ.* **2019**, *162*, 211–228. [CrossRef]

25. Kiss, I.Z.; Miller, J.C.; Simon, P.L. *Mathematics of Epidemics on Networks*; Springer: Cham, Switzerland, 2017; Volume 598, p. 31.
26. Miller, J.C.; Ting, T. EoN (Epidemics on Networks): A fast, flexible Python package for simulation, analytic approximation, and analysis of epidemics on networks. *J. Open Source Softw.* **2019**, *4*, 1731. [CrossRef]
27. Goel, D.; Neumann, A.; Neumann, F.; Nguyen, H.; Guo, M. Evolving Reinforcement Learning Environment to Minimize Learner's Achievable Reward: An Application on Hardening Active Directory Systems. *arXiv* **2023**, arXiv:2304.03998.
28. Goel, D.; Moore, K.; Guo, M.; Wang, D.; Kim, M.; Camtepe, S. Optimizing Cyber Defense in Dynamic Active Directories through Reinforcement Learning. In Proceedings of the European Symposium on Research in Computer Security, Bydgoszcz, Poland, 16–20 September 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 332–352.
29. Goel, D.; Ward, M.; Neumann, A.; Neumann, F.; Nguyen, H.; Guo, M. Hardening Active Directory Graphs via Evolutionary Diversity Optimization based Policies. *ACM Trans. Evol. Learn.* **2024**. [CrossRef]
30. Goel, D.; Ward-Graham, M.H.; Neumann, A.; Neumann, F.; Nguyen, H.; Guo, M. Defending active directory by combining neural network based dynamic program and evolutionary diversity optimisation. In Proceedings of the Genetic and Evolutionary Computation Conference, Boston, MA, USA, 9–13 July 2022; pp. 1191–1199.
31. Guo, M.; Ward, M.; Neumann, A.; Neumann, F.; Nguyen, H. Scalable edge blocking algorithms for defending active directory style attack graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 5649–5656.
32. Ngo, H.; Guo, M.; Nguyen, H. Optimizing cyber response time on temporal active directory networks using decoys. In Proceedings of the Genetic and Evolutionary Computation Conference, Melbourne, VIC, Australia, 14–18 July 2024; pp. 1309–1317.
33. Ngo, H.Q.; Guo, M.; Nguyen, H. Near Optimal Strategies for Honeypots Placement in Dynamic and Large Active Directory Networks. In Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, London, UK, 29 May–2 June 2023; pp. 2517–2519.
34. Dias, J. *A Guide to Microsoft Active Directory (AD) Design*; Technical Report; Lawrence Livermore National Lab. (LLNL): Livermore, CA, USA, 2002.
35. Kang, H.; Liu, B.; Mišić, J.; Mišić, V.B.; Chang, X. Assessing security and dependability of a network system susceptible to lateral movement attacks. In Proceedings of the 2020 International Conference on Computing, Networking and Communications (ICNC), Big Island, HI, USA, 17–20 February 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 513–517.
36. He, D.; Gu, H.; Zhu, S.; Chan, S.; Guizani, M. A Comprehensive Detection Method for the Lateral Movement Stage of APT Attacks. *IEEE Internet Things J.* **2023**, *11*, 8440–8447. [CrossRef]
37. Lawande, S.R.; Jasmine, G.; Anbarasi, J.; Izhar, L.I. A systematic review and analysis of intelligence-based pathfinding algorithms in the field of video games. *Appl. Sci.* **2022**, *12*, 5499. [CrossRef]
38. Sartori, F.; Turchetto, M.; Bellingeri, M.; Scotognella, F.; Alfieri, R.; Nguyen, N.K.K.; Le, T.T.; Nguyen, Q.; Cassi, D. A comparison of node vaccination strategies to halt SIR epidemic spreading in real-world complex networks. *Sci. Rep.* **2022**, *12*, 21355. [CrossRef]
39. Rodrigues, F.A. Network centrality: An introduction. In *A Mathematical Modeling Approach from Nonlinear Dynamics to Complex Systems*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 177–196.
40. Campello, R.J.; Moulavi, D.; Zimek, A.; Sander, J. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discov. Data (TKDD)* **2015**, *10*, 1–51. [CrossRef]
41. Hartigan, J.A. *Clustering Algorithms*, 99th ed.; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1975.
42. Cohen, R.; Havlin, S.; Ben-Avraham, D. Efficient immunization strategies for computer networks and populations. *Phys. Rev. Lett.* **2003**, *91*, 247901. [CrossRef] [PubMed]
43. Nguyen, N.L.; Falkner, N.; Nguyen, H. ADSynth: Synthesizing Realistic Active Directory Attack Graphs. In Proceedings of the 2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Brisbane, Australia, 24–27 June 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 66–74.
44. Ankerst, M.; Breunig, M.; Kriegel, H.P.; Ng, R.; Sander, J. Ordering points to identify the clustering structure. In Proceedings of the ACM SIGMOD, Vancouver, BC, Canada, 10–12 June 2008; Volume 99.
45. Shojafar, M.; Taheri, R.; Pooranian, Z.; Javidan, R.; Miri, A.; Jararweh, Y. Automatic clustering of attacks in intrusion detection systems. In Proceedings of the 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA), Abu Dhabi, United Arab Emirates, 3–7 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8.
46. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation Forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422. [CrossRef]
47. Khoury, J.; Klisura, D.; Zanddizari, H.; Parra, G.D.L.T.; Najafirad, P.; Bou-Harb, E. Jbeil: Temporal Graph-Based Inductive Learning to Infer Lateral Movement in Evolving Enterprise Networks. In Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2024; IEEE Computer Society: Piscataway, NJ, USA, 2023; p. 9.