Scientific Research Publishing

Journal of Information Security, 2025, 16(2), 270-282
https://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

Enhancing Microsoft CyberBattleSim for Enterprise Cybersecurity Simulations

Jackson Norris, Vijay K. Madisetti

School of Cybersecurity and Privacy, College of Computing, Georgia Institute of Technology, Atlanta, GA, USA
Email: jackson.kennedy.norris@gmail.com, madisetti.vijay@gmail.com

How to cite this paper: Norris, J. and Madisetti, V.K. (2025) Enhancing Microsoft CyberBattleSim for Enterprise Cybersecurity Simulations. *Journal of Information Security*, **16**, 270-282.
https://doi.org/10.4236/jis.2025.162014

Received: January 17, 2025
Accepted: April 5, 2025
Published: April 8, 2025

Abstract

Microsoft's CyberBattleSim environment effectively leverages Reinforcement Learning to simulate network intrusions and lateral movement, but its current implementation has limitations. In this paper, we extend the CyberBattleSim framework to support VLAN-based (Virtual Local Area Network) network segmentation. This modification enables researchers to design more realistic corporate network topologies, simulating both local and remote traffic management between isolated network segments. We present a novel methodology for integrating Access-Control Lists (ACLs) to enforce segmentation rules and demonstrate its application in a reinforcement learning (RL) setup. After implementing these enhancements, we benchmark the performance of several RL agents in the modified environment. The results show that network segmentation is effective at slowing an attacker attempting to move laterally through a simulated environment. Our work not only enhances the CyberBattleSim framework but creates opportunities for more robust research in attack-path prediction, lateral movement, and intrusion detection.

Keywords

CyberBattleSim, Network Segmentation, Reinforcement Learning, Lateral Movement, Intrusion Detection

1. Introduction

Quantifying the effectiveness of cyber defense mechanisms is inherently challenging. In the absence of active security incidents or ongoing penetration testing, security teams often lack the actionable data necessary for informed decision-making. Furthermore, reliance on external intelligence about Tactics, Techniques, and Procedures (TTPs) [1] can result in a reactive posture, leaving security teams perpetually fighting the last war.

Leveraging reinforcement learning in simulated environments offers a unique solution to generate actionable data through controlled attack scenarios. Tools like Microsoft's CyberBattleSim enable researchers and practitioners to simulate attacks and mitigate defender biases when analyzing network attack vectors. By studying the behavior of red-teaming agents in CyberBattleSim, researchers can design and test robust network architectures that effectively limit an attacker's success.

One widely used strategy for effective network security is the implementation of network segmentation. This strategy splits enterprise networks into smaller, isolated segments, each governed by specific security policies [2]. This approach can enhance an organization's ability to manage traffic [3], enforce compliance, and monitor activity through detailed logging. Segmentation is particularly effective at limiting attackers' lateral movement by cutting off potential attack paths. By containing threats within isolated segments, organizations can reduce the complexity associated with endpoint-specific security controls while also mitigating the impact of breaches, confining them to specific subnets.

Currently, the Microsoft CyberBattleSim environment does not support network segmentation. Traffic restriction is limited to node-specific firewall policies, meaning researchers can only set allow or deny policies at the node level. This limitation restricts realism and applicability of CyberBattleSim research.

In this research project, we extend the CyberBattleSim project to support a segmented network, explicitly incorporating abstracted Next-Generation Firewalls (NGFWs) into the simulated topology. These firewalls will allow researchers to design architectures more closely aligned with typical corporate network structures.

2. Existing Work

We first briefly describe existing features of Microsoft's CyberBattleSim.

2.1. Microsoft CyberBattleSim [4]

The Microsoft CyberBattleSim project offers a simulated corporate network environment designed for cybersecurity experimentation. Utilizing reinforcement learning, researchers train a red-team agent to infiltrate the network topology, while an optional blue-team agent defends against the intrusion [5]. Built on OpenAI's Gymnasium [6] framework (Gym), CyberBattleSim abstracts the complexities of a real network by representing machines as nodes in a graph. This abstraction simplifies setup and operation, making it accessible for researchers and developers without the need for a fully implemented network infrastructure.

CyberBattleSim's focus is on simulating the lateral movement [7] stage of cybersecurity intrusions, by concentrating on the attacker's ability to navigate the network after an initial breach. The environment emphasizes the attacker's navigation through the network to compromise additional nodes.

In CyberBattleSim, the red team agent's action space includes local attacks, remote attacks, and authenticated connections. Local attacks may expose passwords or reveal sensitive data, such as credentials [8], which are crucial for lateral movement. Remote vulnerabilities allow for direct node compromise without the need for lateral movement. Once a node is compromised, the red team agent can move laterally using common protocols like Secure Shell (SSH) or Remote Desktop Protocol (RDP). These protocols often require valid credentials to use. Lateral movement through these methods mirrors real-world corporate attacks, where threat actors gather credentials, exploit vulnerabilities, and navigate networks to achieve their objectives.

The red team agent's primary objective is to "own" as many nodes as possible. Owning a machine simulates privilege escalation and establishing persistence. The number of owned machines serves as both a reward for the agent and a measure of success in compromising the infrastructure.

In CyberBattleSim, "traffic" is abstracted as edges in a graph, with connectivity determined by local firewall rules at each node. Rather than simulating actual network traffic, the system evaluates the red-team agent's ability to move laterally based on local firewall configurations.

Because CyberBattleSim focuses only on node-specific firewall policies, it does not include an implementation of network segmentation. Without segmentation, the environment lacks typical corporate network structures that analysts rely on for lateral movement detection, such as monitoring firewall traffic logs. In real-world networks, these logs are crucial for identifying suspicious activity [9]. By incorporating network segmentation, CyberBattleSim could better reflect the complexities of actual corporate networks and significantly enhance detection and response capabilities.

2.2. Leveraging Deep Reinforcement Learning for Cyber-Attack Paths Prediction [10]

This research introduces a novel approach to modifying the CyberBattleSim environment by adjusting the Markov Decision Process (MDP) to focus the Red Team agent on source-target node pairs. This adjustment shifts the agent's perspective from a global network view to a localized one, enhancing its ability to generalize across diverse environments. The study demonstrates the effective application of Deep Reinforcement Learning (DRL), enabling the Red Team agent to efficiently identify and exploit vulnerabilities in previously unseen network topologies.

In its current implementation, CyberBattleSim abstracts traffic control at the node firewall level, which aligns well with the Markovian [11] framework. While this study highlights the potential of DRL within a simplified model, real-world networks typically feature more complex segmentation and advanced traffic management. Adapting this approach for broader applicability, particularly in environments managed by Next-Generation Firewalls (NGFWs), may necessitate a

more context-aware agent capable of interpreting traffic patterns and dynamic network segmentation.

Although this research validates the effectiveness of DRL in the existing CyberBattleSim environment, its applicability may be limited in highly segmented and dynamically controlled networks. These findings underscore the need for segmentation in CyberBattleSim.

2.3. Deception in CyberBattleSim [12]

Researchers have previously integrated deception technology directly into the CyberBattleSim interface to enhance detections and explore methods for slowing down attackers. In this setup, deceptive nodes are introduced into the environment, slowing the red team agent and generating a negative reward whenever the attacker interacts with these nodes. The rationale behind the negative reward is to discourage the red team agent from engaging with the deceptive elements. In theory, deception technology is beneficial for the blue team, as honeypots both hinder the attacker and alert the defender.

The research confirms that even obvious deceptive objects can slow down attackers and create better detection opportunities. The implementation of deception in CyberBattleSim was a key driver behind the introduction of network segmentation in the environment. While many existing blue team technologies provide robust detection and response capabilities, network segmentation is a novel CyberBattleSim innovation for more realistic environments.

3. Proposed Approach

To align with CyberBattleSim's abstraction, our implementation will model a segmented network structure without explicitly building traffic flows. As shown in Figure 1, this segmentation will consist of two key components:

1) VLAN Assignment: Each node will be assigned a VLAN ID to identify its segment.

2) Access-Control List (ACL): An ACL will govern traffic between VLANs, determining if connections are allowed or denied.

Whenever an attacker attempts to connect to a node in a different VLAN, they must query the ACL interface to check if the connection is permitted.

Additionally, this approach is not limited to creating remote firewall rules. Researchers will have the ability to define both local firewall rules at the node level or remote firewall rules via the ACL.

This design ensures that local and remote traffic management is available, enabling researchers to simulate more realistic network topologies.

4. Hypothesis

Segmenting a network in CyberBattleSim will reduce an attacker's cumulative reward compared to a flat network. Since cumulative reward quantifies an attacker's success by aggregating the rewards earned through compromised actions, it serves

as a reliable metric for evaluating the effectiveness of network segmentation.

5. Methodology

The CyberBattleSim environments are registered with the Gymnasium API to maintain consistency across the CyberBattle framework. This ensures that each environment can efficiently interact with the broader system while handling environment-specific configurations. The ability to modify these configs enables detailed analysis and testing. For example, a global variable like total_node_size can dynamically adjust the environment's size, aiding in the detection of overfitting in agents. By evaluating system performance under varying levels of complexity, researchers can identify models that achieve optimal generalization.

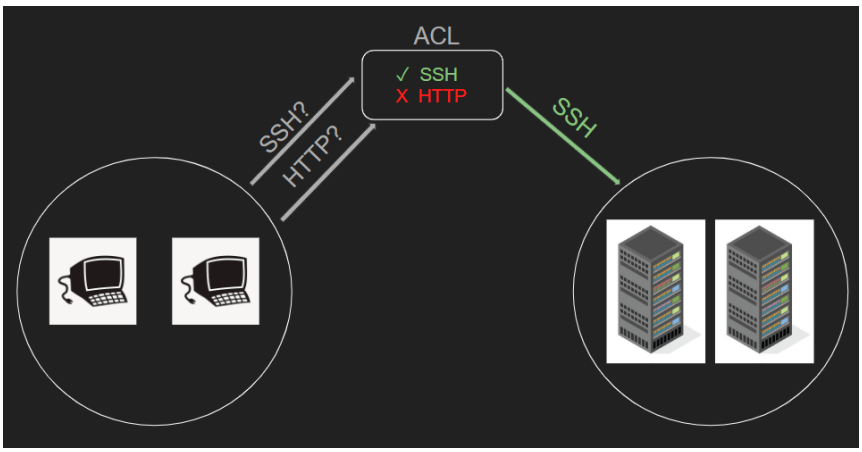

Figure 1. Sample ACL Diagram.

The Gym environment built for this research is registered as "CyberBattleSegmented-v0", following similar CyberBattleSim conventions such as "ActiveDirectoryTiny-v0" or "CyberBattleChain-v0".

When initiating the "CyberBattleSegmented-v0" environment, the Access-Control List (ACL) is automatically configured through the ngfw_config.py file. This file defines the ACL data structure, which regulates traffic flow between nodes within the environment. The ACL is designed with reusability in mind, allowing it to be implemented across other CyberBattle environments. While the Current Limitations section will discuss areas for improvement, this Methodology section focuses on the specific implementation of the ACL within the context of the CyberBattleSegmented-v0 environment.

5.1. Combining the Local and Remote Firewall

In traditional network security models, separating Local and Remote Firewall Rules adds complexity to traffic management because each rule type requires distinct configurations, policies, and enforcement mechanisms tailored to specific contexts [13]. Implementing this separation in CyberBattleSim would significantly increase the complexity of the simulation setup, making it less accessible

for researchers. To address this, the Access-Control List (ACL) in CyberBattleSim was adapted to accept both local and remote firewall rules within a unified framework. While this approach would be impractical in real-world corporate environments due to the nuanced requirements of local and remote traffic control, it simplifies the simulation by allowing researchers to apply both rule types through a single function, apply_rule. This trade-off prioritizes ease of use and experimentation within the simulation environment.

5.2. ACL Implementation

The ACL is implemented as a three-dimensional n × n × p NumPy (numpy) matrix, where:

1) n represents the total number of nodes in the environment.

2) p corresponds to the port designation (such as SSH).

3) Each entry encodes the firewall state between nodes i and j.

The decision to encode the firewall matrix as a numpy array was made to speed up the read/write access to the firewall data, ensuring efficient rule management.

To instantiate the firewall matrix a researcher needs only to supply the number of nodes and the list of ports. The researcher may also specify the default behavior of the firewall to ensure that traffic may be allowed or denied as expected.

The Firewall Matrix does not employ a precedence system; instead, preference is given to the most recently added rule.

5.3. Firewall State Implementation

As mentioned, the Firewall Matrix tracks the states for traffic between nodes. These include:

1) Allowed by both firewalls.

2) Blocked by the local firewall but allowed by the remote firewall.

3) Allowed by the local firewall but blocked by the remote firewall.

4) Blocked by both firewalls.

To enforce segmentation, the method to insert remote firewall rules cannot be called for two nodes of the same VLAN. Only local firewall rules can block traffic between nodes in the same VLAN.

Using numpy, remote firewall rules are inserted using the intersection of all nodes between two VLANs. This is achieved using the _ix command, which speeds up entry into the firewall matrix.

In summary, the Firewall Matrix simulates VLAN-based segmentation within CyberBattleSim by defining both local and remote traffic rules. Local rules govern node-specific permissions (e.g., allowing RDP traffic to a particular machine), while remote rules manage traffic between VLANs (e.g., blocking RDP between VLAN 0 and VLAN 1). These rules are dynamically enforced via the is_traffic_blocked method, which retrieves the firewall state between the two nodes. By default, all traffic is allowed until explicitly blocked, providing researchers with a clear indication of the firewall's status.

5.4. Testing and Validation Process

To ensure the firewall's functionality, a series of unit tests were implemented to verify rule application and traffic management within the matrix. For example, one test, test_block_and_allow, evaluated the behavior of the firewall when both blocking and allowing rules were applied sequentially.

Initially, a remote firewall rule was applied to block RDP traffic between VLANs 0 and 1. The test verified that the traffic was correctly blocked and that the output indicated the remote firewall as the blocking source.

Finally, the test overrode the block by applying a new remote firewall rule to allow RDP traffic between VLANs 0 and 1. The functionality was confirmed by ensuring the traffic was no longer blocked, and the Remote Firewall was correctly identified as allowing the traffic. This test demonstrates the system's ability to dynamically enforce and override rules, ensuring proper functionality under changing configurations.

This test is one example of the extensive testing conducted to ensure the firewall's reliability.

5.5. Default Configuration

The default configuration for CyberBattleSegmented-v0 is carefully designed, as shown in Figure 2, to simulate a realistic yet controlled corporate network environment. The network is segmented into distinct VLANs, including a Windows 10 VLAN for user devices, a Linux VLAN, an unpatched Windows VLAN, a Domain Controller (DC) VLAN, and a Flag VLAN. These segments reflect common enterprise practices of isolating critical resources and systems to enhance security. By default, all traffic to the Flag VLAN is blocked, except from the DC itself.

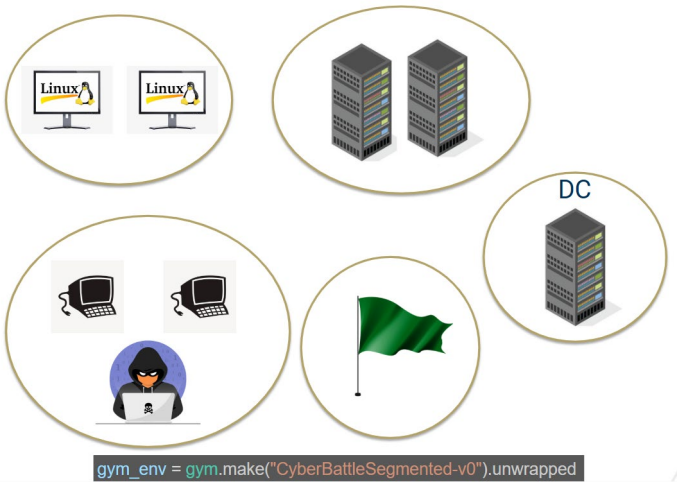

Figure 2. CyberBattleSegmented-v0 representation.

This setup ensures that attackers must compromise the DC to compromise the flag—a scenario that mirrors the importance of DC security in real-world attacks.

In the default CyberSegmented environment, the attacker starts in the Win-

dows 10 VLAN. The initial machine contains a local vulnerability that leaks credentials for another Windows 10 machine within the same VLAN. Beyond this, all other credentials are randomly distributed across nodes, with configurations determined by a random seed. The use of a seed introduces variability, ensuring that no two runs follow identical attack paths. Despite this randomness, the only way to access the Flag is by compromising the Domain Controller and extracting the NTDS database [14]. This design forces the attacker to follow a deliberate and realistic escalation path, mirroring the need for privileged credential access in real-world scenarios.

This configuration, as seen in **Figure 3**, strikes a balance between simplicity and complexity. The environment is small enough to enable extensive experimentation, yet it still challenges attackers to navigate segmentation, bypass restrictions, and exploit misconfigurations. These design choices ensure the framework is both realistic and effective in testing the impact of segmentation strategies on attack progression.

5.6. Training RL Agents

The RL agents used in this research were the out-of-the-box agents provided within CyberBattleSim for benchmarking purposes. These agents vary in complexity, offering a range of outcomes that effectively demonstrate the efficacy of blocking capabilities. This specific benchmarking set is widely used in the research community and was also employed in Microsoft's original research on CyberBattleSim.

These agents are as follows:

1) Random Agent: A baseline agent that exploits an environment randomly, serving as a simple control for evaluating performance against random actions.

2) Credential Exploitation Agent: A random agent that has the ability to exploit credential caches in the environment, providing a stronger baseline for agent performance in scenarios involving credential exploitation.

3) Tabular Q-learning Agent: Implements a discrete, model-free reinforcement learning approach using tabular Q-learning, providing insights into how traditional methods perform in this environment.

4) Deep Q-learning Exploring (DQL Explore) Agent: Utilizes neural networks to approximate Q-values, extending the basic Q-learning method to handle larger, more complex state spaces [15].

5) Deep Q-learning Exploiting (DQL Exploit) Agent: This agent builds upon the previous DQL model, exploiting the learned policy for improved decision-making.

The primary focus of this analysis is not to measure the performance or potential overfitting of the agents themselves but rather on using a diverse set of agents to evaluate the efficacy of network segmentation strategies. This approach ensures that the findings are relevant across a variety of scenarios, emphasizing the robustness of the proposed segmentation model.

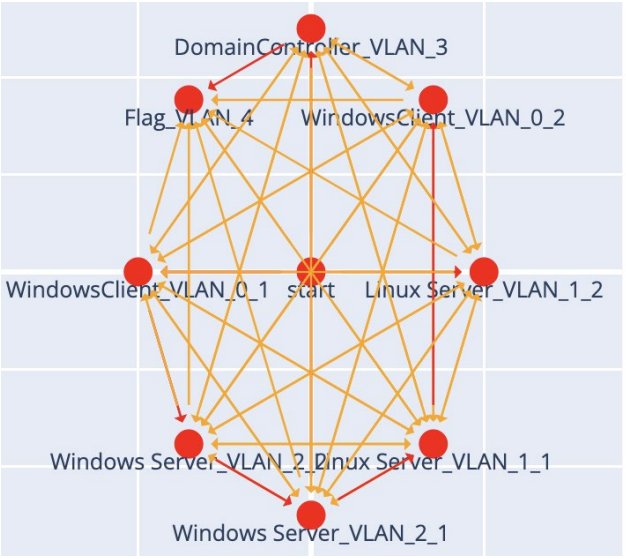

Figure 3. "Owned" CyberBattleSegmented-v0.

5.7. Benchmarking Success

Since CyberBattleSegmented is a custom environment, directly comparing a segmented network to a flat network can introduce bias. Blocking more traffic risks arbitrarily validating our hypothesis without sufficient proof, as higher blocking probabilities are designed to naturally reduce attacker rewards. This approach oversimplifies the analysis and fails to rigorously test the intended hypothesis.

Traditional CyberBattle benchmarks are not suitable for evaluating segmentation efficacy, as they primarily focus on comparing agent performance within a fixed environment. To address this gap, we propose a benchmarking framework that evaluates the relationship between the probability of a firewall rule blocking an attack and the cumulative reward achieved by the attacker.

In this framework, each remote firewall rule has a specified probability of blocking traffic, specifically for connection attempts via SSH or RDP. By varying the blocking probability, we generate a spectrum of outcomes rather than relying on a binary comparison between flat and segmented networks.

To mitigate the effects of outliers and cherry-picking (e.g., environments where the attacker cannot navigate effectively due to extreme configurations), we will use 15 random seeds to define the environment and firewall rules. For each seed, we will run 25 episodes, each consisting of 500 iterations. This setup allows agents sufficient exploration while ensuring variability across different environments.

We will calculate the cumulative reward for each episode and average these rewards within each seed. Finally, we will compute the overall mean and standard deviation of these averages across the 15 seeds for each blocking probability. This statistical approach ensures robustness and provides a comprehensive view of attacker success under varying segmentation scenarios.

By avoiding traditional benchmarks' limitations and focusing on the relationship between blocking probabilities and attacker rewards, this methodology cap-

tures the efficiency and trade-offs involved in deploying segmentation strategies within CyberBattleSim.

6. Results

The four reinforcement learning agents were able to navigate each environment to a similar success rate, though the performance bounds were notably broad. This variability reflected a binary distribution: agents often either successfully compromised the entire environment or were unable to escape their initial VLAN.

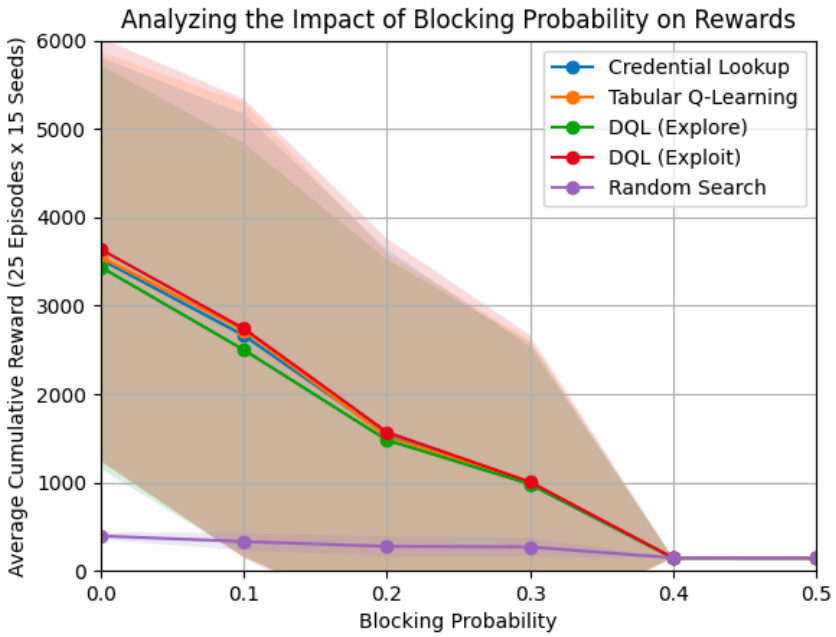

Figure 4. Impact of blocking probability.

In each blocking probability, as seen in **Figure 4**, the exploiting Deep-Q Learning Agent demonstrated the most success. This is a result of the agent having learned an effective navigation path and began exploiting the environment.

As the blocking probability increased, the agents struggled to escape their initial VLAN. In particular, the firewall's ability to block RDP traffic between VLANs significantly hindered lateral movement, effectively containing the attacker within its initial bounds. This demonstrates that segmentation strategies, especially those limiting RDP traffic, can substantially reduce the attacker's ability to move laterally and escalate.

These results highlight a key takeaway from the CyberBattleSegmented project: controlling lateral movement through segmentation and targeted firewall rules is a practical and effective method for mitigating attacker success.

7. Current Limitations

Because the default instantiator class, CyberBattleEnv, requires the configuration of local firewall policies, significant portions of the CyberBattleSim environment

had to be recreated to support this effort. The decision was made to prove the concept with copied and adapted scripts prior to implementing and merging any broader changes. Over the coming weeks and months, we anticipate sharing our results, engaging with the CyberBattleSim community, and soliciting feedback from its creators to inform the reconfiguration of the entire environment.

8. Real-World Implications

Findings from the controlled simulation can guide cybersecurity strategies in designing network architectures. As practitioners and researchers use this environment, they may discover novel methods for deploying policies to limit the spread of lateral movement. Understanding the optimal actions a blue team agent might take in response to a cyber intrusion can help refine incident response processes that mitigate threats effectively. Additionally, researchers may benefit from developing services with more stringent SLAs, ensuring that response actions are carefully evaluated for their operational impact. For instance, shutting off LDAP as part of a containment effort could result in widespread outages if user accounts are unable to authenticate. By exploring such scenarios, researchers can better understand how to design response actions that balance security needs with SLA commitments, minimizing disruption while maintaining robust protections.

9. Conclusions

The introduction of VLAN-aware segmentation and firewall rules significantly enhances the realism of attack and defense simulations within the CyberBattleSim environment. This extension allows red team agents to face the added complexity of discovering vulnerabilities and navigating segmentation boundaries, while blue team agents gain the advantage of enhanced detection capabilities, such as monitoring blocked traffic for signs of intrusion. By simulating these layered network security mechanisms, our approach offers a more accurate platform for evaluating and testing intrusion detection and response strategies.

Findings from this controlled simulation can help guide cybersecurity strategies in real-world environments by informing network architecture design and policy deployment. Practitioners and researchers may discover novel methods for limiting lateral movement and refining incident response processes to mitigate threats effectively. Additionally, insights gained from these simulations can aid in developing response actions that balance security needs with operational constraints. For instance, shutting off LDAP as part of a containment effort could result in widespread outages if user authentication is disrupted. By exploring such scenarios, this research highlights the importance of designing security controls that not only mitigate threats but also account for service-level agreements (SLAs) and operational continuity.

Ultimately, this work provides a more robust foundation for studying lateral movement, attack-path prediction, and the development of effective defense strategies, with the potential for future research into more dynamic, adaptive network

defenses.

10. Opportunities for Future Research

Future opportunities for growth align themselves with the mission of CyberBattleSim: to provide researchers with an environment to train agents and study Lateral Movement. CyberBattleSegmented extends that mission and opens up different methods to analyze Lateral Movement patterns.

Some future opportunities are:

1) Implementing blocks to remote vulnerability detection using VLAN aware nodes.

2. Enhancing agent training and analysis in segmented networks.

3) Designing a more secure architecture that limit lateral movement.

4) Integrating remote firewall changes into the blue team's action space, allowing the agent to dynamically isolate attackers.

5) Enhancing environmental SLA requirements to ensure that firewall rules support services essential for business operations.

Acknowledgements

We acknowledge Ying Zhe Loh (George), Richard Vincent, Steven Carman, and Emma Newcom for their assistance and support.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Raza, M. (2023) What Are TTPs? Tactics, Techniques & Procedures Explained. https://www.splunk.com/en_us/blog/learn/ttp-tactics-techniques-procedures.html

[2] Chandramouli, R. (2021) Guide to a Secure Enterprise Network Landscape (NIST Special Publication No. 800-215). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-215

[3] Internet Engineering Task Force (2012) Firewall Considerations for Operations and Management. https://www.ietf.org/archive/id/draft-ietf-opsawg-firewalls-01.txt

[4] Microsoft Research (2020) CyberBattleSim: A Platform for Simulating and Evaluating Cyber Defense Strategies. Microsoft Research. https://www.microsoft.com/en-us/research/project/cyberbattlesim/

[5] Kunz, T., Fisher, C., La Novara-Gsell, J. and Nguyen, C. (2024) A Multiagent CyberBattleSim for RL Cyber Operation Agents. Systems and Computer Engineering, Carleton University.

[6] Gym Library (2024) Gym Library Documentation. https://www.gymlibrary.dev/index.html

[7] MITRE Corporation (2019) TA0008: Lateral Movement. MITRE ATT&CK. https://attack.mitre.org/tactics/TA0008

[8] Microsoft (2024) Cached and Stored Credentials Technical Overview. https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/hh994565(v=ws.11)

[9] Siddiqui, L. (2024) Security Event Logs: A Beginner's Guide. Splunk. https://www.splunk.com/en_us/blog/learn/security-event-logs.html#:~:text=Network%20infrastructure%20or%20firewall%20logs&text=They%20track%20connection%20attempts%2C%20blocked,security%20teams%20can%20detect%20cyberattacks

[10] Franco Terranova, A., Lahmadi, A. and Chrisment, I. (2024) Leveraging Deep Reinforcement Learning for Cyber-Attack Paths Prediction: Formulation, Generalization, and Evaluation. *The 27th International Symposium on Research in Attacks, Intrusions and Defenses* (*RAID* 2024), Padua, 30 September 2024-2 October 2024, 1-16. https://doi.org/10.1145/3678890.3678902

[11] Moschovakis, J.R. Markov's Principle, Markov's Rule, and the Notion of Constructive Proof. UCLA Mathematics.

[12] Hong, Q.A., Li, J.Q., Guo, X.Z., Xie, P. and Zhai, L.D. (2023) Assessing the Effectiveness of Deception-Based Cyber Defense with CyberBattleSim. In: Goel, S. and Nunes de Souza, P.R., Eds., *Digital Forensics and Cyber Crime. ICDF2C* 2023. *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Vol. 571, Springer, 224-243. https://doi.org/10.1007/978-3-031-56583-0_15

[13] Fortinet (2025) How to Prevent Lateral Movement in Cybersecurity? Fortinet. https://www.fortinet.com/resources/cyberglossary/lateral-movement

[14] MITRE Corporation (2024) T1003.003: OS Credential Dumping: NTDS. https://attack.mitre.org/techniques/T1003/003/

[15] Baeldung (2021) Q-Learning vs Deep Q-Learning vs Deep Q-Network. Baeldung. https://www.baeldung.com/cs/q-learning-vs-deep-q-learning-vs-deep-q-network#:~:text=Essentially%2C%20deep%20Q%2DLearning%20replaces,%2C%20Q%2Dvalue