CyberSim RL: A Simulation Environment to Train, Test, and Evaluate Automated Cybersecurity Offensive Agents and Digital Twins

Daniel Müller
School of Modeling, Simulation, and Training,
University of Central Florida

Orlando, Florida
daniel.mueller@knights.ucf.edu

Sean Mondesire
School of Modeling, Simulation, and Training,
University of Central Florida

Orlando, Florida
sean.mondesire@ucf.edu

ABSTRACT

With the increasing sophistication of cyber threats and the frequency of ever-changing attacks, there is a pressing need to fill gaps in the cyber workforce without sacrificing training quality and relevancy. Additionally, current strategies for cybersecurity operations planning, assessment, and training are traditionally manual and resource-intensive, driving demand for innovative solutions that are automated, cost-effective, and adaptive. This work introduces a new constructive cybersecurity simulation that is expansive in the types of threats and networks it can model, scalable in the number of network entities and vulnerabilities to simulate, and aims to train and evaluate automated and human cyber attackers.

This new simulation is designed to follow the National Institute of Standards and Technology (NIST) for common cyber-attack vectors and standards and OpenAI and Farama's Gymnasium standard to facilitate the development of adaptive machine learning agents that can act as automated offensive forces (OPFOR) within the cyber realm. Additionally, the environment is used to develop and assess digital twins of known attacker and network profiles. These agents and digital replicas can emulate real advanced threats like red teamers, smart botnets, IoT devices, and swarms, ensuring a realistic training ground for cyber defenders, researchers, and cyber BLUFOR systems. By enabling the creation of automated cyber role players and the safe examination of cyber threats and countermeasures, this simulation significantly reduces the financial and temporal costs associated with cybersecurity workforce development and the advancement of automated cyber technologies.

ABOUT THE AUTHORS

Daniel Müller is a graduate of the Modeling & Simulation Master's program at the University of Central Florida in the Master's program. He is a member of the Human-centered AI Laboratory (HAIL), and his thesis focuses on adaptive machine learning agents in cybersecurity.

Sean Mondesire, Ph.D. is an Assistant Research professor at the University of Central Florida's Institute for Simulation and Training. His research specialties are machine learning and big data analytics, focusing on autonomous decision-making, high-performance computing for simulation-based training and education, and predictive modeling of large-scale, human-centered systems.

CyberSim RL: A Simulation Environment to Train, Test, and Evaluate Automated Cybersecurity Offensive Agents and Digital Twins

Daniel Müller

School of Modeling, Simulation, and Training, University of Central Florida

Orlando, Florida

daniel.mueller@knights.ucf.edu

Sean Mondesire

School of Modeling, Simulation, and Training, University of Central Florida

Orlando, Florida

sean.mondesire@ucf.edu

INTRODUCTION

The dynamic and ever-evolving nature of cybersecurity poses significant challenges in both the development of a skilled workforce and the implementation of effective defense strategies against increasingly sophisticated cyber threats. Recent trends in the cyber landscape, such as the proliferation of Internet of Things (IoT) devices, the surge in Internet users, and the expansion of potential vulnerabilities, have resulted in a dramatic escalation in both the frequency and complexity of cyber-attacks. For instance, the early 2010s were characterized by widespread email spam containing viruses and worms, demanding extensive efforts in network security management. However, today's cyber attackers exhibit greater sophistication, employing automated and targeted spear-phishing tactics to infiltrate organizations, leading to significant data breaches and ransom demands. The volume of attacks is overwhelming, with over 43 trillion security signals captured daily in 2022, indicating a rise in identity theft and malicious emails (Microsoft Security, 2022).

Despite the availability of cybersecurity training and ongoing cyberdefense efforts, there remains a significant gap in the labor market's ability to meet the growing demand for skilled cybersecurity professionals. According to the U.S. Bureau of Labor Statistics, cyber-related job openings are expected to rise significantly in the coming decade, yet recruitment challenges and training bottlenecks persist (Bureau of Labor Statistics, 2022). This shortfall is evident in various government departments, including the U.S. Department of Homeland Security, which reported over 2,000 open cyber positions in 2021 (DHS, 2021). The 2018 Department of Defense Cyber Strategy emphasizes the importance of developing a diverse and adaptable cyber workforce to maintain national security in cyberspace (DoD, 2018).

To address these challenges, this paper introduces an innovative cybersecurity simulation platform, designed in accordance with the National Institute of Standards and Technology (NIST) standards (NIST, 2023) and OpenAI and Farama's Gymnasium framework (OpenAI, 2023). This platform aims to facilitate the training and evaluation of both automated and human cyber attackers, providing a realistic, scalable, and controlled environment for conducting synthetic cybersecurity operations. The simulation enables the development of machine learning cyber agents that can act as automated offensive forces (OPFOR), thereby reducing the financial and temporal costs associated with cybersecurity workforce development. The agents and digital replicas created within this simulation can emulate advanced threats like red teamers, smart botnets, IoT devices, and swarms, offering a comprehensive and realistic training ground for cyber defenders and researchers. The remainder of this paper will delve into the state of the art in cybersecurity simulations, the necessity for automated cyber OPFOR, the architecture of the proposed simulation, and an example Monte Carlo experiment conducted using this cyber assessment platform.

BACKGROUND

A simulation is a replica of an actual process or system, and cyber simulation environments ensure the acceleration of cybersecurity training through computer-based models to replicate the behavior of cyber systems, networks, and scenarios. These simulations can be classified as *live*, *virtual*, and *constructive* (DoD, 1989) and allow participants to train and work together geographically, independent of each other. As displayed in Figure 1, participants interact with virtual or real objects provided by integrating different systems, which enables training multi-domain operations.

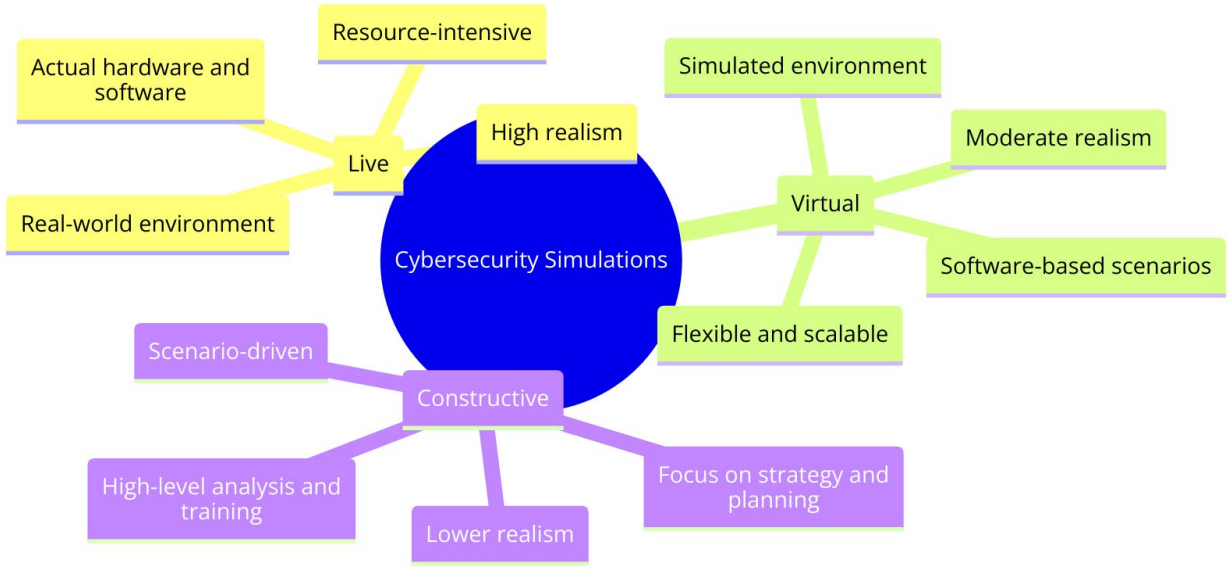

Figure 1. General Live, Virtual, and Constructive (LVC) simulations each have different fidelities and characteristics. Cybersecurity LVCs differ in realism, resource intensity, and user interaction

A *live* simulation is a reproduction of a physical system, event, or process, usually used for training, analysis, or prediction purposes. For example, military, vehicle, device operations, aviation agencies, air traffic control, and first responders use live simulation to conduct physical operations based on domain-specific scenarios. The simulation uses data and models to reproduce the behavior of the actual system, allowing users to make decisions and see the results in real-time interactively (Kavak et al., 2016). Real-Time Immersive Network Simulation Environment (RINSE) is a live simulation that supports large-scale wide-area networks (WAN) consisting of local-area networks (LAN). In the simulation, a user administrates a LAN and can use five categories of commands: Attack, Defense, Diagnostic Network Tools, Device Control, and Simulator Data. In contrast to virtual and constructive simulators, attacks in this live simulation are not executed by a script but by a human game manager. The game manager makes the simulation in a certain way adaptive since the game manager can freely design the attacks, but it does not save any training costs if an instructor has to be on-site at all times. In addition, the game manager is limited by the types of attacks since, even in this simulation, the attacks are limited to DoS, worm, or similar large-scale attacks (Liljenstam et al., 2005).

Virtual simulation is a computer-based simulation that uses computer graphics and other virtual reality technologies to represent real systems. The virtual environment represents motion, individual objects, or the entirety of a system. This type of simulation is used primarily in military training, process control, or architecture to test, evaluate, and make design and operational decisions about systems. With current technology, virtual simulations create highly realistic and interactive environments and provide users with an immersive experience (Kavak et al., 2016). An example of a virtual cyber simulation is NetENGINE, which simulates simultaneous, multiple users and computers interacting on large IP networks. The software is accessible via the internet using any web browser and focuses on the effects of a cyberattack rather than the technical details, which is another difference. The generic cyber-attacks are launched from a predefined master script and cannot be modified during the simulation. The simulated attack catalog includes distributed denial of service (DDoS) attacks, worms, and viruses (Brown et al., 2003).

Constructive simulations use mathematical models and algorithms to represent the behavior of real physical systems and processes. This simulation predicts a given system's behavior under certain circumstances and analyzes, tests, and evaluates systems. Real people make inputs in the simulated environment without being able to influence the output (Kavak et al., 2016). One example of a constructive cybersecurity simulation is SECUSIM. SECUSIM aims to specify attack mechanisms, verify defense mechanisms, and evaluate their consequences. The simulator creates virtual computer networks for its users, and its software has five modes: Basic, Intermediate, Advanced, Professional, and Application. The different modes allow various users to simulate their networks against cyber-attacks. These attacks

are stored in a scenario database and can be selected for each run, but do not change according to current events, let alone adapt to a different network configuration (Park et al., 2001).

In addition, to strictly live, virtual, and constructive simulations, hybrids and cross-over cyber simulations also exist. With StealthNet, the developers have taken a big step forward in developing cyber simulations. StealtNet is a live virtual, constructive simulation with real network hardware, such as routers, and virtual/constructive elements that attack the network hardware and exploit vulnerabilities. In contrast to previous simulations, integrating virtual and physical elements creates a realistic and immersive training experience. However, even in this simulation, a natural person must select and launch the attacks from a library. Although the library has a wide range of attacks with which the attacker can launch accurate cyberattacks, these are again fixed and need to be updated manually to reflect current events. Even this simulation is still far from an adaptive simulation with intelligent machine learning agents, contributing to cost reduction and quality improvement of cybersecurity training (Varshney et al., 2011).
Another development step in cybersecurity simulation is the introduction of cognitive agents as *defenders*. Cyber Security Instruction Environment (CYSTINE) is a virtual training system that develops these defenders for penetration tests. Because there are multiple acceptable paths for penetrating a network, a limited set of expected response actions and attributes and the time window within which these actions should occur are defined. In contrast to the other simulations presented here, CYSTINE does not simulate an attack on a network but is the defense to teach students penetration tests. Using a cognitive agent presents an active defense of the network. Cybersecurity students can use this simulation and the adaptive agent to constantly improve their skills in an authentic, changing scenario in a small area, increasing the quality of education. Unfortunately, the agent's range of action is too small to provide comprehensive, adaptive training in cybersecurity (Nicholson et al., 2016).

A similar live-virtual-constructive simulation to StealthNet is Cy-Through. Cy-Through combines the advantages of StealthNet, the realistic simulation with attacks on physical and virtual objects, and CYSTINE by using an agent to represent a variety of existing or potential cyber threats. The architecture of the Cy-Through platform is divided into two areas: constructive and live/virtual simulation. The constructive simulation is a Discrete Event Simulation (DES) environment that creates over 250 cyber threat scenarios, which trigger a specific threat effect. The Cy-Through agent is the virtual part of the platform and diversifies the cyber threats. The agent downloads the payload that contains a predefined attack script. Once triggered by the agent, the script automatically executes predefined malicious behaviors. Even though the Cy-Through Platform speaks of an agent, it can only be called intelligent to a limited extent since it simulates attacks or threats with the help of a script should certain events take effect (Lee et al., 2021).

The aforementioned existing simulations have been designed for their respective purposes and allow users to test their network configuration with the help of ready-made and predefined attack patterns. However, the use of machine learning cyber agents is a growing concept, and when used, such as in CYSTINE, these agents are used as defenders, not as attackers. Therefore, there is a need to extend AI cyber agents to take on the role of an attacker to increase training opposition for cyber workforce development and training. The solution proposed in this paper explicitly provides a training platform for current and future machine learning agents. Additionally, the simulator is designed to train single- and multi-agent cyber-defending and attacking systems. This novel approach makes cybersecurity training more multifaceted and realistic.

METHODOLOGY

This work introduces *CyberSim*, a continuous multi-agent simulation that provides a constructive cyber security simulation environment for developing intelligent, automated agents. CyberSim is designed to aid researchers and cybersecurity experts in testing their developed countermeasures in a secure environment. Additionally, the simulator provides a platform for developing and evaluating automated agents and adaptable attack scenarios to improve the quality of cybersecurity education to meet the increasing demand for and reduce the cost of cyber personnel. The open-source simulator is designed to address many of the shortcomings of cybersecurity simulations, focusing on fast, multithreaded, and scalable execution, creating a platform to generate automated cyber forces, simulating realistic cyber threats, and providing an extensible constructive environment that can be customized to the researcher and cyber analysts' needs. The remainder of this section defines the technical details of the simulator and how it is a tool that can be used to drive cyber workforce training.

Simulator Architecture

CyberSim is a constructive simulation that implements a simulation heartbeat model, where every agent has an opportunity to evaluate its state and environment and make one decision per time step (one simulation millisecond). CyberSim consists of two components: the core network simulator and its collection of independent decision-making agents. This modular structure allows researchers to develop and implement automated agents who can learn from the simulated environment and evaluate different network configurations and properties, such as the presence of vulnerabilities, internet of things (IoT) devices, and communication routes. Figure 2 (a) contains a class diagram of the simulator and (b) the default Agent class. In these two classes, the attributes and functions of each component are outlined and later defined in this section.

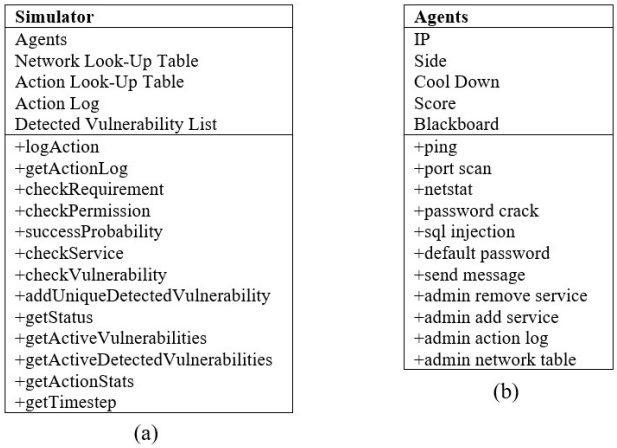

Figure 2. Class diagram of CyberSim that (a) defines the simulator's structure and (b) defines each simulated agent's attributes and functions

CyberSim has a continuous architecture in which each time step advances the simulator's virtual network and the agents within. The simulator relies on two look-up tables to process agent requests and direct network traffic: 1) network table (Table 1) and 2) action table (Table 2, Table 3). The network look-up table allows the simulator to create a virtual network consisting of various devices and routes. The simulator's user defines this network table's configuration before the start of a simulation, where each device in the table has an IP address, open ports, installed applications that use those ports, and possible vulnerabilities. At the simulator's initialization, the number of devices, ports, services, and vulnerabilities are established to allow for individual network customization for the user. The action look-up table defines which commands the offensive (OPFOR), and defensive (BLUFOR) sides of agents can perform on the network. Here, the OPFOR tries to discover network devices and their vulnerabilities by executing specific sequences of commands; each successfully executed command will reveal information about the network or enable vulnerabilities to exploit compromised devices. For example, an OPFOR agent must ping an IP to determine if the address is available on the network before attempting to execute a vulnerability on the IP and port. To encourage the correct execution of actions, each action (command) has a delay for success and failure attempts, and the information is returned to the requesting agent. For instance, ping commands return responses to the OPFOR agent significantly faster than a port scan, so the agent is encouraged to ping for IP discovery before scanning for available ports on undiscovered IPs. The BLUFOR side simulates network admins, and their commands provide the state of each network device and allow BLUFOR to modify the network to protect it from OPFOR exploits. The action look-up table defines which commands are available to both sides of actors, which information is returned when a command is successfully executed, and the time delays it takes for an action to be returned to the agent. Significantly, these look-up tables speed up simulator execution by minimizing network and action searches to near immediate time because they are implemented with table hashing.

Table 1. Network look-up table that contains the properties of each device on the network

IP	Port	Service	Vulnerability
192.168.0.2	-	ping	-
192.168.0.2	22	ssh	password crack
192.168.0.2	43	whois	-
192.168.0.2	80	apache	sql injection
192.168.0.2	443	apache	sql injection
192.168.0.3	3306	mysql	default password

In particular, the network look-up table is indexed (hashed) at the IP address for each device, allowing for instant retrieval of device information. Because of this indexing, retrievals are at an asymptotic runtime of O(1) to access all device information for a particular network device and O(1) for table updates. Action executions use the same indexing

on action types so the simulator can instantly retrieve action information. Lastly, the simulator has an action log that records all actions performed by all agents and the list of discovered vulnerabilities is defined. The action log contains the time step, the agent's IP address, the action, and the target IP, port, service, and vulnerabilities. This log is used for *after-action review* (*AAR*) of all simulated activities, agent performances, and discovered vulnerabilities.

The simulator validates submitted actions at each simulation step before processing its commands. This validation process uses both the network look-up table and the action table to first determine if the agent has permission to execute the action, has supplied the required parameters, and if the target of the action is present on the network.
The tables restrict the agents' ability to act, which is imperative to ensure realistic action. According to the pre-configured tables, the agent receives a reward when successfully discovering a vulnerability.
The action table specifies which information the agent must provide in its action in order to carry it out successfully. The simulator compares the agent's incoming command with the action table and sends an appropriate response. Next, the responses to the actions of the BLUFOR and the OPFOR agent are determined. These are explained in more detail in the agent class section.

Each agent contains properties necessary for it to operate in the simulation. First, each agent has a unique network IP address. The IP address simulates the attack's origin in the case of the OPFOR agent and the administrator's computer, from which he takes protective measures in the case of the BLUFOR agent. Secondly, the agent's side defines whether the agent is an attacker (Red) or on the defender team (Blue). Thirdly, a cool-down timer is defined, which is responsible for limiting the agent so it cannot carry out an action in each time step but must wait for a response from the system after the network command. In machine learning cases, agents also have a reward, which counts how many vulnerabilities the Red agent has discovered and exploited. Next, every agent has a log that lists all the actions performed by the agents during the simulation. This log is for optimization and debugging purposes, especially for machine learning. Lastly, the agent has a blackboard. The blackboard is a list of all action results and messages an agent has received from other agents. The blackboard acts as the agent's primary method to collect information and understand the state of the environment as a result of executing actions. Afterward, the agent is initialized with the corresponding team and the IP address to act in each time step and write it to the blackboard log file. The actions in the step function are the possible actions from which the agent can choose. Each action acts with a specific component in the network and may differ between OPFOR and BLUFOR agents.

Actions

The OPFOR agent has six actions: *ping*, *port scan*, *netstat*, *password crack*, *SQL injection*, and *default password*. The ping action executes the ping command to determine if an IP is available on the network. By default, this command returns the fastest response of all other actions because it should be the first action an agent performs to discover devices on the network. The second action is *port scan*, which determines if a port is open at an IP address. This action requires an IP address and a defined port from the agent to perform. The IP address is necessary so the OPFOR agent can follow a realistic attack graph and not search directly for a vulnerability. The port is necessary to ensure a targeted search by the agent. The *netstat* action identifies a service running at an IP address and on a specified port. Finally, the *password crack*, *SQL injection*, and *default password* actions are used by a Red agent to exploit a vulnerability. Each attack action requires an IP address, a port, and a service.

The BLUFOR agent can perform the same actions as the OPFOR agent to test the network's security and has four other actions available to facilitate its administrator role. The first additional action is *remove service*, which removes a service from the target computer. The *add service* adds a service to the target computer, such as running a database on port 443 or a new webserver on port 80. Additionally, this action can introduce vulnerabilities to those services or add patched services to the network. Second, the *action log* queries a network device for its entire action log to allow the BLUFOR agent to know what actions have been performed on the device so far in the simulation; this action allows the blue agent to take action accordingly to protect the network from current threats. Third, the *network table* action gives the agent insight into the current status of the network. The response displays all IP addresses, open ports, running services, and weak points in the network. This action also has no prerequisites. Lastly, all agents can communicate with others by using the *send message* action. This functionality places messages on a target's blackboard to share information and is particularly useful when multiple BLUFOR agents are simultaneously defending the network, or multiple OPFOR are collaborating to discover weaknesses on the network.

When executed, each of the above actions has a delay. Each agent must wait for this delay to execute a new action. The delay varies but is limited to a configurable minimum and maximum delay specified in the action look-up table. When an action is executed, the simulator randomly chooses a relay between this range and will only place the action's response on the agent's blackboard once that delay has elapsed. Furthermore, each action has a fixed timeout value. This value punishes an agent if it repeatedly performs an invalid action or incorrect commands.

Table 2. Part 1 of 2 of the action look-up table displays several of the actions the OPFOR agent can perform.

Action	Red	Blue	Success Probability	Delay Minimum	Delay Maximum	Timeout
ping	True	True	0.95	5	80	4000
port scan	True	True	0.95	50	100	10000
netstat	True	True	0.95	100	150	20000
password crack	True	True	0.001	25	50	100000
sql injection	True	True	1	10	30	100000

Table 3. Part 2 of 2 of the action look-up table

Action	Require Port	Require Service	Require Vulnerability	Require Misc	Return IP	Return Port	Return Service	Return Vulnerability
ping	False	False	False	False	True	False	False	False
port scan	True	False	False	False	True	True	False	False
netstat	True	False	False	False	True	True	True	False
password crack	True	False	True	False	True	True	True	True
sql injection	True	False	True	False	True	True	True	True

Table 4 Description of the captured metrics

Metric	Description
Successful Unique Actions	Using *Successful Unique Actions*, all successful actions of the agent are captured and recorded.
All Detected Vulnerabilities	The metric captures the number of all vulnerabilities identified and exploited.
Unique Detected Vulnerabilities	*Unique Detected Vulnerabilities* are used to detect only vulnerabilities that have been identified and exploited for the first time.
Active Vulnerabilities	The metric indicates the number of all active vulnerabilities according to the network look-up table.
All Active Detected Vulnerabilities	The metric captures, across all time steps, the vulnerabilities detected and exploited.
Unique Active Detected Vulnerabilities	*Unique Active Detected Vulnerabilities* are used to capture only active vulnerabilities that have not yet been closed by the BLUFOR agent and have been identified and exploited for the first time.

Metrics

CyberSim collects five metrics to evaluate the success of the OPFOR agent. The first metric is the number of unique successful actions. This value is recorded to ensure that the OPFOR agent does not repeatedly perform the same action to get a reward. Ideally, the agent's goal is to get all the information on the network table. The agent must discover different IP addresses, ports, services, and vulnerabilities to achieve this goal. Therefore, it is not beneficial if the agent constantly examines only one IP address for vulnerabilities. The second metric is the *number of vulnerabilities discovered*. This value determines how often the agent penetrates the target and discovers a vulnerability. Similar to capturing unique actions, the *number of unique vulnerabilities discovered* is a metric that serves as a tool for the agent

to find as many vulnerabilities in the system as possible. Furthermore, with the help of this value, it is possible to recognize whether the BLUFOR agent has opened other services with vulnerabilities or closed existing ones.
The OPFOR Agent has to react to the changing situation and try to find and exploit the new vulnerabilities. The last metric is the *number of active vulnerabilities*, which can change due to the actions of the BLUFOR agent. The information from this metric is compared with the OPFOR agent's activity log. This checks whether it has found all vulnerabilities.

Automated Agent Generation

CyberSim follows the OpenAI Gym, Farama Gymnasium, and PettingZoo (Terry et al., 2021) standards to support the development of automated agents using reinforcement learning. In the simulator, agents interact with the constructive environment and receive a positive or negative reward for each action performed. Based on the agents' and environment's states, each agent adjusts its actions and tries to maximize the total value of the reward. CyberSim makes this possible by rewarding both OPFOR and BLUFOR agents for actions taken. For example, the OPFOR agent receives small positive rewards when discovering an IP address or port. However, the agent receives a much larger reward should it identify and exploit a vulnerability. The BLUFOR agent, on the other hand, receives small rewards for changing the layout of the network and a large reward for closing an exploited vulnerability. Through these two measures, the agents will always try to find new ways to maximize the reward. Finally, metrics used for the rewards, action characteristics, network configuration, and action delays are all configurable in the simulation without needing to modify the code. This flexibility allows developers to craft different types of agents, experiment with various network configurations, and explore new machine learning approaches in cybersecurity.

RESULTS

A Monte Carlo simulation was performed to demonstrate CyberSim's functionality. In the demonstration, the simulation has the goal of the red OPFOR agents to discover all of the predefined security vulnerabilities on the network and exploit them. This example refrains from training ML agents but uses agents randomly selecting their limited actions to simulate agent decision-making. This decision was made because this work focuses on developing the environment for the future use of machine learning and agent training.

In the demonstration, the two red agents are given a set of IP addresses, ports, services, and vulnerabilities to simulate. Each agent has a set of actions to select from to discover the network configurations mentioned above. The same set of actions and the network configuration are defined in Tables 1-3 in the Methodology section. Once the simulation begins, each agent uses those actions to obtain feedback about the given network configuration, receiving positive feedback for each successful unique action performed on a unique target, unique vulnerabilities detected, and unique exploits executed. To observe the agent's progress, the action logs containing the metrics are output periodically and evaluated in an AAR.

Though displaying all network activity to the user in real-time slows down the simulation, this level of reporting can be beneficial in understanding the agent behavior as the simulator is running. This experimental setup was repeated 1,000 times, with each run executed for 10,000 time steps to obtain observable results and validate the simulation. It can be seen that the agent is constantly successfully performing actions to exploit the vulnerabilities. On average, the agent performs 151, a minimum of 125, and a maximum of 175 successful actions. This result verifies the functionality of the simulation environment. The results can be seen in Figure 3. All detected and active detected vulnerabilities and unique detected, and unique active detected vulnerabilities are identical due to the lack of changes in the network configuration by a BLUFOR agent. Therefore, they have been combined in Figure 4. Within 10,000 time steps, the agent finds several vulnerabilities twice, which was to be expected due to the chosen network configuration and Monte Carlo simulation. On average, the agent found 5.44 vulnerabilities at the end of the simulation, while only four unique ones were specified in the network configuration. However, the agent could not identify all unique vulnerabilities in every episode, which is the limitation of the maximum timestep of the simulation and the action-target permutation. On average, the agent only exploited 3.6 unique vulnerabilities; in a few runs, the agent only managed to find one. Though these experiments are not designed to develop machine learning agents, uses static agent-behaviors, and are executed for a relatively short simulation duration, the simulator has been evaluated to simulate over 24 hours of network activity in other experiments. The 10,000 timesteps simulator limit was chosen to examine the network's processing of actions, evaluate agents that continuously monitor their states and the environment, and agents that saturate the simulation with actions. Under the 10,000

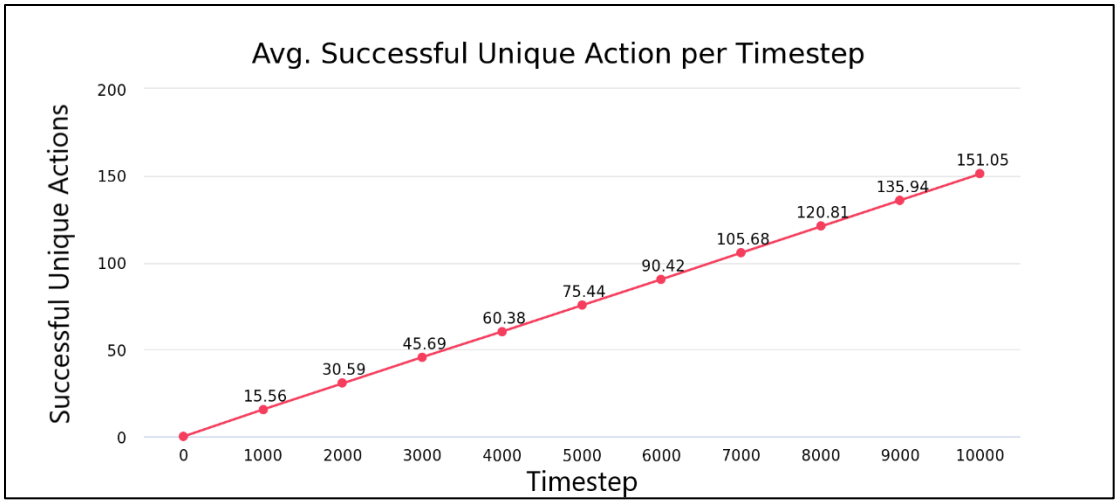

Figure 3. The average number of successful unique actions performed by two OPFOR agents in the Monte Carlo simulation

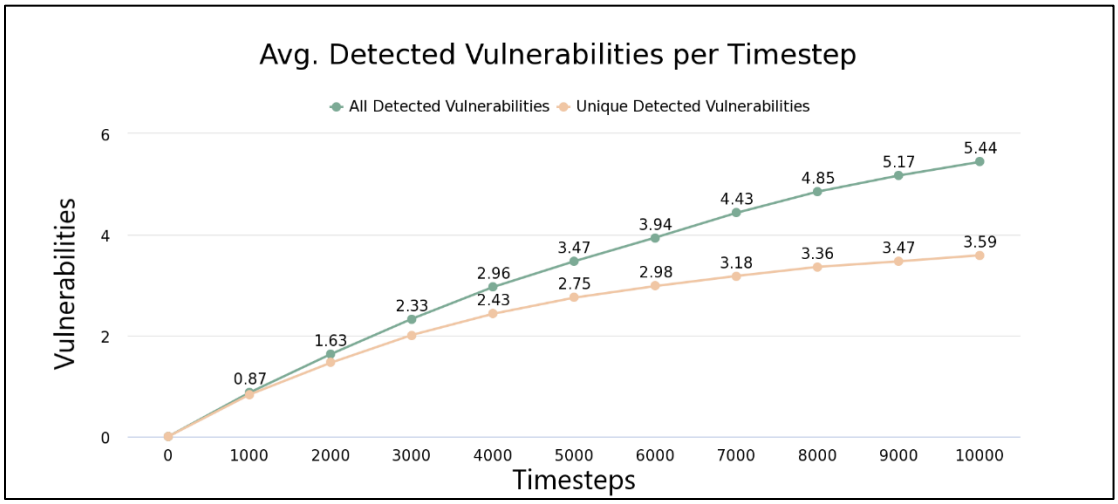

Figure 4. The average number of all detected and unique detected vulnerabilities by two OPFOR agents in the Monte Carlo simulation

timestep condition described in these demonstration results and extended stress-test conditions, the simulator has not displayed any performance issues and demonstrated to be scalable with an increased number of agents, actions, and network devices.

DISCUSSION

From the described CyberSim architecture and test results, the simulator is robust, scalable, and efficient. The simulator is robust because it is highly configurable with little-to-no-code changes to its architecture and agent behaviors. It is scalable because it can support an increased number of agents, network devices, and actions. Further, it efficiently uses indexed look-up tables for the agent, action, and device retrievals and updates. Nevertheless, limitations, challenges, and assumptions have been made in this simulation version to test the environment sufficiently. In this work, the agents used Monte Carlo-based action selection to discover exploits on the network; this omission of machine learning agents is because this work focuses on defining the simulator and its capabilities as a platform for agent development. Therefore, a generic Monte Carlo simulation was used for testing and demonstrating the simulator's capability. However, future iterations will explore the use of state-of-the-art machine

learning approaches, such as Deep Q-learning Networks (DQN) and Proximal Policy Optimization (PPO) (Starken, 2022) (Mondesire, 2023).

The presented results show that OPFOR agents select permissible actions according to the action table to identify the given network configuration and vulnerabilities. The results also show that the agent identifies all vulnerabilities through the Monte Carlo simulation. This capability supports machine learning agent training in this environment with fast action validation, environment updates, and a reward system that can steer agents to explore attack plans and adapt to dynamic network environments. Through this result, CyberSim offers the possibility to simplify the design and development of cybersecurity courses by offering an environment where trainees can take the place of these automated agents and interact with the synthetic network. Additionally, automated, adaptive agents can be trained to fill the human OPFOR or BLUFOR role and provide an adversary to train against, improving convenience and reducing training costs. For these reasons, explicit care was taken to develop the architecture in an open-source language and environment. By this measure, CyberSim meets the high demand for cyber defense software and services for human resources development and training opportunities (Geluvaraj et al., 2018) (Karagiannis & Magkos, 2020).

CONCLUSION

The development and deployment of CyberSim mark a significant step forward in automated cybersecurity training and preparedness. The need for such a simulation platform is underscored by the rapidly evolving landscape of cyber threats, where traditional training methods are no longer sufficient. CyberSim's ability to create realistic, dynamic network environments and simulate sophisticated cyber-attacks offers a cutting-edge tool for both current and future cybersecurity professionals. It provides a safe, controlled environment for practicing defensive strategies against a range of cyber threats, thereby enhancing the readiness and skills of cybersecurity personnel.

The potential of CyberSim extends to its application in developing digital twins of Cyber OPFOR and dynamic networks (Schiller, 2023). CyberSim enables a more profound understanding of potential vulnerabilities and defensive mechanisms by mirroring real-world network configurations and threats (Schiller, 2023). This aspect is crucial for future cybersecurity strategies, as it allows for anticipating and mitigating threats in a controlled setting before they impact real systems (Schiller and Mondesire, 2023) (Dauble and Mondesire, 2023). Digital twins in CyberSim represent an advanced approach to cybersecurity training, offering a realistic and comprehensive platform for understanding and responding to emerging cyber threats. This realistic approach is inspired by digital twin simulators in manufacturing domains and continues the innovation of providing AI-capable synthetic environments and analyses (Tse et al., 2024; Wright et al., 2024; Nsiye et al., 2024).

The extension of machine learning implementations in CyberSim will further enhance its automation and adaptability to model and detect cybersecurity vulnerabilities, as evidenced by the Monte Carlo results of the conducted experiments. The simulation's ability to process dynamic network environments, threat-actor models, and a catalog of NIST vulnerabilities and attack vectors make the platform capable of supporting a wide range of automated Cyber scenarios. Additionally, these characteristics with machine learning agents ensure that CyberSim stays relevant in the face of evolving cyber threats and makes it an invaluable tool for training cyber defenders. By modeling human-designed networks in the simulator, CyberSim provides critical feedback, aiding defenders in refining network designs and bolstering security measures. CyberSim emerges as a pivotal tool in the continuous effort to fortify cyber defenses against increasingly sophisticated threats.

REFERENCES

Brown, B., Cutts, A., McGrath, D., Nicol, D. M., Smith, T. P., & Tofel, B. (2003). Simulation of cyber attacks with applications in homeland defense training. *SPIE Proceedings*. https://doi.org/10.1117/12.500847

Dauble, J. & Mondesire, S. (2023). Prioritizing Improvements in Cyber Defender Knowledge, Skills, and Abilities using Cyber Simulation. In the Proceedings of the Simulation Innovation Workshop (SIW), Orlando, FL., 2023.

Fact sheet: Workforce sprint (May – June 2021) - DHS. (n.d.). Retrieved February 5, 2023, from https://www.dhs.gov/sites/default/files/publications/21_0701_dhs-factsheet-workforce-sprint-july-2021.pdf

Geluvaraj, B., Satwik, P. M., & Ashok Kumar, T. A. (2018). The future of cybersecurity: Major role of Artificial Intelligence, Machine Learning, and Deep Learning in cyberspace. *International Conference on Computer Networks and Communication Technologies*, 739–747. https://doi.org/10.1007/978-981-10-8681-6_67

Karagiannis, S., & Magkos, E. (2020). Adapting CTF challenges into virtual cybersecurity learning environments. *Information & Computer Security*, *29*(1), 105–132. https://doi.org/10.1108/ics-04-2019-0050.

Kavak, H., Padilla, J.J., Vernon-Bido, D., Gore, R.J. & Diallo, S., A characterization of cybersecurity simulation scenarios. (2016). *19th Communications & Networking Symposium (CNS 2016)*. https://doi.org/10.22360/springsim.2016.cns.003

Lee, D., Kim, D., Ahn, M. K., Jang, W., & Lee, W. (2021). Cy-through: Toward a cybersecurity simulation for supporting live, virtual, and constructive interoperability. *IEEE Access*, *9*, 10041–10053. https://doi.org/10.1109/access.2021.3051072

Liljenstam, M., Liu, J., Nicol, D., Yougu Yuan, Guanhua Yan, & Grier, C. (n.d.). Rinse: The real-time immersive network simulation environment for Network Security exercises. *Workshop on Principles of Advanced and Distributed Simulation (PADS'05)*. https://doi.org/10.1109/pads.2005.23

Microsoft Digital Defense Report 2022. Microsoft Security. (n.d.). Retrieved January 29, 2023, from https://www.microsoft.com/en-us/security/business/microsoft-digital-defense-report-2022

Mondesire, S. & Wiegand, R. P. (2023). Mitigating Catastrophic Forgetting with Complementary Layered Learning. Electronics - Special Issue: Modeling and Simulation Methods: Recent Advances and Applications, 1(1), 2023.

National Institute of Standards and Technology (NIST). (2023, December 7). Update on Cybersecurity Profile for Consumer Grade Routers.

National Institute of Standards and Technology (NIST). (2023, August 8).

Nicholson, D., Massey, L., Ortez, E. & O'Grady, R. (MODISM World, 2016) Tailored Cybersecurity training in LVC environments *2016 – MODSIM World 2016*

Nsiye, E., Wright, T., Tse, B., & Mondesire, S. (2024, May). A micro-discrete event simulation environment for production scheduling in manufacturing digital twins. In Proceedings of MODSIM2024. MODSIM.

US Bureau of Labor Statistics. (2022, September 8). *Information security analysts: Occupational Outlook Handbook*. US Bureau of Labor Statistics. Retrieved January 29, 2023, from https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm

OpenAI Gym. (2023). Gym Documentation. [Online Resource]. Available at: https://www.gymlibrary.dev/

Park, J. S., Lee, J.-S., Kim, H. K., Jeong, J.-R., Yeom, D.-B., & Chi, S.-D. (2001). SECUSIM: A tool for the cyber-attack simulation. *Information and Communications Security*, 471–475. https://doi.org/10.1007/3-540-45600-7_53

Schiller, T., Caulkins, B., Wu, A. S., & Mondesire, S. (2023). Security Awareness in Smart Homes and IoT Networks through Swarm-based Cybersecurity Penetration Testing. Information, 14, September 2023.

Schiller, T. & Mondesire, S. (2023). Human-out-of-the-Loop Swarm-based IoT Network Penetration Testing by IoT Devices. In the Annual Modeling and Simulation Conference 2023 (ANNSIM 2023) (p. 2). Hamilton, Ontario, Canada: Society for Modeling and Simulation International (SCS), May 2023.

Schiller, T. & Mondesire, S. (2023). Swarm-based IoT Network Penetration Testing by IoT Devices. In the 21st International Conference on Applied Cryptography and Network Security (ACNS 2023) (p. 5). Kyoto, Japan: Springer (LNCS Series), June 2023.

Starken, A., Caulkins, B., Wu, A. S., & Mondesire, S. C. (2022). Trends in Machine Learning for Adaptive Automated Forces. In the Proceedings of the Interservice/Industry Training, Simulation and Education Conference (IT/TSEC '22), December 2022.

Terry, J.K., Black, B., Grammel, N., Jayakumar, M., Hari, A., Sullivan, R., Santos, L., Perez, R., Horsch, C., Dieffendahl, C., Williams, N., Lokesh, Y. and Ravi, P. (October 26, 2021), PettingZoo: A Standard API for Multi-Agent Reinforcement Learning, *35th Conference on Neural Information Processing Systems* https://doi.org/10.48550/arXiv.2009.14471

Tse, B., Wright, T., Nsiye, E., Azinord, T., Medina, D., & Mondesire, S. (2024). Semiconductor manufacturing data synthesis through GANs. In Advanced Semiconductor Manufacturing Conference (ASMC), Albany, New York.

U.S. department of defense. (n.d.). Retrieved February 9, 2023, from https://media.defense.gov/2018/Sep/18/2002041658/-1/-1/1/CYBER_STRATEGY_SUMMARY_FINAL.PDF

US Department of Defense, *Department of Defense modeling and simulation (M&S) glossary*, DoD 5000.59-M, 1998.

Varshney, M., Pickett, K., & Bagrodia, R. (2011). A live-virtual-constructive (LVC) framework for Cyber Operations Test, evaluation and training. *2011 - MILCOM 2011 Military Communications Conference*. https://doi.org/10.1109/milcom.2011.6127499

Wright, T., Tse, B., Nsiye, E., Azinord, T., Medina, D., & Mondesire, S. (2024). Probabilistic modeling and machine learning for preventative maintenance prediction in semiconductor manufacturing. In Advanced Semiconductor Manufacturing Conference (ASMC), Albany, New York.