MininetGym: A Live Demonstration of RL-Based Cybersecurity Training

Salvo Finistrella
Department of Sciences and Methods for Engineering - University of Modena and Reggio Emilia - Italy
Reggio Emilia, Italy
salvo.finistrella@unimore.it

Stefano Mariani
Department of Sciences and Methods for Engineering - University of Modena and Reggio Emilia
Reggio Emilia, Italy
stefano.mariani@unimore.it

Franco Zambonelli
Department of Sciences and Methods for Engineering - University of Modena and Reggio Emilia
Reggio Emilia, Italy
franco.zambonelli@unimore.it

ABSTRACT

We present MininetGym, a framework for training and evaluating Reinforcement Learning (RL) agents in realistic cybersecurity scenarios. Attendees will experience live RL agents training through an intuitive dashboard that visualizes network traffic and agent decision-making in real-time. The demonstration show-cases three progressively complex scenarios: traffic classification, DoS attack detection with dynamic adversarial behavior, and distributed multi-agent attack mitigation.

KEYWORDS

Reinforcement Learning, Multi-Agent Systems, Cybersecurity, Software Defined Networking, Mininet, Intrusion Detection System

ACM Reference Format:
Salvo Finistrella, Stefano Mariani, and Franco Zambonelli. 2026. MininetGym: A Live Demonstration of RL-Based Cybersecurity Training. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 3 pages. https://doi.org/10.65109/VVUY3381

Demo video: https://youtu.be/pSdEV-MSdA8
GitHub repository: https://github.com/dipi-unimore/mininet-gym

1 INTRODUCTION

Cybersecurity defense systems increasingly require adaptive techniques capable of responding to sophisticated and evolving threats in real-time. Reinforcement Learning (RL) has emerged as a promising approach for developing intelligent intrusion detection and mitigation strategies [4]. However, current tooling for training and evaluation of RL-based cybersecurity solutions lack of reproducible testing environments that support realistic network conditions, has limited flexibility in configuring attack scenarios dynamically, and offer debugging support to understand agents' learnt behavior.

MininetGym [3] addresses these shortcomings by providing a interactive simulation framework that combines Software-Defined Networking (SDN) capabilities with standard RL interfaces, enhanced by a real-time web-based monitoring dashboard. Beyond single-agent learning, the platform specifically targets multi-agent systems (MAS) [2] research by enabling distributed agents with heterogeneous roles, partial observability constraints, inter-agent communication protocols, and team reward mechanisms, all fundamental challenges in autonomous agent coordination.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/VVUY3381

2 SYSTEM ARCHITECTURE

Tech stack. MininetGym integrates key technologies into a modular architecture: Mininet [7] provides network emulation running real Linux kernel network code for realistic virtual networks; OpenDaylight [8] and Open vSwitch [9] serve as SDN controller and switch, providing flow-level statistics via APIs; Gymnasium [6] offers standard RL environment API ensuring interoperability with RL libraries such as Stable-Baselines3 [10], that is also integrated to implement deep RL algorithms alongside custom tabular agents.

Architecture. The architecture follows a clean separation of concerns with five main modules (Figure 1): a web-based configuration editor and real-time monitoring and control dashboard; a Gymnasium-compatible wrapper around Mininet/SDN topology for the RL environment; an agent manager handling instantiation, training, and evaluation of RL agents; a traffic generator producing realistic attack scenarios; a results manager for data persistence, metrics calculation, and generation of plots.

Agents Management. For *single-agent* scenarios, the manager instantiates a single learning agent that interacts with the network

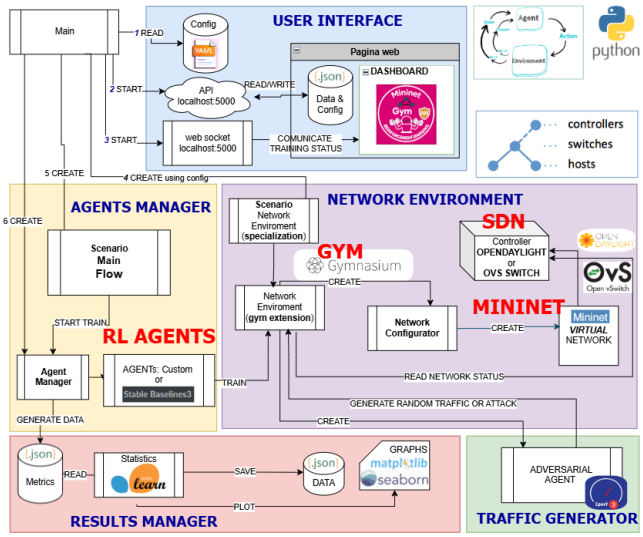

Figure 1: System architecture.

environment, receives observations, selects actions, and updates its policy based on received rewards. For *multi-agent* scenarios, the manager coordinates multiple heterogeneous agents with different roles, local observability of their network interface, and independent learning to detect and mitigate attacks.

Moreover, the manager implements a communication bus enabling explicit message passing between agents for coordination, and team rewards that incentivize collective performance rather than individual goals. The hot-swapping of learning algorithms enables researchers to compare centralized vs. decentralized learning, independent learners vs. coordinated policies, and communication-based vs. implicit coordination strategies.

Extensibility. MininetGym is designed for extension to new cybersecurity scenarios.

(1) Configure the experimental pipeline: environment setup → traffic generation → training → evaluation → visualization.
(2) Extend the `NetworkEnv` base class, which already handles SDN controller integration, traffic monitoring, and Gymnasium interface compliance.
(3) Define normal and attack traffic generators using the traffic module, which supports various protocols (TCP, UDP, ICMP) and dos attack types (udp flood, tcp syn flood, slowloris, icmp flood, DDos, http) [1].

The GitHub repository includes template environments and detailed documentation to guide the extension process.

3 DEMONSTRATION SCENARIOS

The live demonstration will showcase three RL scenarios with increasing complexity, each highlighting different aspects of the platform's capabilities [5].

Scenario 1: Traffic Classification. A single agent classifies network traffic into four categories (None, Ping, UDP, TCP). The agent observes a 4-dimensional state space (packets, bytes) and selects the action to classify traffic type. The reward function is symmetric, encouraging the agent to learn traffic patterns through exploration. This scenario serves as with supervised approaches: traffic samples are i.i.d. across time steps, so there is no sequential dependence, and the RL formulation reduces to a contextual bandit.

Scenario 2: Binary DoS Attack Detection. A single agent performs binary classification with dynamic difficulty. The state space includes 4-dimensional observations with percentage changes over time to detect traffic spikes. The agent chooses among 2 actions (report normal or attack) and receives asymmetric rewards: +2 for correct attack detection, +1 for normal traffic predictions, -0.1 for false positives, and -2 for missed attacks. Notably, attack probability increases over time if attacks go undetected, simulating adaptive adversaries that intensify their efforts when defenses appear weak.

Scenario 3: Multi-Agent DoS Detection. Host agents (one per network node) detect incoming and outgoing attacks under the limitations of partial observability, having available a 9-dimensional observation space (TX/RX packets, TX/RX bytes, variations, and coordinator messages) limited to their own network interfaces. Each host agent selects from 3 actions: report normal, incoming attack, or outgoing attack. Critically, host agents can autonomously detach their network link to block outgoing attacks when detected.

A coordinator agent performs global attack detection and alert broadcasting, also having partial observability of a 5-dimensional observation space (global packets, bytes, variations, and agent messages). The coordinator can broadcast alerts or remain silent, creating a communication bus for message passing between agents.

Workflow. Attendees will be guided through:

(1) *Configuration.* Attendees set up the network topology (switches, hosts, IoT devices) and the RL agents for training through a YAML configuration editor that provides real-time validation with syntax highlighting.
(2) *Training.* The web-based real-time dashboard displays network traffic with color-coded host states, live agent metrics updating every episode (accuracy), and reward curves. Training can be paused to examine specific decision points, and multiple agents can be compared to observe different learning strategies (Figure 2).
(3) *Coordination.* For the MARL scenario, attendees can experience the coordination between individual host agents that detect local threats and detach network links, and the coordinator that aggregates observations and broadcasts alerts.
(4) *Results.* The system automatically generates publication-ready plots: confusion matrices, training progression curves, and comparison histograms. Raw data and trained models can be exported for further analysis or continued training.

4 VALUE ADDED

The platform can serve multiple communities. **For multi-agent systems researchers**, MininetGym provides a realistic testbed for studying emergent coordination, partial observability, and decentralized decision-making in safety-critical domains. **For RL practitioners**, the platform enables fair algorithm comparisons with version-controlled configurations, supporting both classical tabular methods and modern deep RL approaches. **For educators**, the real-time visualization offers an intuitive understanding of agent learning dynamics, exploration-exploitation tradeoffs, and the challenges of multi-agent credit assignment. **For cybersecurity researchers**, it provides a reproducible environment for validating RL-based intrusion detection strategies. The actual impact will depend on adoption and documentation quality; to this end, the GitHub repository provides template environments and detailed documentation to support extension beyond the three scenarios presented.

Figure 2: Web-based, real-time training dashboard.

REFERENCES

[1] G. Carl, G. Kesidis, R.R. Brooks, and Suresh Rai. 2006. Denial-of-service attack-detection techniques. *IEEE Internet Computing* 10, 1 (2006), 82–89. https://doi.org/10.1109/MIC.2006.5

[2] Jacques Ferber. 1999. *Multi-agent systems - an introduction to distributed artificial intelligence*. Addison-Wesley-Longman.

[3] Salvo Finistrella, Stefano Mariani, and Franco Zambonelli. 2025. MininetGym: A modular SDN-based simulation environment for reinforcement learning in cybersecurity. *SoftwareX* 31 (2025), 102312.

[4] Salvo Finistrella, Stefano Mariani, and Franco Zambonelli. 2025. Multi-Agent Reinforcement Learning for Cybersecurity: Classification and survey. *Intelligent Systems with Applications* 26 (2025), 200495. https://doi.org/10.1016/j.iswa.2025.200495

[5] Salvo Finistrella, Stefano Mariani, and Franco Zambonelli. 2026. Experiences in Exploiting Reinforcement Learning for Network Traffic Classification and Attack Detection. In *Proceedings of the 18th International Conference on Agents and Artificial Intelligence (ICAART 2026)*. In press.

[6] Farama Foundation. 2023. Gymnasium. https://github.com/Farama-Foundation/Gymnasium. Accessed: July 2023.

[7] Neelam Gupta, Mashael S Maashi, Sarvesh Tanwar, Sumit Badotra, Mohammed Aljebreen, and Salil Bharany. 2022. A comparative study of software defined networking controllers using mininet. *Electronics* 11, 17 (2022), 2715.

[8] Zuhran Khan Khattak, Muhammad Awais, and Adnan Iqbal. 2014. Performance evaluation of OpenDaylight SDN controller. In *2014 20th IEEE international conference on parallel and distributed systems (ICPADS)*. IEEE, 671–676.

[9] Ben Pfaff, Justin Pettit, Teemu Koponen, Ethan Jackson, Andy Zhou, Jarno Rajahalme, Jesse Gross, Alex Wang, Joe Stringer, Pravin Shelar, et al. 2015. The design and implementation of open {vSwitch}. In *12th USENIX symposium on networked systems design and implementation (NSDI 15)*. 117–130.

[10] Antonin Raffin, Ashley Hill, Adam Gleave, Maximilian Ernestus, Noah Dormann, et al. 2021. Stable-Baselines3: Reliable Reinforcement Learning Implementations. https://stable-baselines3.readthedocs.io/en/master/. Accessed: 2025-05-16.