ISSN 2047-3338

Advancements in Automated Penetration Testing for IoT Security by Leveraging Reinforcement Learning

Abdul Samad[1], Saad Altaf[2], M. Junaid Arshad[3]
[1,2,3]Department of Computer Science, University of Engineering & Technology Lahore, Pakistan
[1]abdulsamad10911@gmail.com, [2]saadmayo9876@gmail.com

***Abstract*—Penetration testing, commonly referred to as pretesting or PT, is a prevalent method for actively evaluating the security measures of a computer network. This involves planning and executing various attacks to identify and exploit existing vulnerabilities. Despite the continuous evolution of tools, current penetration testing methods are becoming increasingly non-standard, intricate, and resource intensive. In this paper, we propose and assess an innovative AI-driven pen-testing system named the Intelligent Automated Penetration Testing System (IAPTS). This system utilizes machine learning techniques, specifically Reinforcement Learning (RL), to comprehend and replicate both average and complex pen-testing activities. IAPTS comprises a module that seamlessly integrates with established PT frameworks, allowing it to capture information, learn from experiences, and replicate tests in subsequent similar testing scenarios. The primary objective of IAPTS is to optimize human resources while delivering significantly improved results in terms of time efficiency, reliability, and testing frequency. The approach taken by IAPTS involves modeling PT environments and tasks as a partially observed Markov decision process (POMDP) problem, which is effectively solved by a POMDP solver. Although the focus of this paper is limited to PT planning for network infrastructures and not the entire practice, the findings strongly support the hypothesis that RL can elevate PT capabilities beyond those of any human PT expert, particularly in terms of time efficiency, coverage of attack vectors, and the accuracy and reliability of outputs. Furthermore, this research addresses the intricate challenge of capturing and reusing expertise by empowering the IAPTS learning module to store and reuse PT policies. This mimics the learning process of a human PT expert but in a more efficient manner.**

***Index Terms*—Machine Learning, Software Security, Automated Penetration Testing, Attack Tree, Deep Reinforcement Learning a Deep Q-Learning Network**

I. INTRODUCTION

IN the contemporary landscape of computer networks, the escalating frequency, complexity, and sophistication of cyber threats have made them more vulnerable than ever. Penetration testing (often referred to as pen-testing or PT) has emerged as a proactive approach to assess the security of digital assets, ranging from individual computers to websites and networks [21], [22]. This method involves actively seeking and exploiting existing vulnerabilities, mirroring the operational mode employed by hackers in real-world cyber-attacks.

In the evolving digital environment, PT has become a vital component of cybersecurity auditing, especially with the implementation of the European General Data Protection Regulation (GDPR) [23] for organizations. Besides meeting legal requirements, PT is recognized by the cybersecurity community as an effective means to evaluate the robustness of security defenses against skilled adversaries and to ensure adherence to security policies. The PT process, as depicted in Figure, unfolds in multiple stages, demanding a high level of competence due to the intricate nature of digital assets, such as medium to large networks.

Efforts have been made in research to explore the potential of automated tools for different PT stages, including reconnaissance, identification, and exploitation. However, while automation can alleviate the burden of repetitive tasks, PT remains a dynamic and interactive process, requiring advanced cognitive skills that are challenging to replicate through automation.

The question arises as to whether Artificial Intelligence (AI) can go beyond simple automation and provide expert-like output. AI, particularly in the sub-field of machine learning (ML), has shown promise in offloading work from humans and handling details that humans may struggle to manage quickly or accurately. The rapid progress in AI and ML leads to the belief that an AI-based PT system, employing well-established models

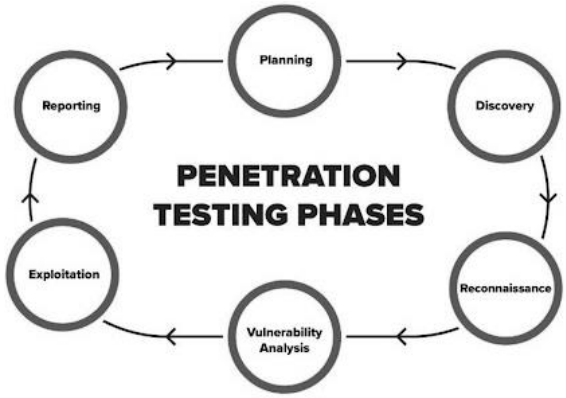

Fig. 1: Stages of Penetration Testing

and algorithms for sequential decision-making in uncertain environments, can bridge the gap between automation and expertise in the PT community.

Existing PT systems and frameworks are evolving towards becoming more autonomous, intelligent, and optimized[20]. The goal is to systematically and efficiently check all existing threats without heavy human intervention. Moreover, these systems should optimize resource utilization by eliminating time-consuming and irrelevant directions, ensuring that no threat is overlooked.

Beyond the execution of PT, the testing results need to be processed and stored for further use. Unlike human PT experts who continuously learn from tests and enrich their expertise, automated systems often lack reusability of data. This becomes crucial, especially in scenarios like regular compliance tests, where the testing is repeated. In practical terms, as network configurations often remain relatively stable, the output of previous tests could be applicable for eventual re-testing triggered by specific changes, such as network modifications, system upgrades, or security policy modifications.

Automation is deemed the optimal solution to save time and resources in various domains, and PT is no exception. The offensive cybersecurity community has significantly focused on automation in the past decade, leading to notable improvements in task efficiency. However, considering the unique challenges of PT, including the growing size and complexity of assets and the multitude of vulnerabilities to be covered, blind automated systems may fall short and perform worse than manual practices. Consequently, researchers are exploring ways to enhance such systems by adopting diverse solutions.

This paper delves into the thorough design and development of an ML-based PT system. The objective is to conduct intelligent, optimized, and efficient testing by autonomously perceiving its environment and deciding how to act in PT tasks, akin to human experts. The envisioned outcome is a system that not only saves time and resources but also enhances accuracy, testing coverage, and frequency.

II. Literature Review

This world is based on multidisciplinary research focused on the performance and optimization of network security assessment methods, specifically Vulnerability Assessment (VA) and Perception Testing (PT). The main points from the study and accepted practices are summarized, with initial emphasis on the planning phase, use in the PT industry, and research. In the realm of PT automation and augmentation, research has explored various axes in the cybersecurity and artificial intelligence field. Early studies modeled PTs as shot charts and decision trees, constructing them as sequential decision processes, but these approaches are prone to poor evaluation [8], [9].

Automating PT tasks is a recommended strategy for effective PT work, but full automation without optimization or prioritization often requires human supervision, particularly in large and medium-sized assets. Blind automation can lead to problems such as long-term testing, vulnerabilities in network connections, and security solution compromises. Previous research primarily focused on optimizing the planning phase but faced scalability limitations in larger networks. A notable work introduced a PT model based on Planning Language Description (PDDL) to handle attack and post-attack scenarios, albeit limited to small and medium-sized networks. Some studies explored how intelligence could enhance physical training, but uncertainty in physical training remains a challenge. An exception is the integration of ML algorithms into Core-Impact PT and VA systems, although it does not model the full PT application [6], [11].

Automating PT tasks is a recommended strategy for effective PT work, but full automation without optimization or prioritization often requires human supervision, particularly in large and medium-sized assets. Blind automation can lead to problems such as long-term testing, vulnerabilities in network connections, and security solution compromises. Previous research primarily focused on optimizing the planning phase but faced scalability limitations in larger networks. A notable work introduced a PT model based on Planning Language Description (PDDL) to handle attack and post-attack scenarios, albeit limited to small and medium-sized networks. Some studies explored how intelligence could enhance physical training, but uncertainty in physical training remains a challenge. An exception is the integration of ML algorithms into Core-Impact PT and VA systems, although it does not model the full PT application [6], [12].

Despite theoretical advances, penetration testing is associated with slow-running routine operations in large networks. Solutions often falter, as evidenced by ongoing challenges in PT implementation, including time and resource limitations. Human capabilities are proven to be limited compared to machines for many tasks, especially given modern computing resources. An average access test takes a significant amount of time, ranging from a few days to weeks for testing the average LAN [13], [14].

III. Methodology

The study focused on assessing the Shadov system's capability in gathering factual data for designing attack trees and the Mulval platform for generating attack trees. A novel approach involved the development of a method to create a cyber intrusion matrix using the Mulval tool. Additionally, enhancements were made to the Deep Q-Learning Network method for analyzing the matrix and determining optimal attack trajectories. The study employed the deep reinforcement learning method, utilizing reward scores based on CVSS ratings for each node. This approach facilitated the reduction of attack trees and improved the identification of high-probability attacks. The comparative analysis included automated penetration testing methods, revealing practical implications for enhancing computer system security through the developed methodology.

A. Dynamic Cyber Intrusion Matrix Formation

Objective: Construct a dynamic matrix to represent cyber intrusions that evolve with real-time data.

Equation 1:

$$Threat\ Matrix_{i,j}(t) = Mulval\left(Shadov\left(Dataset_{i,j}(t)\right)\right)$$

Where $\mathrm{Dataset}_{i,j}(t)$ denotes the dataset at time t for intrusion types i and j, and Mulval and Shadov represent functions of Mulval platform and Shadov system, respectively.

Procedure:

- Continuously update the threat matrix based on real-time data, ensuring a dynamic representation of emerging threats.
- Integrate temporal aspects and attack paths to enhance the matrix's descriptive power.

B. Reinforcement Learning-based Attack Tree Optimization

Objective: Optimize attack trees for automated penetration testing using a Deep Q-Learning Network (DQN).

Equation 2:

$$\Big(Q(s,a) = (1-\alpha).Q(s,a) + \alpha\left(R(s,a) + \gamma.\ \max\big(Q(s',a')\big)\right)\Big)$$

Where Q(s,a) is the Q-value for state-action pair (s,a), R(s,a) is the reward, α is the learning rate, γ is the discount factor, and s' is the next state.

Procedure:

- Train the DQN using the dynamic threat matrix, updating Q-values based on the received rewards and predicted future rewards.
- Incorporate temporal aspects and severity scores from CVSS ratings into the reward assignment process.

C. CVSS-based Reward Assignment and Adaptation

Objective: Enhance the DQN's understanding of cyber threats by incorporating Common Vulnerability Scoring System (CVSS) ratings.

Equation 3:

$$Q(s,a) = CVSS\ Mapping\big(CVSS\ Score(s), Attack\ Likelihood(a)\big)$$

Where CVSS Score(s) represents the CVSS score of state s, and Attack Likelihood}(a) is the likelihood of the attack a.

Procedure:

- Develop a mapping function to translate CVSS scores into reinforcement learning-compatible reward values.
- Integrate CVSS-based rewards into the Q-learning process, adapting the DQN to prioritize actions with higher security impact.

IV. Reinforcement Learning in Intelligent Decision-Making Processes

The exploration and analysis of intelligent decision-making processes are integral fields within computer science, particularly in artificial intelligence (AI). Intelligent Automated Penetration Testing Systems (IAPTS) are designed to streamline penetration testing (PT) activities, aligning with various intelligence-driven cybersecurity solutions. These solutions encompass expert-oriented systems and electronic devices employing maintenance-free methods [16]. Expert-focused systems, including anti-virus (AV), firewall (FW), detection and intrusion prevention systems (IDPS), and security information and management presence (SIEM), rely on the expertise of security professionals..

A. Integration of Reinforcement Learning in IAPTS

The integration of reinforcement learning (RL) techniques has brought about changes in learning objectives, particularly in sustainability, such as vulnerability assessment and PT [11]. Several reasons support the choice of IAPTS that supports RL:

i. *Learning Benefits:* Reinforcement encourages continuous learning through interaction with the environment.
ii. *Learning as Reward:* The system adjusts its behavior as a reward, providing flexibility and the option to delay rewards to achieve long-term goals.
iii. *Improving the Learning Environment:* Learning support captures the characteristics of PT, including uncertainty and complexity.

Reinforcement learning, a part of machine learning and artificial intelligence, enables software developers to analyze performance in specific contexts. Minimal feedback (reward) is required for the agent to learn and change its behavior regarding the interaction between the agent and the environment [5]. Fig. 2 illustrates how reinforcement learning enables employees to learn from interactions in the environment and continually improve their decisions to adapt to the future. Compared to traditional methods, reinforcement learning methods provide better learning and reduce the need for service manuals by adapting expert-driven systems with machine learning. Additionally, reinforcement learning remains an ongoing area of research, with recent algorithmic developments and effective tools proving the effectiveness of solving complex learning problems in limited resources [17], [18].

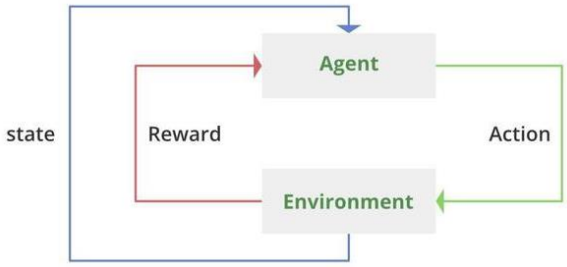

Fig. 2: Reinforcement learning interaction

B. POMDP Modeling of PT: Enhancing Testing Efficiency

In the context of PT, an attack consists of many tasks performed manually by the human tester or the PT platform. These periodic activities aim to meet defined or unknown goals (goals), which may be physical or content-based, including programs, computers, or information stored on computers. The nature of the target during the attack adds to the difficulty. The first challenge is to create a PT system that optimizes testing, coverage, and reliability within a limited time frame.

i. Preliminary Study Using POMDP Modeling: A preliminary study utilizes the partial Perceptual Markov Decision Process (POMDP) model for PT. POMDP stands for the proxy to an unknown destination. The POMDP set is represented as M = { S, A, O, T, Ω , R, b_1} consisting of states (S), actions (A), and observations (O). Random state transitions are controlled by the function $T: S \times A \times S \longrightarrow [0,1]$. Reinforcement learning helps improve the efficiency, effectiveness, and reliability of PT systems according to research objectives [12]. The POMDP model forms the basis of the RL-based PT control system by determining the interaction between the agent and the environment in which the agent works, is evaluated, and receives rewards [12]. The previous discussion [12] explains the rationale behind this model choice, and the next section provides more information [17], [18].

ii. POMDP Modeling of PT: In the PT context, an attack comprises a series of tasks executed either manually by a human tester or through a PT platform. These activities, performed periodically, aim to achieve predetermined or unknown goals (referred to as targets). Targets may be logical or physical entities, encompassing computers, computer networks, or information stored on computers. The dynamic nature of attack targets during an attack adds a layer of complexity. The overarching challenge lies in developing a PT system that optimizes testing efficiency, coverage, and reliability within specified time constraints. The initial study adopts Partially Observable Markov Decision Process (POMDP) modeling for PT. POMDP represents an agent navigating an uncertain environment. The POMDP set, denoted as M = { S, A, O, T, , R, }, encompasses states (S), actions (A), and observations (O). Stochastic state transitions are governed by the function $T: S \times A \times S \longrightarrow [0,1]$.Reinforcement learning is employed to enhance the efficiency, effectiveness, and reliability of the PT system, aligning with the research objectives [12]. The POMDP model elucidates the interaction between the agent and the environment, as agents execute actions, receive evaluations, and garner rewards, forming the bedrock of RL-led PT system management [12]. The rationale behind this modeling choice has been expounded upon in previous discussions [12], with further insights provided in subsequent sections [17], [18].

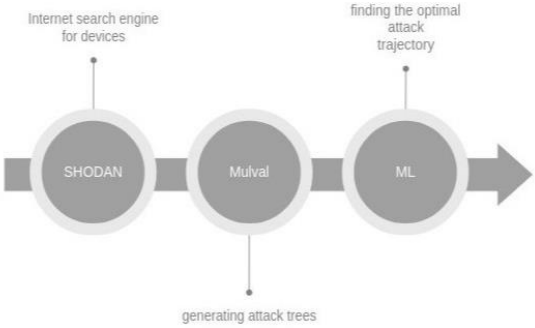

Fig. 3: Attack trees generation

V. Results and Discussion

We utilized Shodan-acquired data to initialize details regarding vulnerabilities, open ports, products, and protocols associated with both the web server and the file server. Regarding the workstation, we assumed it operates without hosting services, relying solely on different transfer protocols.

A. Identification of Vulnerabilities

In the realm of automated penetration testing using reinforcement learning (APTS), datasets like these serve as invaluable building blocks. Each entry, pairing a hardware component with a known vulnerability, forms a training instance for the APTS model (Table I).

Table I: Types of vulnerabilities

Hardware	Vulnerability List
Web server	CVE-2020-1198
First subnet	CVE-2016-0189
Second subnet	CVE-2020-1380
File-server	CVE-2010-0492

B. Generation of Attack Graph

Utilizing the Mulval system, an attack graph for the studied computer system is produced. The attack graph features vertices representing system configuration, potential privileges, and preconditions/postconditions.

C. Refinement of Attack Graph

The attack graph is further refined through algorithms developed by the authors. The refinement involves modeling with deep reinforcement learning, considering the location of the agent, vulnerabilities, and attacker targets as vertices.

D. Automated Penetration Testing Model

The automated penetration testing method incorporates the Common Vulnerability Scoring System (CVSS) to assign rewards based on vulnerability exploitation with the rewarded Deep Q-Learning (DQL) model. For instance, a simulated scenario starting from a node, providing options for the malicious agent to either reach the target node for a reward of 100 or return to the starting position.

E. Generalized Matrix of States

A generalized matrix of states with States/Actions (S/A) is formed to illustrate the transition between agent nodes and associated bonuses. Table II represents the states and actions, providing insights into the attacker's behavior.

Table II: Generalized matrix of states

S/A	12	13	21	25	43	4	15	23	34
12	-1	7	8	9	6	-1	-1	-1	-1
13	0	-1	-1	-1	-1	100	-1	-1	-1
21	0	-1	-1	-1	-1	-1	100	-1	-1
...	...	...	...	...	...	...	...	...	...

F. Submatrix of State Clarifications

A submatrix is derived from the state matrix, providing state clarifications and additional bonuses. Table III illustrates this submatrix.

Table III: Submatrix of state clarifications

Submatrix	12	13	21	25	43	4
12	-1	7	8	9	6	-1
13	0	-1	-1	-1	-1	100
...	...	...	...	...	...	...

G. Bonuses Diagram

Utilizing the submatrix, a diagram of bonuses to the attacker agent is formed for a specified goal for transitioning to target state 1.

H. Practical Examples

Table IV provides practical examples of results obtained through the automated penetration testing method, showcasing the maximum rewards for different attacker node simulations targeting various goals in the computer system.

Table IV: Automated penetration testing method result

Attacker Node Simulator	Target	Maximum Reward
12	4	908
12	15	907
12	23	906
12	34	906

VI. Conclusions

A novel method for automated penetration testing has been devised, featuring an innovative integration of the Shodan search engine, MulVal network security analysis platform, and software vulnerability data (CVE). This integration facilitates the acquisition of input data, enabling the construction of realistic attack scenarios validated through deep reinforcement learning technology.

The method's key strength lies in its ability to generate attack trees for diverse training procedures and optimize corresponding scripts for automated software security testing. The deep reinforcement learning approach leverages reward scores assigned to each node based on the Common Vulnerability Scoring System (CVSS) rating. This enables the reduction of attack trees, pinpointing attacks with higher probabilities of occurrence.

To evaluate the method's practicality, an experiment was conducted, resulting in the generation of an attack tree and the formulation of testing and training scenarios. Notably, the simulation results achieved a 0.9 accuracy in determining the most rational attack path, even with a limited number of training scenarios.

The developed method emerges as an effective solution for software security analysis, providing testers with the flexibility to adopt ethical hacking practices and implement strategies to mitigate potential negative impacts of cyberattacks..

References

[1] Sarraute, C. Automated attack planning. Available online: https://arxiv.org/abs/1307.7808.

[2] Creasey, J.; Glover, I. A guide for Running an Effective Penetration Testing Program; CREST Publication: Slough, UK, 2017. Available online: http://www.crest-approved.org.

[3] Sutton, R.S.; Barto, A.G. Reinforcement Learning: An Introduction; MIT Press: Cambridge, MA, USA, 2018.

[4] Almubairik, N.; Wills, G. Automated penetration testing based on a threat model. In Proceedings of the 11th International Conference for Internet Technologies and Secured Transactions, ICITST, Barcelona, Spain, 5–7 December 2016.

[5] Veeramachaneni, K.; Arnaldo, I.; Cuesta-Infante, A.; Korrapati, V.; Bassias, C.; Li, K. AI2: Training a Big Data Machine to Defend; CSAIL, MIT Cambridge: Cambridge, MA, USA, 2016.

[6] Hoffmann, J. Simulated penetration testing: From Dijkstra to aaTuring Test++. In Proceedings of the 25th International Conference on Automated Planning and Scheduling, Israel, 7–11 June 2015.

[7] Heinl, C. Artificial (intelligent) agents and active cyber defence: Policy implications. In Proceedings of the 6th International Conference On Cyber Conflict (CyCon 2014), Tallinn, Estonia, 3–6 June 2014.

[8] Walraven, E.; Spaan, M. Point-Based Value Iteration for Finite-Horizon POMDPs. J. Artif. Intell. Res. 2019, 65, 307–341.

[9] Sarraute, C.; Buffet, O.; Hoffmann, J. POMDPs make better hackers: Accounting for uncertainty in penetration testing. Available online: https://arxiv.org/abs/1307.8182 (accessed on 20 December 2019).

[10] Ghanem, M.; Chen, T. Reinforcement Learning for Intelligent Penetration Testing. In Proceedings of the WS4 the World Conference on Smart Trends in Systems, Security and Sustainability, London, UK, 30–31 October 2018.

[11] Backes, M.; Hoffmann, J.; Kunnemann, R.; Speicher, P.; Steinmetz, M. Simulated Penetration Testing and Mitigation Analysis. arXiv 2017, arXiv:1705.05088.

[12] Durkota, K.; Lisy, V.; Bosansk, B.; Kiekintveld, C. Optimal network security hardening using attack graph games. In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI-2015), Buenos Aires, Argentina, 25–31 July 2015.

[13] Obes, J.; Richarte, G.; Sarraute, C. Attack planning in the real world. arXiv 2013, arXiv:1306.4044.

[14] Meuleau, N.; Kim, K.; Kaelbling, L.; Cassandra, A. Solving POMDPs by searching the space of finite policies. In Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, Bellevue, WA, USA, 11–15 July 2013.

[15] Spaan, M. Partially Observable Markov Decision Processes, Reinforcement Learning: State of the Art; Springer: Berlin/Heidelberg, Germany, 2012.

[16] Sarraute, C.; Richarte, G.; Hoffmann, J. Simulated penetration testing: From Dijkstra to aaTuring Test++. In Proceedings of the 25th International Conference on Automated Planning and Scheduling, Israel, 7–11 June 2015.

[17] Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay, Google DeepMind. arXiv 2015, arXiv:1511.05952.

[18] Grande, R.; Walsh, T.; How, J. Sample efficient reinforcement learning with gaussian processes. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1332–1340.

[19] Agrawal, S.; Jia, R. Optimistic posterior sampling for reinforcement learning: Worst-case regret bounds. In Proceedings of the Annual Conference on Neural Information

Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1184–1194.

[20] Keshri, A. What is Automated Penetration Testing? Difference between Automatic \& Manual Pentesting. Available online: https://www.getastra.com/blog/security-audit/automated-penetration-testing/.

[21] Imperva. Penetration Testing. Available online: https://www.imperva.com/learn/application-security/penetration-testing/.

[22] Synopsys. Penetration Testing. Available online: https://www.synopsys.com/glossary/what-is-penetration-testing.html.

[23] Intersoft Consulting. General Data Protection Regulation (GDPR). Available online: \url{https://gdpr-info.eu/.