iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** IV **Month of publication:** April 2024

DOI: https://doi.org/10.22214/ijraset.2024.61113

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue IV Apr 2024- Available at www.ijraset.com

An AI-Based Approach for Automating Penetration Testing

Manaswi Patil[1], Devaki Thakare[2], Arzoo Bhure[3], Shweta Kaundanyapure[4], Dr. Ankit Mune[5]
Department of Computer Science and Engineering, Prof. Ram Meghe Institute of Technology and Research Badnera-

Abstract: ***Cyber penetration testing (pen-testing) is important in revealing possible weaknesses and breaches in network systems that can ultimately help in curbing cybercrimes. Nevertheless, even with the current drive to mechanize pen- testing, there are still a number of challenges which include incomplete frameworks and low precision in automation methods. This paper aims at addressing them by suggesting hybrid AI-based automation framework specifically for Pen- Testing through integration of smart algorithms and automated tools. As indicated by recent studies, it goes further into proposing a holistic approach towards maximizing the efficiency as well as effectiveness of Pen-Testing processes. Furthermore, it also identifies the need for machine learning techniques such as reinforcement learning and deep reinforcement learning for automating Pen-Testing activities. MITRE ATT&CK Framework being utilized within the proposed model imitates real-life cyber-attacks and exploits hence facilitating automated testing across diverse target networks. Based on comparison with manual penetration testing reports, this study reviews how effective the new automated method is when compared to old ways used in manual penetration tests while providing some direction for future developments along with suggestions in the field of self- governing intrusion detection systems (IDS).***

Keywords: ***Penetration Testing; Machine Learning; Algorithm; Framework; Cyber Security***

I. INTRODUCTION

Organizations face a dual challenge in the expanding digital landscape. Technology drives innovation but also exposes vulnerabilities to cyber threats. Penetration Testing (pen-testing) emerges as a vital defense, simulating real-world attacks by malicious actors. Yet, achieving thorough and accurate pen-testing is complex. Manual testing offers precision but is time and resource-intensive. Automation streamlines this process, with recent studies exploring Machine Learning (ML) and Artificial Intelligence (AI) integration for quicker vulnerability assessments.

Pen-testing follows a defined lifecycle: planning, reconnaissance, exploitation, vulnerability identification, and post-testing activities. It can be manual, offering detailed examination but demanding skilled professionals and time. Automated tools are faster but lack flexibility. Defensive security fortifies perimeters and installs security software, while offensive security, including pen-testing, proactively identifies system weaknesses. Pen-testing is crucial for identifying entry points and attack vectors, aiding in security prioritization and system hardening. Traditional methods fall short, but ML and AI promise more efficient and comprehensive techniques.

II. BACKGROUND

A. Penetration Testing Life Cycle

Penetration testing, often referred to as pen testing or ethical hacking, is a proactive approach employed by cybersecurity professionals to assess the security posture of an organization's digital infrastructure. This rigorous practice involves simulated attacks on the organization's network, applications, and systems to uncover vulnerabilities that malicious actors could exploit. By mimicking the tactics of real-world attackers, penetration testing aims to identify weaknesses before they can be leveraged for malicious purposes, thereby enabling organizations to strengthen their defenses proactively.

As illustrated in the figure, the penetration testing lifecycle encompasses a series of structured steps aimed at identifying and exploiting vulnerabilities within an organization's network and systems. This lifecycle typically consists of the following phases:

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue IV Apr 2024- Available at www.ijraset.com

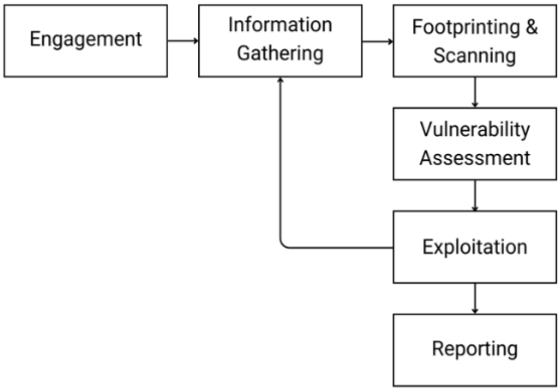


Figure 1. Penetration Testing Life Cycle

1) Engagement:

Define the scope, objectives, and rules of the penetration test, including engagement type (White Box, Black Box, Grey Box), time frame, and targets (IP addresses, domains, etc.). Negotiate the fee based on the complexity of the engagement, time-consuming factors, and the number of targets to be tested.

Establish the scope of engagement in terms of IP addresses, network blocks, domain names, or any other relevant information.

2) Information Gathering:

Information gathering is one of the first and most vital things in successfully conducting a penetration test. Until a few years back, getting to know about a company simply required reading newspapers. Besides, you could look for contacts from the telephone directory while others would visit the corporate's website to find out more about their systems as well as products.

Today, people can now do this faster by employing information from social media sites, public websites or business websites. In fact, professionals who carry out penetration testing may also conduct similar searches on popular social networking sites like Facebook, Twitter, LinkedIn, Google Plus etc.

CrunchBase is another information-gathering tool which provides comprehensive data about IT start-up owners, venture capitalists, employees and acquisitions. The other useful sources are the Who is data base and the system for award management (SAM), as well as GSA eLibrary where you one find contracts between private companies with the U.S. Government. Such information includes Fig. 2 which shows owner name, street address, and email address.

3) Footprint & Scanning:

When penetration testers have gathered information about the target organization, they move to fingerprinting and enumerating nodes on client's network that is an infrastructure component of OSINT. Ping Sweeping, also called the ping command, is a tool aimed at ascertaining if a machine is alive in a network. The ping can be performed from the command prompt (Fig.3) of any major operating system.

On another hand Fping (shown in Fig.3) is a Linux program that is an advanced version of Ping software You can undertake ping sweeps using Fping. Fping gives users more choices. For instance, -g prompts the software to sweep for pings rather than perform standard pings while -a causes the program to list only live hosts.

4) Vulnerability Assessment:

After the foot-printing and scanning stage, the vulnerability assessment process looks for vulnerabilities in applications and network infrastructure. In addition, it is speedier and less resource intensive than this phase. Scanners employ probes to scan TCP and UDP ports operable by other daemons as well.

- Design software network devices operating systems, etc.
- Windows registry

The manufacturer of scanner updates the database regularly with new security audits and vulnerability signatures. The scan results will be more improved so long as its database remains current. Nessus is a well-known vulnerability scanner that has a server-client architecture. Web interface for configuring scans is provided by its client component.

Figure .2 Whois Database[24]

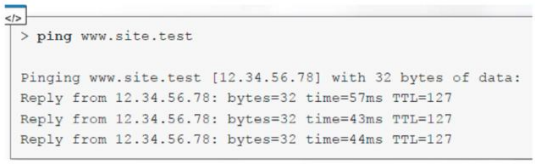


Figure 3. FPing Sweeping[24]

To achieve this purpose, it launches probes into applications and systems, gathers response from them and compares them with its vulnerability database.

5) Exploitation:

Exploitation involves the use of vulnerabilities identified in the vulnerability assessment process. In exploitation, a penetration tester examines a vulnerability before extending and elevating their access rights to the target networks and systems. Exploitation has two types:

a) *Web application attacks:* Challenges in web security are numerous. Primarily, internet applications are inherently complicated. The Open Web Application Security Project (OWASP) is assigned to find out how this can be done. OWASP refers to an open-source web application security project. OWASP helps assess levels of web application security. OWASP seeks to create awareness about software security and enable organizations make better choices on it.[6]. For example, the attacker may engage in dictionary assaults. Dictionary attacks represent another way that automates password attacks.

b) *System Attacks:* Another kind of exploitation is system attacks. "Malicious software" or malware constitutes one type of system attack aimed at:

- Causing denial of service
- Spying on user activitie
- Gaining unauthorized control over one or more computer systems
- Undertaking other malicious activities
- Examples include viruses, worms, spyware and Trojan horses among others.

Users passwords are at a risk when they are in the format of clear texts. An attack on the password system will take place if an intruder gains access to just one network computer. A number of techniques for preventing password attacks include cryptographic hashing functions. Additionally, cryptographic hashing functions are used to convert a password into encrypted and safe-to-store form. Trying to manually crack a password is very unlikely. Two main ways of automating these procedures are as follows:

Brute force attacks are the only sure way of getting someone's password. To automate a brute force attack, create a programme that generates all possible passwords like is shown below in Fig 4.

```
password_found = false
password_length = 1

while password_found == false
do
    while can_create_password_of_length(password_length)
    do
        password = create_password_of_length(password_length)
        if (hash(password) match attacked_hash)
        then
            password_found = true
    done
    password_length = password_length + 1
done
```

Figure 4. Brute force code[24]

Dictionary assaults: Another way to automate password attacks is using dictionary attacks. However, dictionary attacks do not try every possible password as others would do since they run against a set of previously identified passwords stockpiled in memory. Such stockpiling demands information regarding hashed passwords.
- A dictionary or wordlist of passwords.
- A tool for checking each password in the wordlist against the user's file

6) Reporting:

Compile a detailed report outlining the findings of the penetration test, including detected flaws, associated risks, and recommendations for mitigation. Present the report to the client, providing specific details about vulnerabilities, their impact on the organization, and actionable advice for remediation.

Include an executive summary highlighting the objectives and results of the test, as well as a technical report detailing the methodology and findings. Collaborate with the client to ensure a clear understanding of the findings and facilitate the implementation of remediation measures.

B. Manual vs Automation Penetration Testing

Manual and automated testing are two vital methodologies utilized in the evaluation of vulnerabilities within a network [7], [8]. Each method offers unique advantages and is adept at detecting different types of vulnerabilities across various settings, including those outlined in the sample instances discussed above. This section will delve into the nuanced disparities between manual and automated testing methodologies, elucidating their respective strengths, limitations, and suitable scenarios for implementation. These distinctions will be presented comprehensively in Table [1], providing a clear reference point for cybersecurity practitioners and stakeholders alike.

TABLE I
MANUAL VS. AUTOMATION PEN-TESTING

	Automated Pen-testing	**Manual Pen-testing**
Advantage	- Web apps get examined quicker. - The maintenance of an updated database is used to write attack codes for various systems.	- Humans can detect flaws that automated systems miss. -False positives are detected and removed
Disadvantage	- False positives and negatives are frequently encountered. - cannot identify weaknesses in unusual circumstances.	- Slower process. - Rewriting attack programs to work on various platforms requires manual database upkeep.
Testing process	- Reduce human error and fatigue by using a regular procedure and repeated testing.	- unconventional, costly to modify, and capital-intensive.
Training process	Easy	Customisable, time consuming and expensive.
Tools	Burp suite , Nmap, Metasploit.	Manual

III. SURVEY ANALYSIS

A. Penetration Testing Techniques

Using a machine learning approach, the authors [1] have devised a penetration testing technique that involves obtaining the root shell of a target within a Meterpreter session, searching for vulnerabilities, and utilizing machine learning to find the most effective exploit. They use a decision tree technique to choose which exploit is the best. Two concepts are used by the decision tree algorithm:

1) *Information Gain:* choosing the testing feature for the node division based on the maximum entropy reduction at that specific node. (Equation l).

$$\text{Gain}(S, A) \equiv \text{Entropy}(S) - \sum_{v \in V \text{ alues } (A)} \frac{|S_v|}{|S|} \text{Entropy } (S_v) \quad (1)$$

2) *Gini impurity:* We can compute a weighted sum of impurities for each partition in addition to doing a binary split for each attribute to lower the possibility of classification error. (Equation 2).

$$G_I = 1 - \sum_{i=1}^{k} p(i|n)^2 \quad (2)$$

On the one hand, as Fig. 5 illustrates, the decision tree has a 33% accuracy rate to determine the appropriate exploit. However, even a slight alteration in the training set will result in a notable shift in the prediction, making the decision tree highly sensitive. This also suggests that the variance of the model will be relatively large. The model's prediction for the new data will be off in this case.

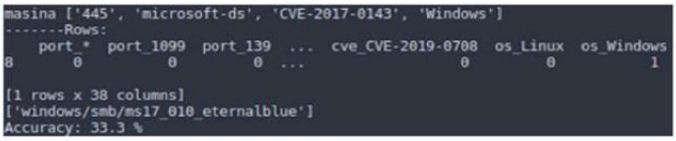


Figure 5. Decision tree result[24]

In a different method described in [2], the authors employed automated penetration testing and Reinforcement Learning technology to find numerous exploits on the target computer in an astute manner. In specifics, they separated their experiment into training and testing cases, using the Q- learning and Deep Q-learning algorithms in each case.

- Q-learning: the state and action are utilised to create the Q-table. The actions that specify the modules of the Metasploit framework are kept in a row within the Q-table. On the other hand, a state will be defined for the attributes of the target machine, such as its port number, service name, etc.
- Deep Q-learning: This technique is similar to Q- learning but relies on a neural network to approximate the Q-values for each action based on a state rather than a Q-table. Consequently, during training, the search strategy gives the Q-learning agent (Fig. 6) an erroneous, which leads the exploit module to find false vulnerabilities.

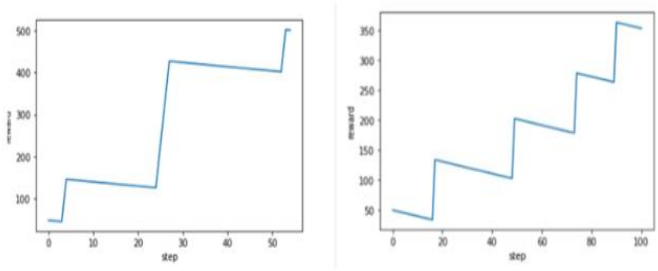


Figure 6. Q-learning Training [1]

The scenario's estimation is displayed in Figure 6, where testing against Metasploitable 2 requires 70 steps to complete and yields six exploits. It is better than testing without any applied algorithms. Additionally, redundancies in the exploit module were found as a result of Q¬ learning's training.

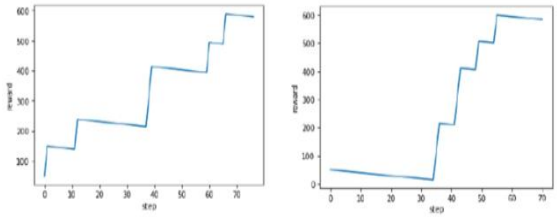


Figure 7. Q-learning Training [2]

These days, we may utilise Android operating systems with smartphones, tablets, smartwatches, and other devices that we use on a daily basis. Consequently, to detect persistent infection, the authors in [3] used Convolutional Neural Networks (CNN), Bidirectional Long Short-Term Memory (BiLSTM), and Gated Recurrent Units (GRU) on the Android platform. First, Android APK is used for data collecting, and data sets such as Androzoo [9] and Android Malware Dataset (AMD) [10] are extracted. In addition, the planned dataset is divided into three categories: Backdoors, Trojans, and Benign. Subsequently, pre-processing entails training the features that were extracted from the datasets using CNN-BiLSTM, CNN-LSTM, and CNN-GRU algorithms to obtain better predictions. In order to quantify performance, the authors lastly assessed recollection, testing, training time, accuracy, precision, and F1-score. Additionally, they employ 10-fold cross-validation to provide impartial findings, and they compute the average to display the cumulative performance for each fold. by using the 10-folds and the planned dataset with the three algorithms. The experimental results show that CNN- BiLSTM achieved the best performance with an accuracy of 99.2% because BiLSTM offers two layers—Forward and Backward layers—for assessing in both directions for highest detection rate. In contrast, LSTM simply retains historical data, or what is referred to as the Forward direction, which produces subpar predictions [11].

Penetration testing, part of the Belief-Desire-Intention (BDI) paradigm, utilizes outside tools to identify vulnerabilities. In dynamic networks, the BDI model acts as an agent, guiding tasks from information gathering to reporting. Simulations demonstrate interactions between the BDI agent and target, showing successful investigations and attacks. Integration of the BDI model and automated pen-testing streamlines reporting.Penetration testing methods include black box, white box, and grey box testing. Black box testing simulates real cyber-attacks by identifying vulnerabilities from an external perspective. White box testing examines internal workings using source code analysis. Grey box testing provides attackers with additional information, reducing reconnaissance time.

A framework collects network data from Shodan and uses MulVAL to create attack trees, simplifying them into a matrix. Scores are assigned based on vulnerability severity and predefined actions. DQN, an AI algorithm, navigates the matrix as an attacker, aiming to maximize the total score by reaching the target server.

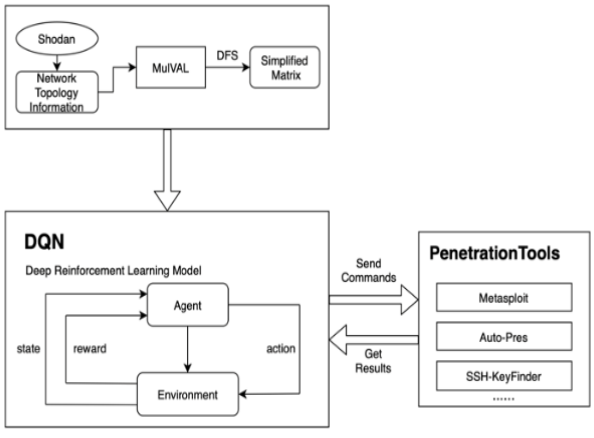


Figure 8. The architecture of the automated penetration testing framework using DRL.[22]

In deep learning applications, training data collection involves three steps: gathering real network information, building attack trees, and preprocessing data into a DQN model while ensuring privacy. The DQN module uses a simplified attack matrix to learn optimal attack paths, with rewards calculated based on vulnerability scores. Penetration tools like Metasploit facilitate communication with real networks, coordinated with the DQN model's output through a wrapper for decision-making.Testing the framework showed promising results, with DQN achieving 86.3% accuracy in identifying optimal attack paths. Integrating Reinforcement Learning (RL) with the PenBox automated penetration testing framework aims to maximize overall reward through Q-learning. PenBox, an automated security testing tool, integrates seamlessly with ESA's Ground Segment Reference Facility, offering diverse attack scenarios and real-time impact analysis.

PenBox's modular design enables diverse attack strategies and flexible scenario definition, but interoperability with diverse tools poses technical challenges. Future improvements may include integration with machine learning techniques to automate decision-making and enhance testing procedures, prioritizing relevant tools and optimizing results analysis.

Test	lr	training_steps	batch_size	replay_size	exploration_steps	gamma	target_update_freq	Actions	Time	Success
1	0.01	20000	32	100000	10000	0.05	1000	703	35 s	Y
2	0.01	10000	32	100000	2000	0.05	1000	1868	1 m 33 s	Y
3	0.01	1000	32	100000	500	0.05	1000	2000	21 s	N
4	0.01	100000	32	100000	500	0.05	1000	601	8 m s	Y
5	0.01	100000	128	10000	500	0.05	1000	1410	15 m 18 s	Y
6	0.00001	100000	8	100000	500	0.05	1000	1247	31 s	Y
7	0.0001	100000	8	100000	500	0.05	1000	601	8 m s	Y
8	0.001	10000	8	100000	500	0.05	1000	1867	1 m 3 s	Y
9	0.01	100000	8	1000	500	0.05	1000	1936	7 m	Y
10	0.001	100000	32	1000	500	0.05	500	2000	9 m 10 s	N
11	0.001	100000	32	100000	500	0.05	500	2000	16 m 11 s	N
12	0.01	10000	32	10000	20000	0.6	400	2000	33 m 12 s	N
13	0.1	10000	32	10000	20000	0.7	300	2000	12 m 11 s	N
14	0.01	100000	32	10000	10000	0.8	300	2000	19 m 32 s	N
15	0.1	100000	32	1000	5000	0.3	200	1342	16 m 4 s	Y
16	0.01	100000	32	1000	5000	0.25	200	1549	19 m 5 s	Y
17	0.01	200000	16	500	1000	0.2	400	876	12 m 50 s	Y
18	0.01	200000	16	400	700	0.1	500	190	13 m	Y
19	0.01	200000	16	400	500	0.09	600	84	22 34 s m	Y
20	0.01	200000	16	400	500	0.08	700	33	15 m 12s	Y

Table 2. Reinforcement learning result[23]

B. Penetration Testing Tools

Penetration testing tools are used throughout the PT life cycle. Manual and automated approaches can also be employed using these tools, and some of them are quite beneficial because they are open source. Table [II] provides an overview of the most frequently utilized technologies [15] for information gathering, vulnerability detection and exploitation.

During our investigations we examined seven such tools and explained their traits based on available literature.

- Nmap: Nmap is a network vulnerability scanner which gathers intelligence on IP addresses and open ports of hosts.
- Burpsuite: A security testing tool for web applications that unearths vulnerabilities.
- Wireshark: Looks at the micro interactions of the network.
- Metasploit: Get past the security defences and drop a payload, which takes action on the victim machine.
- Nessus: The system sends probes to systems and applications and compares responses against its vulnerability database to perform scans.
- Intruder: Software version identification is performed using both passive and active checks in scans. Passive checks employ software fingerprinting techniques while active checks search for known vulnerabilities.
- Netsparker: Exploits vulnerabilities to assess trueness of the results.

This survey aims to correlate and compare all the pentesting means available, to find the most efficient and low cost way to execute exploits. The objectives of this survey are represented in the following figure:

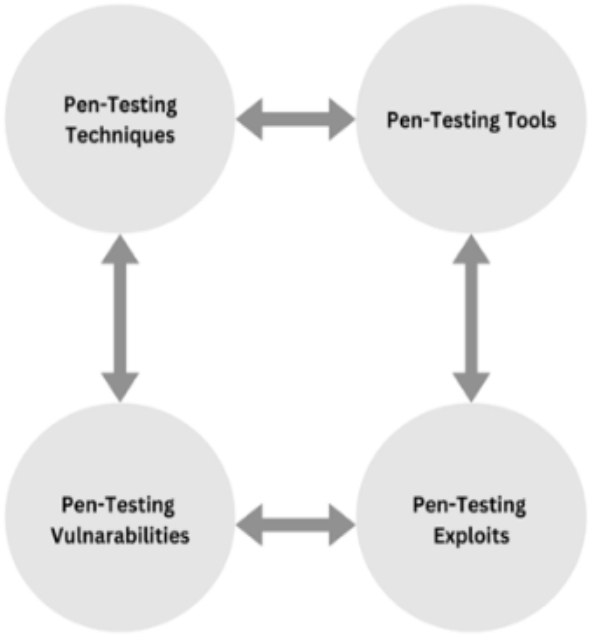


Figure 9. Survey's Objectives

IV. METHODOLOGY

In the area of cybersecurity, it is good to try out the development and assessment of a comprehensive framework that is based on AI and automates penetration testing tasks. Analysis for existing AI techniques was initiated. It identified which were most appropriate methods for different task in penetration testing cycle. Such as, learners that are supervised seemed to be a powerful way for training AI models against datasets that have labelled vulnerabilities. These models can identify likely points of access with greater accuracy by studying system behaviour and network configurations, thereby saving security professionals time and resources.

Another significant technique considered here was reinforcement learning. The approach uses an AI agent interacting with a simulated network environment (Leung etal., 2014). Over time the agent will learn how to navigate across the network finding more efficient ways to exploit any vulnerability along its path (Tabari et al., 2015). With this analogy in mind consider the agent as a student enrolled in virtual course on network security whereby he/she practices continuously and refines his/her skills. Consequently, applying this method during real world penetration test has significantly decreased both efforts and time spent on searching an exploitation path.

Then, the researchers designed specific AI models for each task they adopted. This entailed a painstaking process of data incubation. They assembled real world penetration testing data with known exploit techniques into comprehensive datasets. These sets were then applied in the training of models. One can imagine how you feed these models thousands of examples of systems vulnerable through which exploits have been made successfully. Consequently, this helps in teaching the models how to recognize vulnerabilities, prioritize the most effective exploit paths and identify anomalies that might suggest a security breach.

The researcher also probed into unsupervised learning for anomaly detection. This method is concerned with spotting departures from the regular system functioning. By inspecting network traffic and system logs, it is possible for this model to detect potential security incidents that may not be easily identified using traditional techniques. The models are like ever-alert sentries scanning for any suspicious activities that could indicate an ongoing cyber- attack.

Researchers integrated them into a fully-fledged penetration testing framework to demonstrate how the skills acquired from trained models could be practically applied in the real world. An automated penetration test tool, this framework serves as a central point that manages all of the penetration test operations associated with various security assessments. In this way, AI models are smoothly knit into every stage of the foregoing tests for them to deliver their expertise. To envision such an incident, it is similar to leading an orchestra by a conductor where each AI model has its own role towards general security assessment.

This was followed by a comprehensive evaluation of how effective the AI-driven penetration testing framework had been implemented so far. It involved employing robust testing techniques on a number of target systems having known vulnerabilities. As such, these systems acted as practical playgrounds for assessing how well it performed through different measures of performance. The workout didn't mean success only but also aimed at finding areas needing improvement. The researchers went through results and based on what they found, took a step back and started refining and retesting things again until there were no more weaknesses in its functioning.

These insights resulted in an iterative approach to fine-tuning that required changing either some settings within the artificial intelligence models or even modifying data used for training itself as well as entire editing process run within testing frameworks if necessary. Through this continuous loop of evaluation and refinement stage-adjustments are made to the AI models, the testing framework itself as well as potentially data used for training. The researchers aimed at enhancing overall effectiveness and robustness of the AI-driven approach to penetration testing through this continuous loop of evaluation and refinement. .

V. CONCLUSION

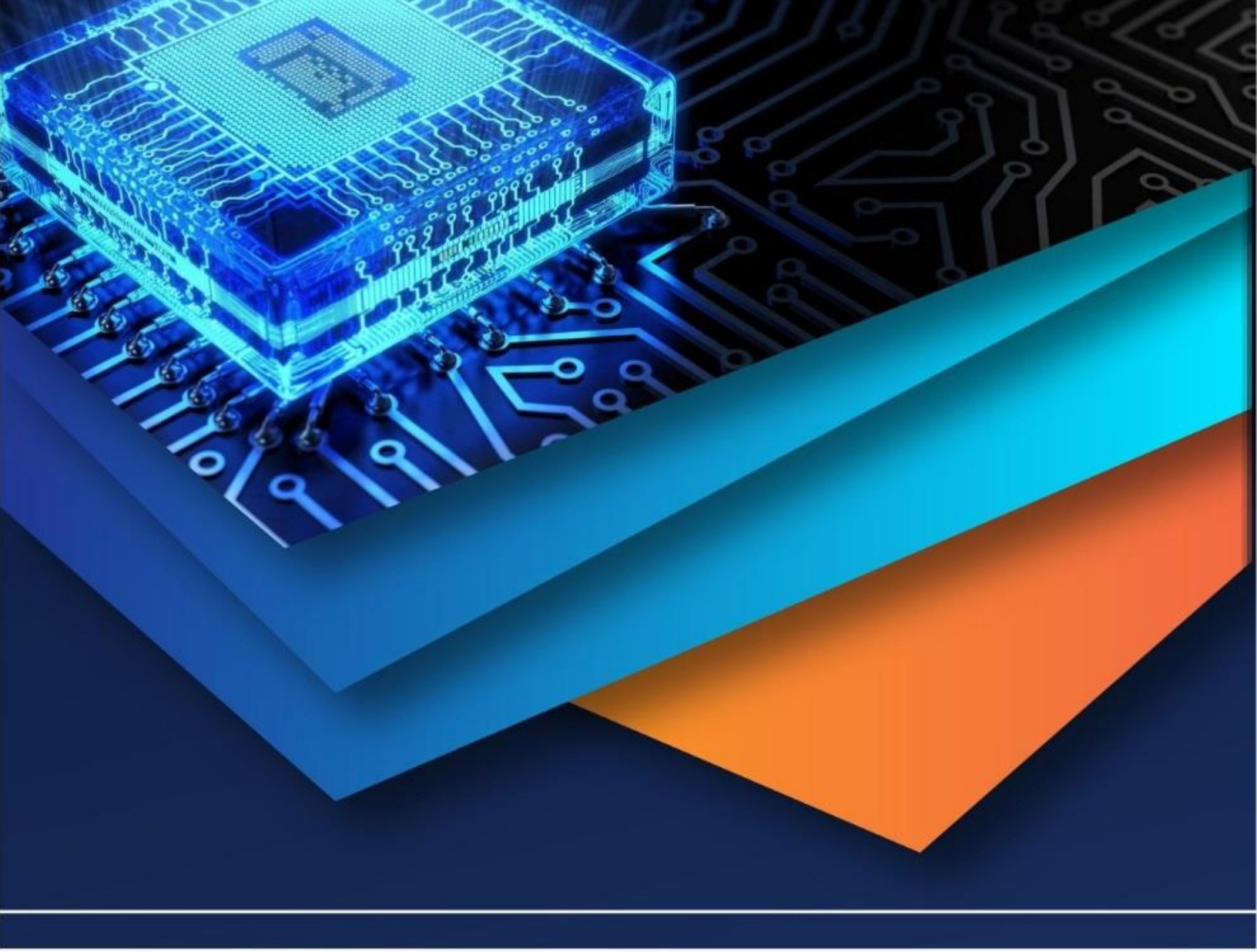
Penetration testing is crucial for safeguarding digital infrastructure by simulating cyber-attacks to identify vulnerabilities. Traditional methods are time and resource-intensive, leading to the adoption of specialized tools and algorithms for efficiency. These tools automate tasks in the pen-testing lifecycle, such as scanning networks and analyzing results, enhancing speed. Machine learning algorithms, like decision trees, improve accuracy but may require significant computing power.

Recent research focuses on identifying effective algorithms for pen-testing, highlighting the success of Belief-Desire-Intention (BDI) and Bidirectional Long Short-Term Memory (BiLStM) in vulnerability detection. However, advanced research is needed for broader vulnerability detection, often requiring reinforcement learning despite high computational costs. Understanding algorithm strengths and weaknesses allows security professionals to employ automated pen-testing tools that balance efficiency and accuracy, enhancing overall cybersecurity posture.

REFERENCES

[1] 0. Valea and C. Opri§a, "Towards pen-testing automation using the Metasploit framework," in 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 171- 178, IEEE, 2020.
[2] K. C. Goh, Toward Automated Penetration Testing Intelligently with Reinforcement Learning. PhD thesis, Dublin, National College of Ireland, 2021. 379 Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY BOMBAY. They were downloaded on January 19, 2024, at 07:07:36 UTC from IEEE Xplore. Restrictions apply.
[3] I. U. Haq, T. A. Khan, and A. Akhunzada, "A dynamic robust dl-based model for android malware detection," IEEE Access, vol. 9, pp. 74510-- 74521, 2021.
[4] G. Chu, Automation of Penetration Testing. The University of Liverpool (United Kingdom), 2021.
[5] A. Sultan, "eLeamSecurityJunior Penetration Tester," 2023.
[6] A. Cordelia, L. Bononi, and F. Crin, "Web application penetration testing: an analysis of a corporate application according to owasp guide lines," ALMA MATER STUDIOUM UNIVERSITY OF BOLOGNA, 2018.
[7] F. Abu-Dabaseh and E. Alshammari, "Automated penetration testing: An overview," in The 4th International Conference on Natural Language Computing, Copenhagen, Denmark, pp. 121-129, 2018.
[8] E. A. Altulaihan, A. Alismail, and M. Frikha, "A survey on web application penetration testing," Electronics, vol. 12, no. 5, p. 1229, 2023 .
[9] K. Alx, T. F. Bissyande, J. Klein, and Y. Le Traon, "Androzoo: Collecting millions of android apps for the research community," in Proceedings of the 13th international conference on mining software repositories, pp. 468-471, 2016.
[10] F. Wei, Y. Li, S. Roy, X. Ou, andW. Zhou, "Deep ground truth analysis of current android malware," in Detection of Intrusions and Malware, and Vulnerability Assessment: 14th International Conference, DIMVA 2017, Bonn, Germany, July 6-7, 2017, Proceedings 14, pp. 252-276, Springer, 2017.
[11] S. Liaqat, A. Akhunzada, F. S. Shaikh, A. Giannetsos, and M. A. Jan, "Sdn orchestration to combat evolving cyber threats in internet of medical things (iomt)," Computer Communications, vol. 160, pp. 697- 705, 2020.
[12] G. Chu and A. Lisitsa, "Poster: Agent-based (bdi) modeling for automa- tion ofpenetration testing;' in 2018 16th Annual Conference on Privacy, Security and Trust (PST), pp. 1-2, IEEE, 2018.
[13] M. Franois, P.-E. Arduin, and M. Merad, "Artificial intelligence & cybersecurity: A preliminary study of automated pentesting with offensive artificial intelligence"
[14] M. Liu and B. Wang, "A web second-order vulnerabilities detection method," IEEE Access, vol. 6, pp. 70983-70988, 2018.
[15] I. U. Haq and T. A. Khan, "Penetration frameworks and development issues in secure mobile application development: A systematic literature review," IEEE Access, vol. 9, pp. 87806-87825, 2021.
[16] N. Singh, V. Meherhomji, and B. Chandavarkar, "Automated versus manual approach of web application penetration testing," in 2020 11th International Conference on Computing, Communication and Network- ing Technologies (ICCCNT), pp. 1--6, IEEE, 2020.
[17] R. Sahani and S. Randhawa, "Clickjacking: beware of clicking," Wireless Personal Communications, vol. 121, no. 4, pp. 2845-2855, 2021.
[18] A. Alanda, D. Satria, M. I. Ardhana, A. A. Dahlan, and H. A. Mooduto, "Web application penetration testing using sql injection at- tack," JOIV: International Journal on Informatics Visualization, vol. 5, no. 3, pp. 320-3- 26, 2021.

[19] S.Sodagudi,S.K.Kotha,andM.D.Raju,"Novelapproachestoidentify and prevent cyber-attacks in web," in 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), pp. 833- 839, IEEE, 2019.
[20] S . M, "https://www.simplileam.com/tutorials/cyber-security-tutoriall," 2023.
[21] S. Nagpure and S. Kurkure, "Vulnerability assessment and penetration testing of web application," in 2017 International Conference on Com- putting, Communication, Control and Automation (ICCUBEA), pp. 1--6, IEEE, 2017.
[22] Z. Hu, R. Beuran and Y. Tan, "Automated Penetration Testing Using Deep Reinforcement Learning," 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Genoa, Italy, 2020, pp. 2- 10, doi: 10.1109/EuroSPW51379.2020.00010
[23] A. Confido, E. V. Ntagiou and M. Wallum, "Reinforcing Penetration Testing Using AI," 2022 IEEE Aerospace Conference (AERO), Big Sky, MT, USA, 2022, pp. 1-15, doi: 10.1109/AERO53065.2022.9843459.
[24] V. Saber, D. ElSayad, A. M. Bahaa-Eldin and Z. Fayed, "Automated Penetration Testing, A Systematic Review," 2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), Cairo, Egypt, 2023, pp. 373-380.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY