Reinforcement Learning-Based Adaptive Penetration Testing Framework for Wireless Communication

Saken Tleuberdin[1], Konstantin Malakhov[2], Nurlan Tashatov[3], Dina Satybaldina[4], Didar Yedilkhan[5]
Research Institute of Information Security and Cryptology, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan[1, 2, 3, 4]
Smart City Center, Astana IT University, Astana, Kazakhstan[5]

***Abstract*—Wireless Fidelity (Wi-Fi) technology is widely used in the environment of Internet of Things (IoT), and the importance of security assessment has increased. Nowadays, Wi-Fi security assessment is based on security tools that are operated manually, and some methods face issues due to the lack of automation. In this study, we suggest a method for adapting Wi-Fi penetration testing in which a reinforcement learning (RL) agent interacts with the environment by choosing actions based on the current state to maximize the total reward received. We model a tabular Q-learning algorithm as an agent interacting with the wi-fi environment. The action space is made up of denial-of-service attacks, while the environment state vector includes parameters of the network and indicators of attack success, which all contribute to the reward function. The experiments show that the RL agent successfully finds vulnerabilities in the Wi-Fi Protected Access 2 (WPA2) and Wi-Fi Protected Access 3 (WPA3) protocols.**

***Keywords*—Internet of things; security; penetration testing; reinforcement learning; wireless communication**

I. Introduction

The large-scale implementation of smart city projects worldwide is significantly driven by the pervasive adoption of Wireless Fidelity (Wi-Fi) technology. This technology serves as the basic layer for Internet of Things (IoT) devices and services. Wi-Fi networks make up about a third of all connections to the global Internet of Things system. Over the next few years, the number of connected devices is expected to grow by more than 15% each year [1]. But the widespread use of IoT systems leads to a significant increase in cybersecurity threats (by 400% overall, of which 54% of incidents are related to the Industrial Internet of Things) [2]. This motivates research into the security of Wi-Fi technologies in modern digital ecosystems of smart cities.

The evolution of Wi-Fi security protocols, from the vulnerable Wired Equivalent Privacy (WEP) to Wi-Fi Protected Access 2 (WPA2) and the more recent Wi-Fi Protected Access 3 (WPA3), shows a consistent desire to enhance the security of wireless communication. Wired Equivalent Privacy (WEP) utilized the Rivest Cipher 4 (RC4) stream cipher with a 24-bit initial vector. This led to the vulnerable key reuse in the case of statistical attacks that could recover the encryption keys in just a few minutes. These flaws prompted the development of Wi-Fi Protection Access (WPA) as a temporary solution to the problem. This protocol introduced the Temporal Key Integrity Protocol (TKIP) in order to address the issue of key re-use and fake packets.

The implementation of WPA2, which is standardized under Institute of Electrical and Electronics Engineers (IEEE) 802.11i, represented a significant advancement because of its replacement of RC4/TKIP with an Advanced Encryption Standard (AES) based on Counter Mode with Cipher Block Chaining Message Authentication Code Protocol (CCMP). CCMP enhanced the confidentiality, integrity, and replay protection, which increased the resilience of the system against digital attacks. However, despite its impressive strength, WPA2 was not invulnerable to flaws. One of the greatest limitations was the exposed vulnerability through the Key Reinstallation Attack, commonly known as KRACK, which utilized flaws in the four-way handshake process to reinstall the already-in-use session keys, enabling the transmission of packets, decryption, and manipulation of traffic without recovering the master key [3].

As a result of these limitations, WPA3 was incorporated into the Wi-Fi Alliance in 2018 to augment the authentication's resilience and secrecy. WPA3 replaces the Pre-Shared Key (PSK) mechanism in WPA2-Personal, which is used to authenticate keys. This mechanism is resistant to dictionary attacks that take place offline. Even if a password is frail, hackers can no longer take part in a handshake and attempt to guess numbers without limit. Also, WPA3 provides forward secrecy, which means that the loss of long-term credentials doesn't lead to the previously captured encrypted traffic being exposed.

For organizations with multiple branches, WPA3-Enterprise increases the security of the system further by supporting the 192-bit cryptographic strength associated with the Commercial National Security Algorithm (CNSA) suite. Other enhanced security features, such as Protection Management Frames (PMF), are also required in WPA3, which combats the deauthentication and disassociation attacks that were common in WPA2 deployments.WPA3 improves key exchange and management frame protection by integrating Simultaneous Authentication of Equals (SAE) and Management Frame Protection (MFP) [4]. Recent analysis, however, indicates that the WPA3 authentication protocol has a Dragonblood vulnerability that makes it possible for a range of denial of service (DoS) and security downgrade attacks [5].

Manual inspection and previously determined attack scenarios are often used in traditional penetration testing methodologies [6, 7, 8]. In dynamic network environments, this approach is not very effective.

We present a framework for intelligent wireless network security analysis using a reinforcement learning algorithm. Our approach exploits the ability of an agent to change its policy based on the analysis of the environment state as a result of interactions with the agent.

The contributions of this study are as follows.

- We propose an adaptive Wi-Fi penetration testing methodology based on a Q-learning algorithm to effectively find vulnerabilities against denial-of-service attacks.
- We present a software tool for real-time wireless network auditing, the functionality of which includes selecting and launching attacks, collecting and analyzing data, and retraining the agent based on reward calculation.
- We test our approach and tool method in a real network environment and show how intelligent attack strategies work differently in WPA2 and WPA3 protocols, and identify new vulnerabilities in WPA3.

The rest of the study is organized as follows. Section II highlights the related work. Section III presents the system design and model. Section IV presents the experimental methodology. Section V provides results and evaluation. Section VI concludes the work with future directions.

II. Related Work

The most recent state-of-the-art reinforcement learning research for penetration testing can be highlighted in works that apply a hierarchical deep reinforcement learning framework incorporating expert knowledge to manage complex state spaces and sampling efficiency of the automated process of vulnerability exploitation [9].

Because it does not allow any abnormal packet to transmit over the network, which would consume extra resources from normal behavior, the Deep Q-Network-based implementation enhanced by noisy nets, prioritized experience replay and intrinsic curiosity module has been able to scale up automatic penetration testing due to the sparse reward problem and large action space in big environments [10].

NASimEmu provides both simulator and emulator environments under a common interface [11]. NASimEmu allows RL agents to be trained in a simulation that can successfully transfer to structurally different real scenarios. This helps bridge the reality gap between training and deployment.

Security research shows autonomous pentesting as a Markov Decision Process (MDP) and applies model-free RL (tabular/neural Q-learning) for discovering optimal attack paths in a simulated network environment without any pre-defined model of the environment, thereby demonstrating that RL is capable of outperforming classical model-based planners, with noted limitations on scalability to larger networks [12].

Our recent study presents an approach to improving the security of the Internet of Things and wireless networks in smart cities by integrating penetration testing with machine learning methods for packet-level anomaly detection [13]. The study demonstrates that modeling and executing attack scenarios using penetration testing tools yields enriched vulnerability data useful for anomaly detection. The adaptive penetration testing framework based on reinforcement learning presented in this study is also part of the aforementioned integrated system. In the first stage, penetration testing generates datasets collected from real networks. The collected datasets serve as the basis for training and testing anomaly detection algorithms, as well as the predictive models used in the process.

III. System Design and Model

Penetration testing (PT) involves utilizing controlled and simulated attacks to identify the vulnerable components of computer systems, communication devices, and wireless networks before they can be utilized by criminals. Unlike the traditional method of vulnerability detection, PT involves interacting with the target system dynamically, which simulates the behavior of a real attacker in order to evaluate both technical flaws and defensive mechanisms.

In this study, we propose that the PT process is modeled as a Markov decision process (MDP) [14], which provides a mathematical framework for the sequential decision-making in uncertain environments. To address this problem, we utilize reinforcement learning (RL) [15,16]. RL allows a character to learn a successful attack strategy via repeated interactions with the target habitat. By observing the state transitions and receiving rewards as a result, the agent's strategy is progressively improved to maximize the expected total return. This method is particularly beneficial for scenarios that involve wireless detection, the environmental dynamics associated with this approach, such as signal changes, user movement, authentication methods (e.g., WPA2/WPA3), and protective measures that introduce ambiguity and partial observability [17].

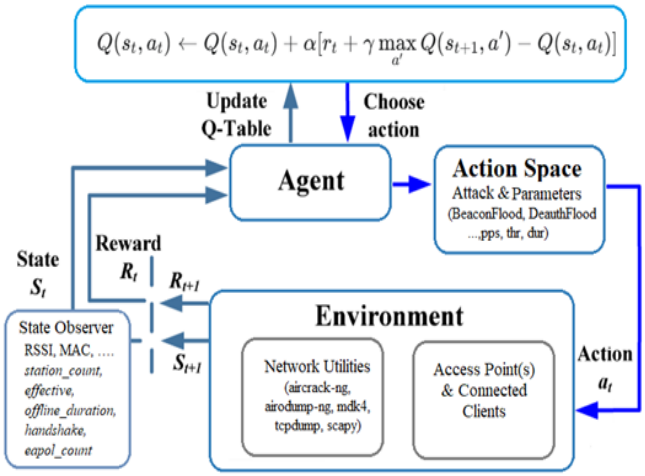

Fig. 1. RL agent for wireless network penetration testing.

By combining the formalization of MDP with the optimization based on RL, the proposed framework allows PT to transition from a manual, driven procedure to an adaptive, intelligent process that chooses the most effective actions based on the changing conditions of the network. The RL agent, adapted for wireless network testing, interacts with the environment as illustrated in Fig. 1. At time step t, the agent

observes the target network using network utilities to perceive the current state st ∈ S. Based on this state, the agent selects an action at ∈ A, representing an attack or a combination of attacks. Following the execution of at, the environment transitions to a new state st+1, and the agent receives a reward signal rt, which quantifies the effectiveness of the chosen action.

In the context of penetration testing, the *State Space* is defined by the observed conditions of the Wi-Fi network at different time instants. The state vector includes common environmental factors collected from network tools, like Received Signal Strength Indication (RSSI), Client Media Access Control (MAC) addresses, Basic Service Set Identification (BSSID), Channel, Extended Service Set Identification (ESSID), and Round Trip Time (RTT). Additionally, it includes specific metrics and indicators reflecting the effectiveness of executed attacks:

- Station_count (the number of unique client MAC addresses detected in the network);
- Effective (the number of station_count clients remaining connected after an action);
- Offline_duration (the number of consecutive episodes during which the effect remains 0);
- Handshake (a flag indicating the detection of a re-authentication handshake during client reconnection attempts);
- Eapol_count (a counter for Extensible Authentication Protocol over LAN (EAPOL) packets, indicating the number of handshake attempts).

The Action Space includes a wide range of attack strategies, created by mixing six types of cyberattacks (explained in Table I) with different settings:

- Pps (packets per second, the intensity parameter for an attack);
- Thr (the number of parallel threads launched for a single attack);
- Dur (a discrete time level of attack duration).

TABLE I. THE ACTION SPACE FOR THE RL AGENT

Attack Type	Description
DeauthFlood	This attack forcibly disconnects clients from a network by continuously sending deauthentication frames. An excessive volume of these legitimate session-termination messages causes connected users to lose their wireless connection.
BeaconFlood	The attacker generates a high volume of spurious beacon frames, which are legitimate messages typically informing devices about network presence. These false signals create informational noise, overloading the communication channel and disrupting normal operation for clients and management systems.
AuthDOS	This method involves sending an overwhelming number of authentication requests to an access point. The target device becomes unable to process the influx of requests, leading to degraded performance and difficulties for legitimate clients to establish connections.
WIDSConfusion	This attack sends misleading management frames specifically targeting Wireless Intrusion Detection Systems (WIDS). These deceptive messages disorient the WIDS, potentially allowing the attacker to conduct further attacks unnoticed and contributing to network interference.
RTSCTSFlood	The network is saturated with Request-To-Send/Clear-To-Send (RTS/CTS) control frames. These frames are designed to manage channel access, but an excessive amount prevents normal data transmission and overloads the access point.
DeauthFloodMDK	The mdk4 utility executes the classic DeauthFlood. This variant may incorporate additional techniques, such as MAC address randomization, complicating source identification and enhancing attack effectiveness.
EAPOLStartFlood	This attack targets the authentication process by flooding the network with EAPOL-Start messages. An overwhelming number of these frames prevents the proper establishment of secure connections between clients and the access point.

The *Reward* R_t helps the agent reach its goal: to collect more information about the target network and connected devices, which will help find weaknesses to carry out DoS attacks, from disconnecting single devices to completely cutting off all clients for a long time. Consequently, the reward rt for each action at each time interval t is defined as follows:

$$R_t = \begin{cases} +0.01 \times eapol\_count \\ +0.01 \times handshake\_flag \\ +0.5 \times drop\_clients \\ +5.0 \times full\_DoS\_event \end{cases} \quad (1)$$

where drop_clients represents a reward for each unit decrease in station_count if the current value is below the baseline; full_DoS_event is a significant reward when effective drops to 0 and was greater than 0 in the previous episode, indicating a successful full DoS-effect.

IV. METHODOLOGY

A. Proposed PT Method

Fig. 2 illustrates the procedure of the proposed PT (penetration testing) method. First, the airmon-ng network utility is employed to initiate the mode of wireless adapter monitoring. This step transitions the wireless interface from managed mode to monitoring mode, which enables the capture of all IEEE 802.11 frames that are within range, including beacon frames, request responses, and authentication packets, instead of only traffic that is addressed to the host device.

After allowing the monitor to activate, a first passive inspection of the surrounding wireless networks is conducted using airodump-ng. Unlike active listening, this phase does not create additional traffic; instead, it is passive and listens to the broadcast management frames that are transmitted by access points (APs) and client devices. During this procedure, critical network parameters are gathered, including the BSSID, ESSID, channel number, signal strength (RSSI), encryption type (e.g., WPA2, WPA3), authentication method, and the number of associated client stations.

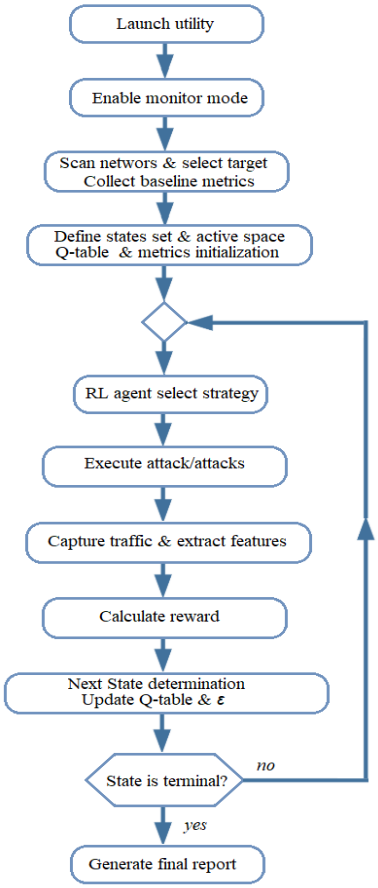

Fig. 2. Workflow of proposed framework.

All metrics that are captured are automatically written into a Comma-Separated Values (CSV) file for analysis later on. Storing the data in a structured format enables the comparison of available networks based on predefined criteria, such as the quality of their signals, security options, or client activity levels. Information derived from the CSV file is used to select the target network for vulnerability assessment. The selection may be facilitated by factors like the type of encryption, the network's load, the stability of the signal, or specific experimental goals.

Once the target BSSID and channel are known, the procedure moves on to the focused monitoring of traffic and subsequent security assessment stages, which may involve handshaking, authentication, or vulnerability evaluation in accordance with the selected testing scenario.

Before the RL agent starts working, the adapter is configured for the target channel and temporary CSV files created during the passive scanning of wireless networks are deleted.

The Q-learning algorithm initialization step includes building a state space S based on baseline metrics, an action space A based on fixed attacks, and filling the Q-table cells with zero values for all pairs (state, action). The core of the framework is an iterative attack cycle structured into multiple episodes, each comprising several steps.

The Q-learning algorithm's initial step includes a discrete space of states S and the definition of a space of actions. A, and setting all of the Q-table's state values to zero for all of the state-action pairs (s, a). The territory's space S is designed based on the metrics of the passive phase of network scanning: this is the basis for S. These metrics may include the strength of the signal (RSSI), the number of connected clients, the utilization of channels, the status of authentication, and the behavior of observed managers. Each state is associated with a numerical combination of these parameters that represents a quantized version of the MDP. The wireless environment is therefore modeled as a finite MDP.

The space of action A is defined by a pre-defined set of attacks that are fixed in nature (predefined). These may involve, for example, detaining the frame during the injection, attempting to capture the handshake, switching channels, monitoring traffic, or testing protocol-specific authenticity. By segmenting the strategies for attack, the agent can systematically assess and explore their effectiveness in different environmental circumstances.

The core of the framework is a repetitive cycle of attacks that is structured into multiple episodes, each of which contains multiple consecutive decision steps. An episode is the entire interaction with the target network that begins from a starting state and ends when a pre-designed stopping condition is reached (e.g., the goal was accomplished, the maximum number of steps was taken, or the environment was reset). Within each episode, the agent employs actions based on a strategy of exploration and exploitation, such as the ε-greedy policy, which enables the agent to balance between discovering new strategies and improving on previously learned effective actions. Through repeated interactions with the wireless world, the agent's behavior eventually approaches a policy that is optimal in terms of total reward; it identifies the most effective way to attack various networks.

At each step, the RL agent, employing an ϵ-greedy exploration strategy, selects a combination of one or two attack types from the predefined Action Space (e.g., "BeaconFlood + AuthDOS" or "DeauthFlood + EAPOLStartFlood"). The framework then launches the chosen attacks in parallel, typically running 2-3 instances concurrently for a short duration (e.g., 3-5 seconds).

To kill the attacking processes, *pkill -f* should be used for the mdk4 and aireplay-ng tools. Packet capturing with tcpdump: tcpdump should be run for 2 seconds to listen to packets on the network and write data into a PCAP file. Reading this PCAP file by a custom analyzer based on Scapy would return meaningful indicators of attack success, such as the number of EAPOL packets—the indicator of a successful WPA/WPA2 handshake.

The airodump-ng tool runs for 5 seconds to sniff the wireless network and fetch the count of left hooked clients (*station_count*). This value lets you guess the effect of the strike on linked gadgets. From the data collected, the reward is computed using Eq. (1). Q-learning updates Q-values for state and action, thereby helping to optimize a policy of selecting combinations of attacks. If the reward value returns greater than a certain level, then it returns a success label (*success_label* = 1). Otherwise, it returns a failure (*success_label* = 0). The set of DoS attacks can be termed as optimal by machine learning performance metrics (Accuracy, Precision, Recall, F1-score) derived from the total number of successes and failures.

A detailed final report on the adaptive pentesting execution is generated after all episodes of the algorithm are completed. The report is generated as a text log file that records all the details of the utility's operation. The report in HTML format contains the parameters of successful DoS attacks and performance metrics obtained during testing (number of stations in the network, outage time, number of successful DoS events, and machine learning algorithm performance metrics). For every vulnerability found, the report generates a few practical recommendations to improve the target network's security.

B. Experimental Setup

Our proposed approach has been implemented in Python 3.x as an adaptive penetration testing automation tool for wireless networks wherein dynamic attack parameter adaptation is based on feedback gained at the network interaction analysis stage, Fig. 3. The Python subprocess library executes external commands and processes (mdk4, aireplay-ng, tcpdump, and airodump-ng) enabling orchestration of attacks and data collection without direct usage of low-level Application Programming Interface (API). The scapy library will be used to develop a custom network packet analyzer. It performs parallel attacks that utilize the threading module, hence increasing attack efficiency due to parallel streams under very high intensity scenarios such as flood or deauthentication attacks. The tool operates under Linux and is set up on a Huawei MateBook X Pro Premium Edition notebook with an Intel Ultra 7 155H Processor, running for both WPA2 and WPA3.

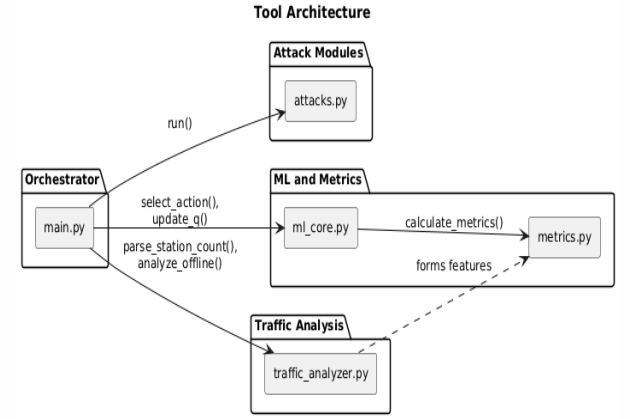

Fig. 3. Architecture of the tool.

The user chooses a wireless network interface (e.g., wlan0) to transition into monitoring mode, Fig. 4. After choosing, the script will automatically invoke the airmon-ng utility to reconfigure the interface. The tool terminates the processes of interference with the network if necessary and switches the adapter from the mode of management to the mode of observation.

Once the procedure is successfully carried out, the script will tell the user that the monitor mode has been toggled on. In the majority of Linux-based distributions (e.g., Kali Linux), the interface name is altered to represent the new operational state; for example, wlan0 may be renamed to wlan0mon. This mechanism of renaming ensures that the monitoring interface is distinguished from the other interfaces and prevents conflicts of configuration.

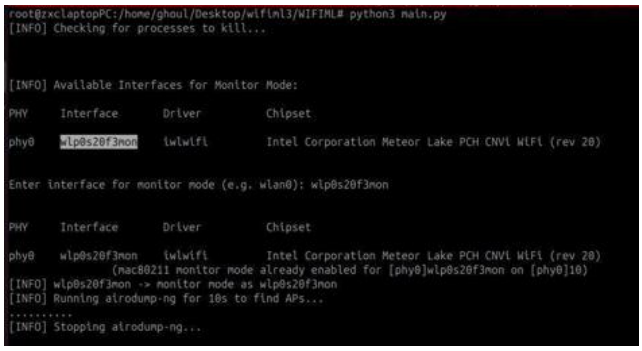

Fig. 4. Activating monitoring mode on a wireless interface.

At the next step, the automated detection of surrounding wireless networks is carried out. The script initiates the airodump-ng utility in passive mode of observation. The program listens to the wireless medium for a predetermined duration of 10 seconds; during this time, no active probing or injection of packets is undertaken. Instead, the tool takes no action but simply collects IEEE 802.11 management frames that are broadcast by nearby access points and associated devices.

During this time, airodump-ng automatically creates a structured CSV file that contains the detected network properties. The dataset that was collected includes the BSSID, the operating channel number, the ESSID, the RSSI, when present, the number of associated stations, the encryption method, and other pertinent information.

After the scanning phase is complete, the script decodes the generated CSV file and offers the user a list of discovered access points in a format that is intended to be user-friendly. The information displayed typically includes the BSSID, channel number, and ESSID for each detected network. Based on the information, the user chooses the target network by entering the appropriate index number. The selected network is then the subject of the following analysis and attacks. The procedure described in the figure is illustrated in Fig. 5.

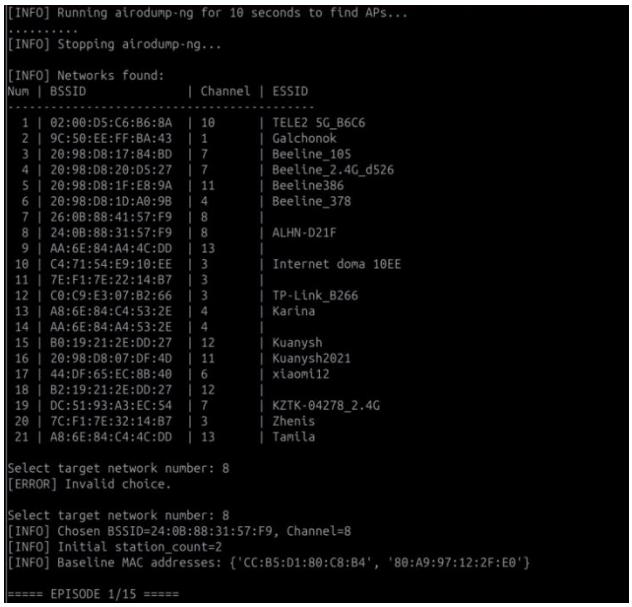

Fig. 5. Network scan results from the airodump-ng tool.

The next phase is for attack preparation and channel escape, Fig. 6. After choosing the target network (BSSID) and its associated testing channel, the program will begin with the preparation phase for active testing. First, the wireless interface that is in monitor mode is assigned to the selected channel using the command: iwconfig <monitor_if> channel <chan>

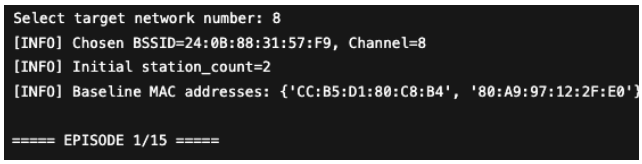

Fig. 6. Selecting a channel to conduct an attack.

This step guarantees that the monitoring interface remains attached to the target channel, preventing channel hopping and ensuring that the following operations will have a consistent outcome. Channel locking is of paramount importance in order to ensure a successful handshake interception and the management of frames with confidence.

To ensure the integrity and reproducibility of the experiments, previously generated temporary files, such as CSV records and files with the pcap extension that are captured during the testing process, are removed before initiating a new session. This procedure for cleaning eliminates remnants of earlier experiments and guarantees that the gathered data is specific to the intended target network.

The next phase is launching the attack cycle (episodes) and analysis. The program's process involves multiple episodes, each episode having multiple steps (STEPS_PER_EPISODE). At each step, the ML agent (using a (\varepsilon)-attentive strategy) chooses a combination of one or two attacks from the available Action Space, for example, “BeaconFlood + AuthDOS” or “DeauthFlood + EAPOLStartFlood”. The selected attacks are executed in succession (2–3 instances if necessary) and have a duration of several seconds (e.g., 3–5 seconds).

After stopping the attacks with the process of `pkill -f` for the mdk4 and aireplay-ng processes, the program acquires information. First, the application of tcpdump is briefly undertaken (2 seconds) to produce a pcap file that is captured. Second, a custom analyzer that extracts EAPOL packets and other features associated with handshakes. Third, the term airodump-ng is employed for 5 seconds in order to determine the number of remaining clients connected to the station (station_count).

The system then determines the reward according to predefined criteria (e.g., handshake was detected, the station count decreased, or the EAPOL count was > 0). A label that is successful has a value of 1 if the reward is greater than a specific threshold. The ML algorithm updates the Q values based on the success or failure of the attacks, and performance metrics (Accuracy, Precision, Recall, F1 score) are recorded based on the total number of successes and failures to determine the most effective combinations of attacks. The described procedure is represented in Fig. 7.

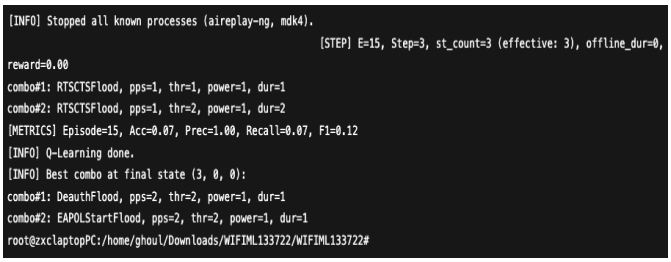

Fig. 7. Final choice of "best combination of attacks".

A TP-Link Archer C64 AC1200 Wi-Fi router with the ability to monitor modes, WPA2, and WPA was employed to create the intended network for the experiments. The router functioned as the entrance to the laboratory in controlled conditions.

To replicate the heterogeneous real-world wireless environment, three different devices were connected to the target router during the experimental analysis:

- MacBook M3 Pro computer;
- Samsung Galaxy S24+ (Exynos-based) smartphone;
- The Mijia Smart Evaporative Humidifier (model MJJSQ06DY) has a Wi-Fi component and is equipped with a humidification system.

The addition of devices with different hardware configurations, operating systems, and network layers enabled the evaluation of the proposed methodology under different patterns of traffic and client behavior.

The following experiment involved practical attacks that targeted the WPA3 protocol. The entire scheme for interaction between the devices and the practical experiment's device list is shown in Fig. 8.

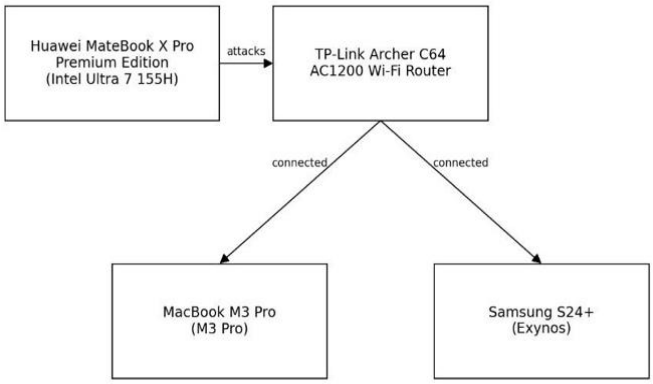

Fig. 8. Schema of equipment for the WPA3 network.

For this portion of the research, the router was additionally configured to operate in WPA3 mode, which enabled the use of authentication based on the Simultaneous Authentication of Equals (SAE) mechanism. A modern wireless network manager supporting the WPA3-SAE protocol was employed to create the intended network, as illustrated in Fig. 9. This configuration facilitated the evaluation of the proposed methodology under the enhanced security features incorporated in WPA3.

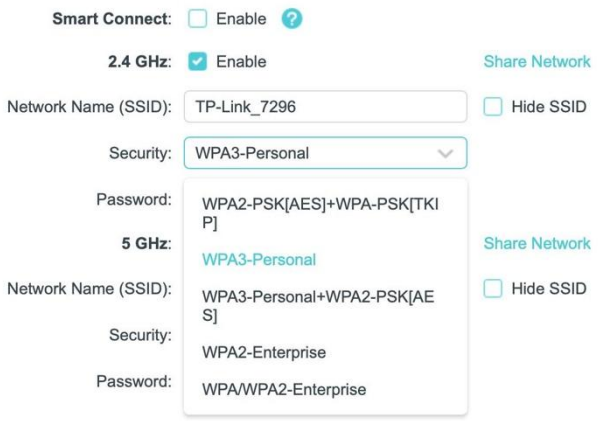

Fig. 9. Router settings with WPA3 support.

C. Performance Metrics

The metrics in Table II are used to judge how well the proposed approach is based on the Q-learning algorithm.

TABLE II. PERFORMANCE METRICS FOR THE RL AGENT

Metric	Description & Application
Accuracy (A): $A = \frac{TP + TN}{TP + FP + FN + TN}$	Accuracy quantifies the percentage of successful episodes (with a reward exceeding the threshold and a pred equal to 1) in comparison to all attacks carried out. Given that the true label is always 1 (as the objective is DoS/handshake), accuracy effectively reflects the rate at which the system achieves the desired DoS effect.
Precision (P): $P = \frac{TP}{TP + FP}$	Precision indicates the percentage of episodes in which the system accurately forecasted a successful outcome (reward exceeding the threshold), which in turn led to a genuine DoS effect. Due to the setup where the true label is always 1, precision will almost always be 1 if any successful episodes are registered, indicating that when the attack is deemed successful, the effect is indeed achieved.
Recall (R): $R \frac{TP}{TP + FN}$	It reflects the system's capability to detect successful attacks, specifically the proportion of episodes in which the reward exceeded the threshold. This is a key metric, as the primary goal is to ensure effective DoS or handshake capture. False negatives (FN) represent cases where the critical reward was not attained, despite the expected DoS effect being possible.
F1-score $F1 - score = \frac{2 * (P * R)}{P + R}$	The F1-score represents the harmonic mean of precision and recall. In scenarios where precision approaches 1, the F1-score effectively mirrors recall, serving as a balanced measure of overall model effectiveness.

V. RESULTS AND DISCUSSIONS

For WPA2 networks, the initial station_count was set to 3 clients, including the MacBook, Samsung smartphone, and Smart Humidifier. Fig. 10 shows an excerpt from a text log file that records all steps of the utility's operation.

```
===== EPISODE 7/15 =====
[ATTACK] BeaconFlood (instance 1): sudo mdk4 wlp0s20f3mon b -g -s 200
[ATTACK] BeaconFlood (instance 2): sudo mdk4 wlp0s20f3mon b -g -s 200
[ATTACK] RTSCTSFlood (instance 1): sudo mdk4 wlp0s20f3mon z
[ATTACK] RTSCTSFlood (instance 2): sudo mdk4 wlp0s20f3mon z
[INFO] Stopped all known processes (aireplay-ng, mdk4).
    [STEP] Episode=7, Step=1, station_count=2 (effective: 2), offline_duration=0, reward=0.00
  combo#1: BeaconFlood, pps=2, thr=2, power=1, dur=1
  combo#2: RTSCTSFlood, pps=1, thr=2, power=1, dur=2
[ATTACK] AuthDOS (instance 1): sudo mdk4 wlp0s20f3mon a -a 24:0B:88:31:57:F9 -s 50
[ATTACK] AuthDOS (instance 2): sudo mdk4 wlp0s20f3mon a -a 24:0B:88:31:57:F9 -s 50
[ATTACK] WIDSConfusion (instance 1): sudo mdk4 wlp0s20f3mon w -e 24:0B:88:31:57:F9 -c 8 -s 10
[INFO] Stopped all known processes (aireplay-ng, mdk4).
    [STEP] Episode=7, Step=2, station_count=1 (effective: 0), offline_duration=1, reward=5.00
  combo#1: AuthDOS, pps=1, thr=2, power=1, dur=2
  combo#2: WIDSConfusion, pps=2, thr=1, power=1, dur=2
[ATTACK] WIDSConfusion (instance 1): sudo mdk4 wlp0s20f3mon w -e 24:0B:88:31:57:F9 -c 8 -s 5
[ATTACK] WIDSConfusion (instance 2): sudo mdk4 wlp0s20f3mon w -e 24:0B:88:31:57:F9 -c 8 -s 5
[ATTACK] RTSCTSFlood (instance 1): sudo mdk4 wlp0s20f3mon z
[INFO] Stopped all known processes (aireplay-ng, mdk4).
    [STEP] Episode=7, Step=3, station_count=1 (effective: 0), offline_duration=2, reward=1.00
  combo#1: WIDSConfusion, pps=1, thr=2, power=1, dur=2
  combo#2: RTSCTSFlood, pps=2, thr=1, power=1, dur=2
[METRICS] Episode=7, Acc=0.10, Prec=1.00, Recall=0.10, F1=0.17
```

Fig. 10. The fragment of the WPA2 network adaptive pentest report.

Table III presents the experiment results for WPA2 network (only successful DoS event results are presented). The Q-learning agent found that using AuthDOS (pps=1, thr=1, power=1, dur=2) together with EAPOLStartFlood (pps=2, thr=2, power=1, dur=2) was the best choice, resulting in the highest machine learning results: Accuracy (A) = 0.12, Precision (P) = 1.00, Recall (R)= 0.12, and F1-score = 0.22. Other combinations that effectively reduced client count included DeauthFlood + WIDSConfusion and WIDSConfusion + AuthDOS.

TABLE III. Q-LEARNING PERFORMANCE FOR WPA2

Combo parallel attacks and attack parameters	reward	A	P	R	F1-score
DeauthFlood, pps=1, thr=2, power=1, dur=1 WIDSConfusion, pps=1, thr=2, power=1, dur=1	5.00	0.08	1.00	0.08	0.15
AuthDOS, pps=1, thr=1, power=1, dur=2 EAPOLStartFlood, pps=2, thr=2, power=1, dur=2	5.00	0.12	1.00	0.12	0.22
AuthDOS, pps=1, thr=2, power=1, dur=2 WIDSConfusion, pps=2, thr=1, power=1, dur=2	5.00	0.10	1.00	0.10	0.17
Best combo (3,0,0) AuthDOS + EAPOLStartFlood	Full DoS event				

In WPA3 network experiments, the initial station_count was two clients, as the Mijia Smart Humidifier (not supporting

WPA3) could not connect. The schema for the experiment is shown in Fig. 8.

The most effective combination found by the Q-learning agent for WPA3 was EAPOLStartFlood + WIDSConfusion (Table IV). Although WPA3 SAE enhances handshake resilience, the attacks effectively overloaded the access point, causing extended timeouts and high latency. DeauthFloodMDK + RTSCTSFlood also proved highly effective in inducing prolonged client disconnections. Other successful combinations included DeauthFlood + DeauthFloodMDK, DeauthFlood + EAPOLStartFlood.

TABLE IV. Q-LEARNING PERFORMANCE FOR WPA3

Combo parallel attacks and attack parameters	reward	A	P	R	F1-score
DeauthFlood, pps=1, thr=1, power=1, dur=2 DeauthFloodMDK, pps=1, thr=2, power=1, dur=2	5.00	0.05	1.00	0.05	0.09
EAPOLStartFlood, pps=2, thr=2, power=1, dur=2 WIDSConfusion, pps=2, thr=1, power=1, dur=2	5.00	0.07	1.00	0.07	0.14
DeauthFlood, pps=1, thr=1, power=1, dur=1 EAPOLStartFlood, pps=2, thr=1, power=1, dur=2	5.00	0.06	1.00	0.06	0.11
Best combo at final state (2, 0, 0): EAPOLStartFlood + WIDSConfusion	Full DoS event				

A notable difference emerged in attack effectiveness between WPA2 and WPA3 networks. Combinations that were highly successful in WPA2 demonstrated reduced effectiveness when applied to WPA3. This improvement is primarily attributed to WPA3's mandatory or optional Management Frame Protection (MFP/PMF), which blocks forged deauthentication and other management frames. Consequently, attacks with identical parameters yielded 0 reward in WPA3, where APs rejected attack markers, allowing clients to recover quickly. In contrast, WPA2 networks without MFP experienced significant delays or complete connection loss even under moderate loads, registering as a successful DoS attack.

The effective WPA3 attack combinations exploited specific weaknesses in WPA3 design by circumventing PMF. These attacks achieve their effect through:

- PMF bypass (they utilize unprotected management or control frames transmitted before key establishment);
- resource exhaustion (they overload client frame-processing queues, depleting CPU and memory).
- Sustained disruption occurs when attackers employ persistent frame floods instead of isolated bursts to maintain a continuous denial-of-service (DoS) attack.

The combinations specific to WPA3, especially those using control frames (RTS/CTS) and authentication frames, usually caused a longer-lasting DoS effect on WPA3 clients than just deauthentication attacks by themselves.

VI. CONCLUSION

This study introduces a reinforcement learning system that greatly improves testing the security of wireless networks by allowing an intelligent agent to find the best attack methods on its own. By treating the auditing process like a Markov Decision Process and using a Q-learning algorithm, our framework gets around the problems of fixed, preset attack plans. The practical tests on WPA2 and WPA3 networks show that the framework can find effective DoS strategies, even with the stronger security measures of WPA3, like Management Frame Protection. For WPA2 and WPA3 networks, non-coinciding combinations of successful DoS attacks were identified, which confirms the presence of vulnerabilities specific to network protocols. The agent's ability to learn and improve attack strategies in real time demonstrates the potential of intelligent approaches in designing security systems for modern wireless networks.

Further research is related to expanding the range of attacks, using Large Language Models (LLM) for post-processing of analysis results and generating recommendations for eliminating identified vulnerabilities in natural language, and integrating the proposed approach with intrusion detection systems.

We acknowledge several important limitations in the research study. Specifically, the current study lacks a comparison of the proposed Q-learning agent with baseline methods such as random attack selection, static scripts, or existing penetration testing tools. This omission limits the ability to fully assess the advantages and effectiveness of the reinforcement learning approach. Furthermore, the experimental setup was restricted to a single router and three client devices, which reduces the statistical significance of the results. Future work will focus on expanding the experimental scope by incorporating a greater diversity of devices, network configurations, and attack scenarios, as well as conducting comparative evaluations against existing solutions to provide a more comprehensive assessment of the proposed method's performance.

REFERENCES

[1] "IoT Analytics. State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally," September, 2024. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/

[2] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum and N. Ghani, "Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations," IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2702-2733, Apr. 2019.

[3] M. Vanhoef and F. Piessens, "Key reinstallation attacks: Forcing nonce reuse in WPA2," in Proc. 24th ACM SIGSAC CCS, Dallas, TX, USA, 2017, pp. 1313–1328.

[4] A. Halbouni, L. -Y. Ong and M. -C. Leow, "Wireless Security Protocols WPA3: A Systematic Literature Review," IEEE Access, vol. 11, pp. 112438-112450, Oct. 2023.

[5] M. Vanhoef and E. Ronen, "Dragonblood: Analyzing the Dragonfly Handshake of WPA3 and EAP-pwd," in 2020 IEEE Symposium on Security and Privac, San Francisco, CA, USA, 2020, pp. 517-533.

[6] S. Jain, S. Pruthi, V. Yadav and K. Sharma, "Penetration Testing of Wireless Encryption Protocols," in 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, March 2022, pp. 258-266.

[7] A. Chaudhary and K. Kumar, "Vulnerability Analysis of WPA Security Protocols," 2 in 024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kamand, India, June 2024, pp. 1-7.

[8] HJ Lu, Y. Yu, "Research on WiFi penetration testing with Kali Linux," Complexity, Vol. 2021, Art. No. 5570001, Feb. 2021.

[9] Q. Li, M. Zhang, Y. Shen, R. Wang, M. Hu, Y. Li, & H. Hao, "A hierarchical deep reinforcement learning model with expert prior knowledge for intelligent penetration testing," Computers & Security, Vol.132, 103358, 2023.

[10] Author, A. A., Author, B. B., & Author, C. C. (2026). Towards an AI-powered cyber resilience model: A systematic evaluation of frameworks against emerging threats. *International Journal of Advanced Computer Science and Applications*, 17(1), 54–66.

[11] J., Janisch, T., Pevný, & V. Lisý, "Nasimemu: Network attack simulator & emulator for training agents generalizing to novel scenarios," In European Symposium on Research in Computer Security, Springer Nature Switzerland, Septembe 2023, pp. 589-608.

[12] Schwartz, J., & Kurniawati, H. (2019). Autonomous penetration testing using reinforcement learning. *arXiv preprint arXiv:1905.05965*.

[13] T. Zhukabayeva, Z., Ahmad, A., Adamova, N., Karabayev, Y., Mardenov, & D. Satybaldina, "Penetration Testing and Machine Learning-Driven Cybersecurity Framework for IoT and Smart City Wireless Networks," IEEE Access, vol.13, pp. 86144-86166, May 2025.

[14] S. El Hamdani, S. Loudari, S. Novotny, P. Bouchner and N. Benamar, "A Markov Decision Process Model for a Reinforcement Learning-based Autonomous Pedestrian Crossing Protocol," 2021 3rd IEEE Middle East and North Africa Communications Conference (MENACOMM), Agadir, Morocco, pp. 147-151, Dec. 2021.

[15] B. Jang, M. Kim, G. Harerimana and J. W. Kim, "Q-Learning Algorithms: A Comprehensive Classification and Applications," IEEE Access, vol. 7, pp. 133653-133667, Sep. 2019.

[16] Tleuberdin S., Malakhov K., Issainova A., Boranbay Zh., Sergazin G. A Practical Evaluation of Wireless Network Security for Internet of Things: Vulnerability Assessment and Penetration

[17] Tleuberdin, S., Issainova, A., Adamova, A., Aidynov, T., Samashova, G., & Satybaldina, D. Analysis of Vulnerabilities and Practical Attacks on WPA3-Enterprise in Corporate Wireless Networks. – Procedia Computer Science. – 2025. – № 272. – Pp. 613-618