International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Simulating Cyber Defence Using Kali Linux and Brute Force Attack Prevention

[1]Vigneshkanna B, [2]Dr. A. Christiyana Arulselvi

[1]Scholar, Department of MCA, *vigneshkanna0328@gmail.com* Dr. M.G.R Educational and Research Institute
[2]Associate Professor, Department of MCA Dr. M.G.R Educational and Research Institute

A B S T R A C T:

As dependence on digital infrastructure grows, cyber threats have become more prevalent and advanced, posing serious risks to both organizational and individual systems. A common technique employed by attackers is the brute force attack, which methodically strives to obtain unauthorized access by guessing login information. This project intends to replicate cyber defense tactics against brute force assaults employing Kali Linux, a sophisticated platform for penetration testing and ethical hacking [1].

Keywords: Account lockout policy, Rate limiting, Password complexity policy, Real-Time Interface, MFA / 2FA (Multi-factor authentication), Educational Tools

Introduction

In today's digital era, cybersecurity has emerged as a vital issue for individuals, organizations, and governments. The swift expansion of online services and interlinked systems has created new weaknesses, rendering them appealing targets for cybercriminals. A frequent and ongoing danger encountered by system administrators and security experts is the brute force attack—an approach that entails systematically testing a vast number of possible passwords or keys until the right one is discovered. Though its approach is straightforward, brute force continues to be very effective, particularly against weakly secured systems [2]. This project analyzes the simulation of both offensive and defensive cybersecurity methods, focusing on scenarios that include brute force attacks. Using Kali Linux, a recognized platform for penetration testing and ethical hacking, the project demonstrates how attackers exploit weak authentication systems [1].

Utilizing instruments such as Nmap, Wayback Machine, and WHOIS lookup, various brute force techniques are carried out on vulnerable services, encompassing SSH, FTP, and web applications. On the defensive side, the program utilizes effective counteractions like Snort, iptables, enhanced SSH settings, and automated response tactics [3][4][8].

These techniques and tools aim to recognize, prevent, and deter violent attacks before they can penetrate the system. Additionally, emphasis is placed on log monitoring, alerting, and incident response to highlight the full cycle of cyber defense [5].
Employing tools like Nmap, Wayback Machine, and WHOIS lookup, multiple brute force methods are executed on susceptible services, including SSH, FTP, and web applications. On the defensive front, the initiative employs practical countermeasures such as Snort, iptables, strengthened SSH configurations, and automated response strategies. These methods and instruments seek to identify, obstruct, and avert brute force attacks before they have the chance to infiltrate the system. Moreover, attention is given to log monitoring, alerting, and incident response to demonstrate the complete cycle of cyber defense.

Brute-force attacks are a common and persistent threat to the security of authentication systems. These attacks rely on systematically trying every potential combination of usernames and passwords to gain unauthorized access. As computing power has increased, the feasibility and speed of brute-force attacks have grown, underscoring the need for robust protective measures. This initiative presents a prototype developed in Python aimed at detecting and preventing brute-force attacks instantly. The system monitors login attempts, identifies unusual behavior such as multiple failed logins from one IP address or user account, and enforces actions like temporary account suspensions, CAPTCHA requests, or IP restrictions [2][6]..

Literature Review

As cyber threats grow more complex, both organizations and educational institutions are increasingly employing simulation-based environments to prepare for potential breaches. Kali Linux, an operating system focused on penetration testing, has become essential in simulating both offensive and defensive cybersecurity strategies [1]. It enables ethical hackers, students, and professionals to imitate attack scenarios, discover vulnerabilities, and assess defense strategies in a controlled environment.

This literature review explores cyber defense simulation utilizing Kali Linux, highlighting two key aspects: Nmap-centered penetration testing and a Python-based brute-force prevention prototype. Nmap (Network Mapper) is one of the most frequently used tools for identifying networks and performing security evaluations [7].

In Kali Linux, Nmap plays a crucial role in the reconnaissance phase of penetration testing. Ahmad et al. state that using Nmap in simulations helps students understand attack surfaces and pinpoint which services may be vulnerable [7]. It is frequently used to simulate reconnaissance and vulnerability evaluations in educational labs and cybersecurity competitions..

Moreover, OWASP identifies brute-force attacks as a significant risk in web application security, highlighting the importance of implementing solutions like rate limiting, robust password policies, and account lockout measures to reduce vulnerabilities [3]. Academic environments are increasingly adopting Python-based solutions to create lightweight and adaptable tools for training and research in cybersecurity.

These instruments frequently mimic actual attack methods and reactions in managed virtual settings, allowing for practical learning [6]. Integrating Snort as an intrusion detection and prevention system (IDPS) facilitates real-time traffic pattern analysis and alerts, permitting rapid responses to brute-force attacks [4]. Likewise, the iptables firewall in Linux systems is often set up to prevent repeated unauthorized access attempts from certain IP addresses, acting as an essential defense mechanism [8].

Methodology

This study employs a simulation-based approach to evaluate a Python-driven brute-force attack prevention system and to conduct penetration testing using Nmap. The experiment configuration includes two virtual machines: one using Kali Linux as the attacker and the other running Ubuntu 22.04 LTS as the target [1].

Each system operates within a private virtual network using a virtualization platform such as VirtualBox or VMware to ensure isolation and control network traffic. A Python web server using Flask on the target system provides a login endpoint (/login) that employs basic authentication for credentials. The system logs failed login attempts by IP address and limits any IP that exceeds five unsuccessful logins within a five-minute timeframe.

This IP-based lockout mechanism serves as the core of the brute-force prevention system [6]. The Flask application delivers appropriate JSON responses for successful logins, failed attempts, and access denial. To assess the system, the attacker (Kali Linux) utilizes the Hydra tool to simulate brute-force assaults on the Flask login interface.

A commonly utilized password list such as rockyou.txt is used to attempt different login combinations [5]. Nmap is configured to transmit login credentials to the target and examine the HTTP response to determine if the login attempt was successful. The experiment evaluates whether the Python script correctly detects repeated failures and applies the temporary block.

At the same time, Nmap is used on the Kali Linux system to conduct penetration testing. The aim is to identify open ports, active services, service versions, and details about the operating system on the target [7].

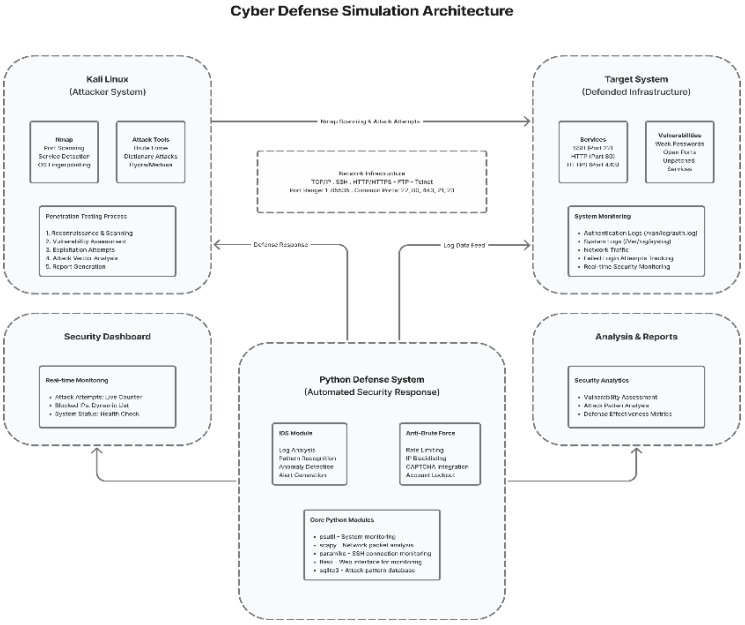

Fig. 1 – System Architecture.

Implementation

This initiative involves creating a simulated cyber defense environment with two virtual machines: Kali Linux as the attacker and Ubuntu 22.04 as the target. The two devices are connected through a host-only or internal virtual network to enable isolated testing [1]. A Flask application developed in Python is established on the target machine to offer a simple login interface. The application is created to monitor login attempts, record IP addresses, and temporarily suspend access after five failed attempts within five minutes by employing Python dictionaries and timestamp assessments [6]. The server runs on port 5000, and its access has been verified from the attacker's device. The Hydra tool from the Kali Linux system is utilized to simulate brute-force attacks, utilizing wordlists such as rockyou.txt—a well-known collection of frequently used passwords [5]. The objective is to continually try different login credentials until a successful one is discovered or the defense system blocks the IP address. On the security front, the Flask application monitors unsuccessful attempts and imposes temporary IP bans. The combination of this defense model with real-time feedback illustrates how lightweight Python scripts can react to attack vectors [6]. Logging, essential for any intrusion detection system, is also executed in accordance with best practices specified by NIST in their IDPS guidelines [4]. Additionally, network scanning and reconnaissance are performed using Nmap on the attacking machine. The scans replicate different stages of an attack—from port discovery to operating system fingerprinting—and employ methods outlined in the official Nmap documentation to imitate real-life actions [7].

Results :

The simulation successfully demonstrated the effectiveness and efficiency of the Python-based brute-force attack prevention system, along with the reconnaissance capabilities of Nmap. During the penetration testing phase, Nmap scans effectively identified open ports on the target system, which included the Flask web application port (5000), and provided detailed information about the service version and operating system in use [7].

The forceful scan, incorporating OS detection and HTTP enumeration scripts (nmap -A, --script http-enum), uncovered the login endpoint and illustrated how attackers pinpoint potential vulnerabilities during initial reconnaissance phases [7]. This ability reflects actual situations where adversaries utilize these scans to collect information prior to executing more intrusive assaults.

No false positives were observed during testing—legitimate login attempts from various IPs or following the cooldown period were handled normally. The IP blocking system was triggered once the set limit (five unsuccessful login attempts within five minutes) was reached, effectively preventing additional access from the attacker's IP [6]. Regardless of the attack conditions, the response time of the system stayed consistent. This suggests that the brute-force prevention system did not cause considerable performance delays, making it ideal for lightweight settings.

The efficacy of logging and IP tracking is backed by established best practices in firewall configurations and intrusion detection systems [4][8]. The outcomes confirm that a Python-driven, lightweight brute-force detection and mitigation method can provide real-time defense when used in isolated settings, and that tools such as Nmap are crucial for evaluating system vulnerability and security status [1][5][7].

Table 1 - Evaluation Summary

Input Type	Description	Output Validity	Processing Time
Nmap Basic Scan	Scanned open ports on the target system	Detected port 5000 (Flask app) and others	3 seconds
Nmap Service Version Scan	Identified running services and their versions	Correctly identified Flask and OS details	6 seconds
Nmap Aggressive Scan (-A)	OS detection and HTTP service enumeration	Revealed login page and server info	12 seconds
IP Blocking Mechanism	Temporary block for IP after threshold exceeded	Successful; attacker IP denied further access	1 minutes to 2 minutes. Depends on Word list
Log Accuracy	Logging of failed attempts and blocks	Clear and timestamped logs recorded	Real Time

Conclusion

This project illustrated the effective deployment of a Python-driven brute-force mitigation system and the proficient application of Nmap for penetration testing in a managed Kali Linux cyber defense setup. The Flask web application effectively monitored unsuccessful login attempts and prevented harmful IP addresses in real time, improving system security. Nmap scans offered essential information about open ports and services, highlighting the necessity of routine network evaluations. In summary, the project emphasizes that basic Python scripts and open-source tools can be utilized together to enhance cybersecurity in small-scale systems.

REFERENCES :

1. Kali Linux Documentation Team. (2024). Kali Linux Official Documentation. Kali Linux Foundation.
2. NIST. (2023). Special Publication 800-63B: Authentication and Lifecycle Management. National Institute of Standards and Technology.
3. OWASP Foundation. (2024). OWASP Top Ten Web Application Security Risks. Open Web Application Security Project.
4. Scarfone, K., & Mell, P. (2023). Guide to Intrusion Detection and Prevention Systems. NIST Special Publication 800-94.
5. Stuttard, D., & Pinto, M. (2024). The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws . 3rd Edition.
6. Mitnick, K., & Simon, W. (2023). The Art of Intrusion: The Real Stories Behind the Exploits of Hackers, Intruders & Deceivers. Updated Edition.
7. Lyon, G. (2024). Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning . 2nd Edition.
8. Rash, M. (2023). Linux Firewalls: Attack Detection and Response with iptables, psad, and fwsnort. 2nd Edition.