Rev. Roum. Sci. Techn.– Électrotechn. et Énerg.
Vol. **71**, *1*, pp. 145–150, Bucarest, 2026

SIMULATING CYBERATTACKS IN A VIRTUAL ENVIRONMENT: A CASE STUDY IN THREAT DETECTION

FLAVIA MARIA BARBU [1], CONSTANTIN VIOREL MARIAN [2], AMADOU SADIO DIALLO [3]

Keywords: Cybersecurity; Network security; Penetration testing; Virtual simulation; Ethical hacking.

In this study, we present how we simulated cyberattacks in a controlled virtual environment to assess the security and detection abilities of a Windows 7 system. Using common penetration testing tools, we recreated realistic scenarios including port scanning, brute-force login attempts, password cracking from NTLM hashes, and phishing with cloned websites. Our findings revealed that outdated systems, when left unpatched and poorly monitored, are especially vulnerable, thereby underlining the need for regular updates, strong password practices, and real-time monitoring to reduce risk, while providing insights into how such environments can be tested and strengthened against different cyberthreats.

1. INTRODUCTION

As cyberattacks continue to grow in complexity [1], analyzing malignant behaviors has become a necessity [2-4]. Institutions face increasing pressure to develop and test effective countermeasures [1,5,6], yet their experimentation is fraught with legal and technical risks [7-10]. A promising solution is self-contained network environments [11,12] that allow for the safe replication and study of common attacks and vulnerabilities without exposing external information.

We present a setup designed to launch cyberthreats. We configured our own private network, composed of a purposely vulnerable Windows 7 target system, along with a Kali Linux attacker, a REMnux analysis machine, and a pfSense router (read §III), to execute a wide range of attacks, while monitoring traffic and system behavior. This allowed us to build a bridge between what an attacker does on the wire and how the target reacts deep inside its processes.

Our contribution goes beyond spotlighting how effortless data theft is from unprotected hosts; we identify specific telemetry patterns that reveal where the so-called “invisible" threats are. To do so, we detect subtle warning signs before a breach occurs. Unlike prior work that focuses only on damage or recovery (read §II), we propose a setup (Fig. 1) that links host-level artifacts with network behavior. Our method isolates the traffic to capture a clean stream of malicious data, exploits system vulnerabilities, records network stuttering, extracts memory artifacts, and stores our findings in a central repository for further comparison. The main aim of our dual-layer approach is to showcase that attacks may be concealed on the host but not in their behavior.

The subsequent sections cover the following: the current state of the art (Section 2); the details of the roles and specifications of each virtual machine (Section 3); a comprehensive description of our scenarios and their execution (Section 4); an analysis of their activity (Section 5) to identify patterns and traces; and finally, a discussion of our outcome and potential directions (Section 6).

2. BACKGROUND

Over time, researchers have pursued various approaches to replicate cyberattacks in ethically controlled conditions. Yet the *visibility gap* persists. On this point, our solution addresses it by implementing what we call the Evidence Feedback Loop: it collects evidence from both host and network activity, uses feedback to compare and correlate these signals, and forms a continuous loop where each new piece of information improves detection, revealing threats that would otherwise remain hidden.

In October 2024, Wan et al. [13] studied how communication in Multi-Agent Systems (MAS) can be disrupted by combined cyberattacks. They analyzed scenarios involving False Data Injection (FDI), Denial-of-Service (DoS), and their simultaneous use, showing how intruders corrupt data, disrupt communication, or both. To address the issue, they proposed a hybrid approach using neuroadaptive control and event-triggered communication to detect malicious patterns, reduce unnecessary traffic, and improve system resilience.

Two months later, the same attacks and others were employed by Nguyen et al. [14] with a completely different purpose, namely, to showcase how power systems may be destroyed. They simulated phishing to send emails with corrupted attachments to install malware and create backdoors into networks; credential theft to exploit vulnerabilities in Active Directory servers to acquire account information; and scanning to carefully explore and map the internal network. For deeper disruptions, they did not apply DoS solely to overwhelm communication channels but specifically targeted **R**emote **T**erminal **U**nit (RTU) nodes to stop them from responding to control center commands. In addition, they executed unauthorized remote access and **M**an-**i**n-**t**he-**M**iddle (MiTM) to secretly intercept messages between multiple components of the system and eventually inject misinformation.

While both studies [13, 14] analyze attack damage and defense mechanisms, they leave a key problem unresolved: attacks can remain hidden when defenders observe only the host or only the network. Unlike theirs, our work adds an important dimension by demonstrating how crucial data analysis is for detecting subtle predictive signs of attacks (see §V). To broadly prevent destructiveness, we suggest that this evidence be immediately considered, before any serious damage is caused, and eventually apply injection testing so that anomalies are promptly noticed [15].

Though their context is different, we consider the studies of Liu et al. [16] and Dumitrache et al. [17] to be worth mentioning as the authors propose a model to remove [16], respectively reduce [17] the number of points of failure, thereby restraining attackers from threatening the entire system. They used blockchain to ensure that any harmful

[1] Doctoral School of Electronics, Telecommunications & Information Technology, National University of Science and Technology Politehnica Bucharest, Romania.
[2] National University of Science and Technology POLITEHNICA Bucharest, Romania.
[3] Faculty of Engineering in Foreign Languages, National University of Science and Technology Politehnica Bucharest, Romania.
E-mails: flavia_maria.barbu@upb.ro, constantin.marian@upb.ro (correspondence), amadou_sadio.diallo@stud.fils.upb.ro

DOI: 10.59277/RRST-EE.2026.1.24

attempt is easily detectable and essentially impossible to execute without consensus. Their approach may be a relevant inspiration to mitigate brute-force attacks, as the one we launched (see Section 4), as the cryptography they propose is stronger than standard credentials, and MiTM attempts, as exemplified in [13], by leveraging the blockchain's immutable ledger. However, it may not be efficient enough to protect against an attacker who decides to target the unpatched flaws (see Section 4).

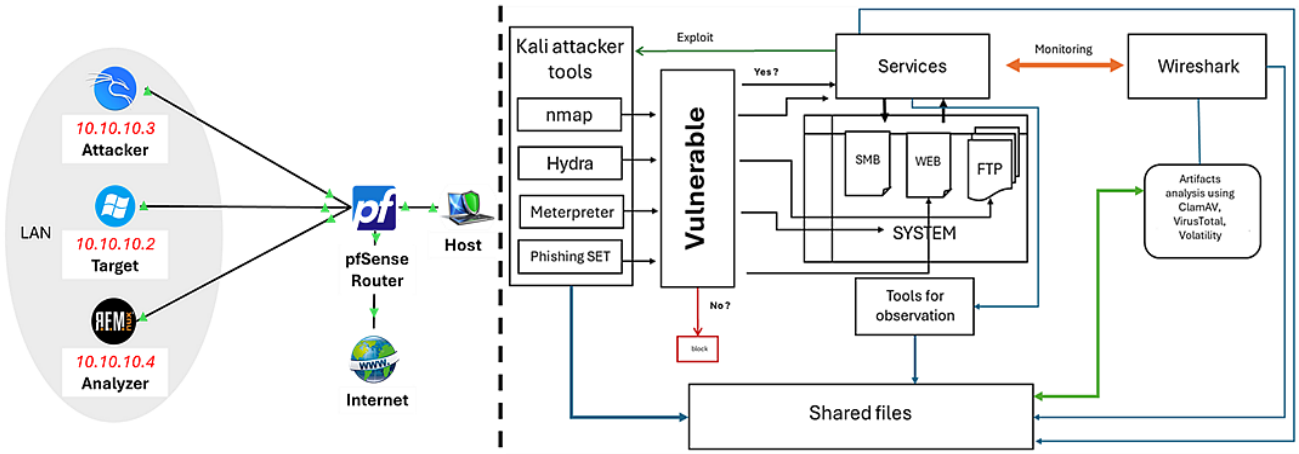

Fig. 1 – On the left side, we illustrate our experimental setup and analysis workflow. On the left, we present a controlled network environment where a router connects a target system to the rest of the network and allows us to safely observe interactions. This setup includes an attacking machine, a target machine, and an analysis system used to monitor and record activity. On the right, the workflow features how attacks are launched against the target, how their success or failure is observed, and how all related activity is continuously monitored. Network traffic is captured for analysis, while key system artifacts are collected to understand malicious behavior. All collected evidence is stored in a shared repository, creating a feedback loop that supports ongoing detection and analysis.

3. PROPOSED SETUP

We developed an experimental virtual testbed using VMware Workstation Pro to manage an isolated, multi-node network. We structured the environment as a WAN-LAN setup (Fig. 1), where a pfSense router serves as the critical intermediary between the host machine and the private subnet (10.10.10.0/24). Our configuration ensures total isolation for safe exploit execution:

3.1. KEY ENTITIES

Our topology consists of multiple nodes, each playing a distinct role, as follows:

Kali Linux (10.10.10.3): This machine served as our primary attacker workstation. Its main benefit is that it comes pre-loaded with an extensive collection of penetration tools [18], thereby becoming for us a valuable resource to simulate various attack scenarios. From network scanners that had assisted us in discovering hosts and services, to powerful vulnerability exploitation frameworks, we employed these penetration tools to thoroughly test how well systems stand up against known weaknesses (Fig. 1).

Windows 7 (10.10.10.2): We intentionally created this virtual machine as our target, as its Operating System (OS), unfortunately, has more security holes compared to newer alternatives [19, 20], resulting in increased vulnerability of both its core and the common applications it typically runs to threats. We intentionally set it up to be exposed to well-known vulnerabilities, so we could thoroughly observe and analyze the impact of our attacks. Fig. 1 shows the deployment of deliberately misconfigured web servers, File Transfer Protocols (FTPs) with weak authentication, and poorly secured Server Message Block (SMB) shares to investigate how they directly impacted the OS of the target. Readers more interested in how we performed all these attacks can skip directly to §IV.

REMnux (10.10.10.4): Dedicated to forensic analysis and reverse engineering, it examines malicious software in detail [21], therefore being essential for a thorough understanding by supporting both static (code-level) and dynamic (behavioral) analysis [22]. With it, we easily identified the malware patterns, thereby revealing how the malware operates and the specific traces it leaves on the network.

pfSense: It effectively protected our internal network from unauthorized external connections [23], including those from the host machine and the internet, preventing unintended data leaks or interactions. Through it, we configured firewall rules to enforce security policies, control traffic flow, and monitor network activity. This feature was vital to check how different settings influence attack spread and to evaluate the strength of our defenses.

3.2. WORKFLOW ANALYSIS

Our framework follows a workflow that links offensive actions to defensive monitoring:

Monitoring and Scrutiny: During execution, we employed Wireshark to capture raw network traffic for packet-level analysis (Fig. 2, Fig. 3).

Artifact Collection: Beyond network traffic, we extracted critical host-level artifacts. We used Volatility for memory forensics, alongside ClamAV and VirusTotal, to identify and classify malicious payloads.

Shared Folder: This served as a repository to store all our experiment notes and results. It was crucial for keeping our research data safe and easy to access, as we used it to organize all attack logs, network traffic captures, any malware samples, specific configurations, and detailed notes from each experiment. Being extremely important, it ensured that all valuable information from the subsequent analyses was readily available for this paper.

We fortunately designed (Fig. 1) a practical environment to simulate different types of attacks, from network scanning and brute-force attempts to complex exploits and malware deployment (read more §IV). Although our workflow is already highly effective (see §V), potential improvements could involve incorporating more recent OS as the target or

cloud virtual machines to test against a wider variety of threats.

4. ATTACK SCENARIOS AND EXECUTION

We executed a series of attacks, which hackers commonly perform to acquire sensitive information [24–26]. We believe this approach (Fig. 5) was efficient for assessing both the resilience and the threat recognition of our setup (Fig. 1). Below, we detail them as follows:

4.1. PORT SCAN

From our attacker machine, we scanned the target and found 11 open TCP ports and 989 closed ones. Using a SYN scan (-sS) and service version detection (-sV), we successfully identified key services like FTP on port 21, IIS 7.5 on port 80, SMB on port 445, and several RPC-related services. The results clearly pointed to the system running Windows 7, and further analysis of the MAC address confirmed it was a VMware Workstation Pro virtual machine, exactly matching our expectations.

We also manually connected to the FTP service, confirming it was active and ready for deeper inspection. Overall, the scan provided accurate and valuable insights to map out potential paths while remaining hidden.

4.2. BRUTE FORCE

We applied Hydra to perform a brute-force attack on the FTP server at the IP address, testing combinations of usernames and passwords from our predefined lists. To speed up the attack, we ran 4 threads simultaneously, which allowed multiple login attempts to be executed in parallel. To further proceed, we used the valid credentials to successfully access the FTP server and upload a customized shell. This step was crucial as it demonstrated that we could have infiltrated malicious files at any time to permanently compromise the target (Fig. 5).

4.3. EXPLOITING SMB FLAWS

We specifically targeted the EternalBlue vulnerability (MS17-010), which, according to Liu [27], can affect the unpatched versions of Windows. After launching Metasploit, we loaded the corresponding exploit module, configured the target and attacker IP addresses: remote and local host.

While the first attempt resulted in an authentication error, the second one successfully confirmed that the OS was completely exposed. Metasploit then delivered a series of crafted packets that corrupted the SMB service's memory, enabling us to inject a Meterpreter payload. This led to a successful compromise, granting full remote access through a 64-bit session.

Then we ran the commands sysinfo and getuid (Fig. 5) to confirm that our session (NT AUTHORITY\SYSTEM) had root access, the highest level of access available on a Windows machine [28]. We contend that we have accurately pointed out the significant risk posed by unpatched SMB vulnerabilities, which can be exploited to gain full control over an entire OS, therefore emphasizing the necessity of robust threat detection and enabling timely security updates.

4.4. PASSWORD CRACKING

After gaining access to the target via Meterpreter, we extracted the password hashes using the hashdump command and saved them locally in the Shared folder for further analysis. By applying John the Ripper with the NTLM format, we quickly cracked one user's password, instantly executing an unauthorized takeover. This successful outcome (Fig. 5) proves the vulnerability of weak and/or default credentials.

4.5. PHISHING

We simulated an automated phishing attack (Fig. 5) using the Social-Engineer Toolkit (SET) to demonstrate how attackers can deceive users into revealing sensitive data (e.g. login credentials) by replicating a legitimate website. We conducted this experiment within a secure, isolated LAN without Internet access. Since SET's automatic site cloning feature requires an Internet connection, we previously downloaded the Facebook login page and transferred it to our attacker machine.

We then configured SET to host our replica on our local web server. For this specific step, we applied the Credential Harvester Attack: we loaded the clone, specified our local IP address (as this is where the clone is hosted), and attached the original URL to enhance credibility.

After completing the setup, we sent the phishing link to the target machine via SMB. As the victim entered their credentials on our replica, SET captured the data in real time and displayed it in our terminal. Our goal was to demonstrate how relatively simple it is for an attacker to harvest personal data by employing the most elementary tactics, even within an isolated environment. Consequently, besides relying on protective software, we recommend educating users to recognize and avoid these deceptive tactics.

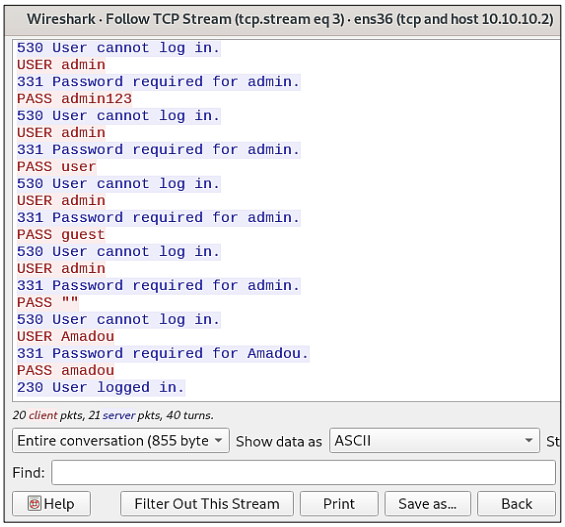

Fig. 2 – A Wireshark screenshot of the traffic captured during our FTP.

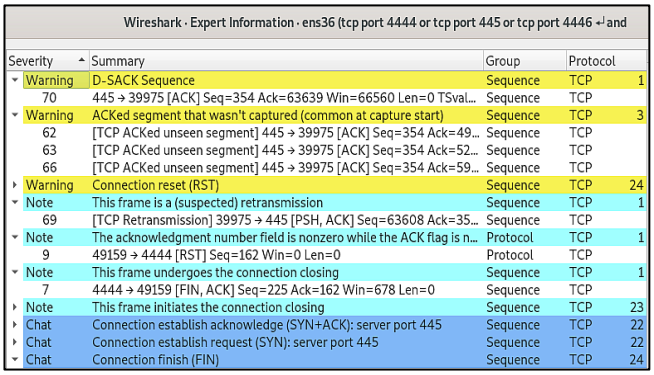

Fig. 3 – An analysis, specifically filtered for traffic involving host 10.10.10.2 and ports 4444, 4446, and 445 (which we associated with Meterpreter and SMB).

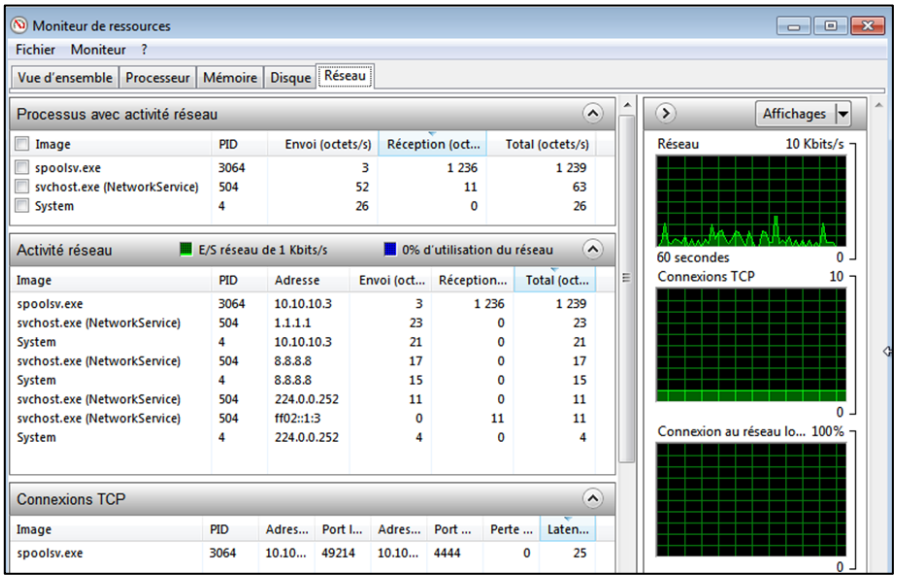

Fig. 4 – The Network tab presents the activity of the standard printer service (PID 3064), exhibiting atypical network activity. Notably, the TCP Connections pane confirms that it has an established connection from our compromised target machine (10.10.10.2) to the attacker (10.10.10.3) on port 4444, which is well-known for reverse shells. Please, notice that the content of this image is displayed in French.

5. BRIEF ANALYSIS OF THE ACTIVITY

Our analysis focused on correlating how each attack manifested across the network and the host. By synthesizing data from Wireshark and Windows Resource Monitor, we identified the following key patterns and **I**ndicators **o**f **C**ompromise (IOCs):

5.1. SEQUENCING

Remark: A repetitive sequence of "530 User cannot log in" messages originating from a single IP (Fig. 2).

Synthesis: This allows defenders to pinpoint the exact second of a breach (marked by the final "230 User logged in" status).

5.2. PREDICTIVE INSTABILITY

Remark: high frequency of TCP retransmissions, D-SACK sequences, and connection resets on Port 445 (Fig. 3).

Synthesis: these artifacts act as early warnings. They signal an exploitation attempt in progress before the final Meterpreter payload is even executed.

5.3. DISCREPANCIES

Remark: The "Printer Spooler" service (PID 3064) appeared legitimate on the host but maintained an active connection to an outside attacker on Port 4444 (Fig. 4).

Synthesis: This clearly highlights the presence of a visibility gap. Malicious behavior was unfeasible to detect via host logs alone, but became obvious when cross-referenced with network telemetry.

In addition to these, we conducted a host-level analysis using the Windows Resource Monitor (Fig. 4). We disguised our exploits to appear as normal system processes, allowing them to blend seamlessly with legitimate activity and evade detection unless network traffic was closely examined. The process itself appeared entirely benign and retained legitimate metadata. However, the deception became evident when PID 3064 maintained a persistent outbound connection to the attacker (10.10.10.3) on port 4444.

Based on our monitoring, we strongly assert that a best practice for effective attack detection is to simultaneously analyze both network traffic and host behavior to detect anomalies, which may not be apparent at first glance, since they might be missed when looking at only one of them.

6. CONCLUSIONS

This experimental project partially extends on earlier work [29, 30] to reveal how easily attackers can compromise systems that are not properly secured. It helps highlight the weaknesses of current security measures and points to areas where further improvements are needed.

First, we observed that even basic techniques, including brute-force login attempts and phishing via cloned websites, remain unsurprisingly still effective when systems are poorly configured or outdated.

Second, our ability to exploit known weaknesses (e.g., EternalBlue) and to extract hashes for offline cracking showcased how quickly an initial breach can escalate into full control over a target. This highlights the importance of designing secure architectures, where careful design of communication layers can prevent unauthorized access [31].

We trust that our findings effectively stress the importance of advanced threat detection and more rigorous monitoring, especially at the network level, as our analysis (Fig. 3) revealed that meaningful IOCs often emerge from subtle patterns of malicious activities across multiple trials.

As previously mentioned in §III, potential future work may prioritize testing against more recent and diversified environments. We consider this to be essential as this research relies heavily on legacy systems, which no longer receive security updates [32, 33], and thus may partially address the challenges of the current infrastructures.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Flavia Maria Barbu: manuscript structuring, investigation, visualization, writing-reviewing, and editing.
Constantin Viorel Marian (corresponding author): conceptualization, methodology, supervision, writing-the submitted version.
Amadou Sadio Diallo: software, validation, writing original draft.

Received on 25 June 2025

APPENDICES

Fig. 5 – The results of all our five attacks in the same order we present them in §IV. The process starts with an Nmap scan, identifying open ports and confirming the target as a Windows virtual machine. This is followed by a successful brute-force attack, which compromised the FTP service and gained valid access to credentials. Upon this, a Meterpreter session is established, giving full control over the OS. Further post-exploitation activities involved John the Ripper successfully cracking extracted password hashes and phishing to visualize private credentials. The content must be parsed top-down.

REFERENCES

1. I.C. Bogdan, E. Simion, *Cybersecurity assessment and certification of critical infrastructures*, U.P.B. Scientific Bulletin, Series C, **86**, *4* (2024).
2. Z. Zhang, H.A. Hamadi, E. Damiani, C.Y. Yeun, F. Taher, *Explainable artificial intelligence applications in cyber security: state-of-the-art in research*, IEEE Access, **10**, pp. 93104–93139 (2022).
3. B.K. Mamade, D.M. Dabala, *Exploring the correlation between cyber security awareness, protection measures, and the state of victimhood: the case study of Ambo University's academic staff*, Journal of Cyber Security and Mobility, **10**, *4*, pp. 699–724 (2021).
4. I. Ahmad, F. Rodriguez, T. Kumar, J. Suomalainen, S.K. Jagatheesaperumal, S. Walter, *Communications security in industry X: a survey*, IEEE Open Journal of the Communications Society, **5**, pp. 982–1025 (2024).
5. A. Arteche, C. Asher, C. Bull, H. Dare, I. Datey, E. Elshoff, M. Mahmoud, *Data approach to biometrics in cybersecurity with related risks*, 2022 International Conference on Computational Science and Computational Intelligence, Las Vegas, NV, pp. 1059–1066 (2022).
6. S. Jain, P. Ashok, S. Prabhu, *Emerging technologies for cybersecurity in healthcare: evaluating risks and implementing standards*, 2024 International Conference on Cybernation and Computation, Dehradun, India, pp. 725–731 (2024).
7. M. Xiao, A. Sellars, S. Scheffler, *When anti-fraud laws become a barrier to computer science research*, arXiv preprint arXiv:2502.02767 (2025).
8. A.-I. Concea-Prisăcaru, T. Nițescu, V. Sgârciu, *SDLC and the importance of software security*, U.P.B. Scientific Bulletin, Series C, **85**, *1* (2023).
9. A.J. Burstein, *Conducting cybersecurity research legally and ethically*, First USENIX Workshop on Large-Scale Exploits and Emergent Threats, San Francisco, CA (2008).
10. K. Macnish, J. van der Ham, *Ethics in cybersecurity research and practice*, Technology in Society, **63**, 101382 (2020).
11. H. Jiang, T. Choi, R.K.L. Ko, *Pandora: a cyber range environment for the safe testing and deployment of autonomous cyber attack tools*, arXiv preprint arXiv:2009.11484 (2020).
12. P. Cao, Z. Kalbarczyk, R.K. Iyer, *Security testbed for preempting attacks against supercomputing infrastructure*, arXiv preprint arXiv:2409.09602 (2024).
13. Y. Wan, X. Shi, X. Zhao, J. Cao, *Distributed secure consensus tracking of multiagent systems under hybrid cyberattacks: an event-triggered neuroadaptive approach*, IEEE Systems, Man, and Cybernetics Magazine, **10**, *4*, pp. 77–91 (2024).
14. T.-T. Nguyen, R. Kadavil, H. Hooshyar, *A real-time cyber-physical simulation testbed for cybersecurity assessment of large-scale power systems*, IEEE Transactions on Industry Applications, **60**, *6*, pp. 8329–8340 (2024).
15. S.T. Velayudhan, K. Devi, *BUFIT: fine-grained dynamic burst fault injection tool for embedded field programmable gate array testing*, Rev. Roum. Sci. Techn. – Électrotechn. et Énerg., **69**, *3*, pp. 299–304 (2024).
16. Z. Liu, L. Meng, Q. Zhao, F. Li, M. Song, Y. Jian, H. Tian, *Authenticated key agreement scheme based on blockchain for AMI communication security*, Rev. Roum. Sci. Techn. – Électrotechn. et Énerg., **68**, *2*, pp. 218–223 (2023).
17. C.-G. Dumitrache, C.V. Marian, G. Predusca, F.M. Barbu, M. Neferu, *Wireless authentication system for Internet of Things using FreeRADIUS and blockchain*, Rev. Roum. Sci. Techn. – Électrotechn. et Énerg., **70**, *4*, pp. 585–590 (2025).
18. I. Nedyalkov, *Study the level of network security and penetration tests on power electronic devices*, Computers, **13**, *3*, 81 (2024).
19. B. Nijssen, L. Langer, *Comparing security vulnerabilities in Windows 7 and Windows 10* (2020).
20. P. Kaluarachchi, C. Attanayake, S. Rajasooriya, C. Tsokos, *An analytical approach to assess and compare the vulnerability risk of operating systems*, International Journal of Computer Network and Information Security, **12**, pp. 1–10 (2020).
21. H. Ai, *REMnux: a Linux distro for malware analysis and reverse engineering*, Undercode Testing (2025), https://undercodetesting.com/remnux-a-linux-distro-for-malware-analysis-and-reverse-engineering/ (Accessed: Jun. 18, 2025).
22. P. Paganini, *REMnux: malware analysis*, Security Affairs (2020), https://securityaffairs.com/106380/malware/remnux-malware-analysis.html (Accessed: Jun. 18, 2025).
23. *********Ingress and egress firewall rules*, Netgate Documentation, https://docs.netgate.com/pfsense/en/latest/firewall/ingress-egress.html (Accessed: Jun. 18, 2025).
24. J.M. Pittman, *Machine learning and port scans: a systematic review*, arXiv preprint arXiv:2301.13581 (2023).
25. L. Livera, *Top 50 common types of cybersecurity attacks: a comprehensive guide*, LinkedIn (2025), https://www.linkedin.com/pulse/top-50-common-types-cybersecurity-attacks-guide-lahiru-livera-ndcwc (Accessed: Jun. 18, 2025).
26. C. Harry, I. Sivan-Sevilla, M. McDermott, *Measuring the size and severity of the integrated cyber attack surface across US county governments*, Journal of Cybersecurity, **11**, *1*, tyae032 (2025).
27. Z. Liu, *Working mechanism of EternalBlue and its application in ransomworm*, arXiv preprint arXiv:2112.14773 (2021).
28. *********Privilege escalation – Windows introduction*, InfoSec39 (2025), https://infosec39.home.blog/2025/01/17/privilege-escalation-windows-introduction (Accessed: Jun. 18, 2025).
29. D.N. Răceanu, C.V. Marian, *Cybersecurity virtual labs for pentesting education*, The 13th International Symposium on Advanced Topics in Electrical Engineering, Romania (2023).
30. R.Ș. Lungu, O.A. Frasin, C.V. Marian, *Design and implementation of lightweight virtualized firewalls for industrial cybersecurity and medical services*, The 2025 IEEE International Black Sea Conference on Communications and Networking, Moldova (2025).
31. B.-I. Ciubotaru, V.-G. Sasu, A. Vasilateanu, A. Mitrea, N. Goga, *Improved secure internet of things system using web services and low-power single-board computers*, The 8th IEEE International Conference on E-Health and Bioengineering, Romania (2020).
32. G. Thiyagarajan, V. Bist, P. Nayak, *The hidden dangers of outdated software: a cyber security perspective*, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, **11** (2025).
33. V. Duvvur, *Securing the future: strategies for modernizing legacy systems and enhancing cybersecurity*, Journal of Artificial Intelligence & Cloud Computing, **1**, pp. 1–3 (2022).