International Journal of Research Publication and Reviews

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

# PHISSING ATTACK DETECTION

*Vishnupriya.S[1],Monisha.M[2]*

Assistant Professor [1], Student [2]

Department of Computer Science and Engineering, Tagore Institute Of Engineering And Technology, Deviyakurichi ,Salem ,Tamil Nadu,India

ABSTRACT :

The rapid expansion of technology and IoT devices has amplified the need for robust security measures, as vulnerabilities like Log4j expose critical systems to exploitation. Log4j, a widely used logging library in Java-based software, was discovered to have a zero-day vulnerability in December 2021, enabling attackers to inject and execute malicious code via its communication functionality. This poses significant threats to IoT ecosystems due to Java's prominence in IoT application development. To counteract such vulnerabilities, a honeypot-based solution is proposed, integrating an internal scanner to detect Log4jShell vulnerability and SSH-based threats, alerting the relevant teams. This approach enhances security by adding an additional layer of protection to existing systems.

**Keywords:** Technology expansion, IoT devices, Log4j vulnerability, zero-day exploit, Java, Log4jShell, honeypot, internal scanner, cybersecurity, SSH threats.

## I.INTRODUCTION :

Common Vulnerabilities and Exposures (CVE) is a publicly accessible standard list of known security flaws and vulnerabilities in hardware and software systems. Each CVE entry includes a unique identifier, a description of the vulnerability, its impact, and its severity, helping organizations identify, prioritize, and manage risks effectively. CVE entries are assigned by the CVE Numbering Authority (CNA) and are frequently updated as new vulnerabilities are discovered. By proactively identifying and addressing vulnerabilities, organizations can enhance their cybersecurity posture and mitigate the risk of exploitation by attackers. Java, developed in 1995, is one of the most widely used programming languages globally, powering over 3 billion devices. Known for its versatility, Java is employed in mobile apps, web apps, desktop applications, and servers across various platforms, including Windows, macOS, and Linux. Java's usability, scalability, and security features make it a preferred choice in the industry. However, its popularity also makes Java-based systems a prime target for cyberattacks, necessitating robust security measures to protect sensitive applications and data. In December 2021, a critical vulnerability in the Log4j logging framework, known as Log4Shell (CVE-2021-44228), was discovered. This severe security flaw allows attackers to inject malicious code into vulnerable systems, enabling remote code execution. The widespread use of Log4j in Java-based applications, including cloud services and web servers, amplified the threat, with cybercriminals launching millions of exploitation attempts per hour. The vulnerability has caused significant concern across various sectors, including government, healthcare, and finance, emphasizing the need for immediate and effective countermeasures. The exploitation of Log4j's JNDI lookup feature underscores the severity of the vulnerability. Attackers can craft malicious requests that, if processed by a Log4j-enabled application, execute arbitrary code on the system. This vulnerability has been linked to ransomware attacks, data breaches, and system compromises, with prominent examples such as Aquatic Panda targeting academic institutions. Organizations must adopt a proactive approach to address this issue by updating to secure versions of Log4j, implementing firewalls, and deploying monitoring tools to detect and prevent malicious activity. Remote Code Execution (RCE) is a critical security flaw that enables attackers to execute arbitrary code on remote systems. This type of attack can arise from user inputs being evaluated by programming language parsers.

RCE attacks are highly destructive, potentially leading to privilege escalation, data theft, and system compromise. Effective prevention requires regular software updates, access controls, and rigorous security assessments to identify and mitigate vulnerabilities. Honeypots play a significant role in cybersecurity as decoy systems designed to detect, deflect, or study unauthorized access attempts. These systems attract attackers by simulating vulnerabilities, allowing security teams to monitor malicious activities and gather intelligence. Honeypots come in various forms, including high-interaction and low-interaction types, as well as virtual honeypots and honeynets. By deploying honeypots, organizations can gain valuable insights into attacker behavior while safeguarding critical systems. Addressing the Log4j vulnerability requires a multi-faceted approach. Organizations must first identify the presence of Log4j in their systems and evaluate their vulnerability. Upgrading to the latest secure version of Log4j is critical, along with implementing additional security measures such as traffic monitoring and restricting access to known malicious IP addresses. Security patch management strategies can further enhance defenses, ensuring all systems are up-to-date with the latest patches. The widespread impact of the Log4j vulnerability has highlighted the importance of proactive cybersecurity practices. While patches and updates are essential, maintaining robust security involves continuous monitoring and regular vulnerability assessments. The incident underscores the need for developers and organizations to prioritize security, adopting preventive measures to safeguard systems against emerging threats. In conclusion, the Log4j vulnerability serves as a stark reminder of the importance of vigilance in cybersecurity. The development of projects aimed at preventing such exploits is crucial for protecting sensitive systems and data. By employing advanced techniques like honeypots, strengthening defenses against RCE, and fostering collaboration among developers and security experts, organizations can mitigate risks and ensure the resilience of their applications and services.

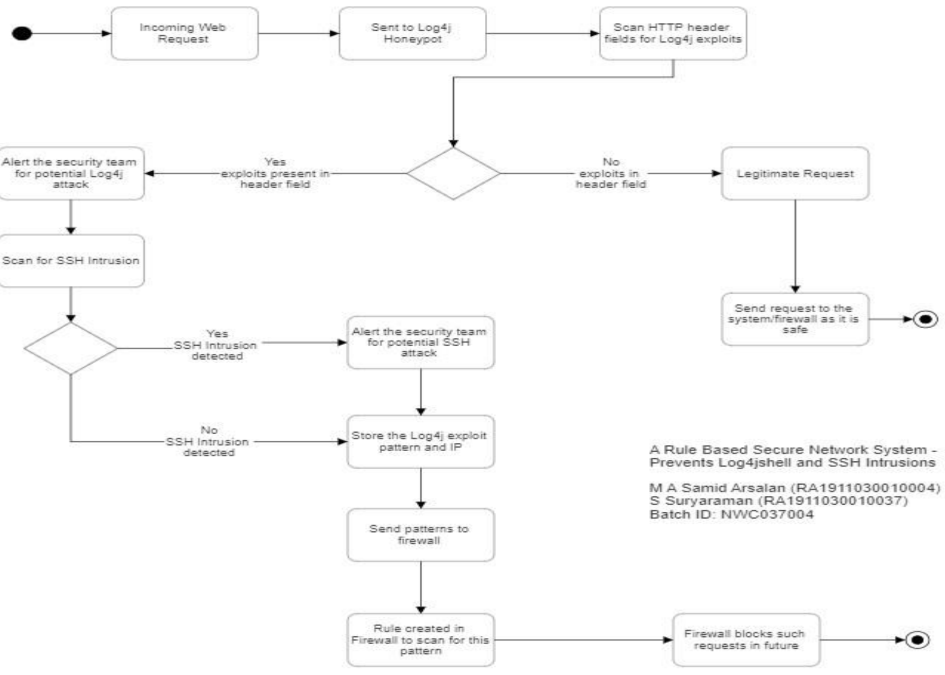
## II.RELATED WORKS :

For any successful project or novel idea, studying and surveying are important. The analysis of existing models is essential, and only after that can a project reach its full fruition. A literature survey, often referred to as a literature review, involves a thorough evaluation of the existing body of knowledge on a given subject or research question. It aims to provide a comprehensive review of the research on a specific topic, identify gaps and inconsistencies, and highlight areas requiring further investigation. Conducting a literature survey entails defining the research question, identifying relevant sources, critically evaluating these sources, organizing information, and synthesizing findings. This enables researchers to build upon existing knowledge and strengthen their arguments in academic or practical applications. This review focuses on the Log4j vulnerability, its implications, and current mitigation strategies. The Log4j vulnerability impacts versions 2.0 and earlier, posing a critical security risk. Exploiting this vulnerability allows attackers to execute arbitrary code on servers by leveraging the Java Naming and Directory Interface (JNDI) lookup functionality. Such exploitation can lead to severe consequences, including unauthorized access to sensitive data, system compromise, and malicious code execution, resulting in significant financial, reputational, and legal repercussions. Various research works have explored methods to mitigate the Log4j vulnerability and its exploitation. Kaushik Keshav et al. examined the phases of identifying, scanning, exploiting, and patching vulnerabilities, emphasizing runtime security tools to detect reverse shells. They analyzed social media sentiment on the issue and advocated for container lifecycle admission controls in CI/CD pipelines. Koniaris Ioannis et al. demonstrated a specialized honeypot for Secure Shell (SSH) services, gathering and analyzing data such as login attempts and attack sources. They showcased how such tools could enhance security by monitoring and visualizing attack patterns. Similarly, Srinivasa Shreyas et al. deployed honeypots simulating Log4j vulnerabilities to study LDAP attacks, highlighting brute force methods and suspicious queries. Other researchers, like Juvonen Artturi, investigated Log4j exploitation in mission-critical systems like aviation and maritime communication protocols, illustrating the potential for catastrophic outcomes without proper safeguards. Their findings underscored the importance of preventative measures like intrusion detection systems to mitigate risks. Varlioglu's work analyzed fileless attacks leveraging vulnerabilities like Log4j, emphasizing the challenges of detection due to their stealthy nature. These studies highlight the evolving landscape of cybersecurity threats and the necessity for robust defenses. Hybrid models combining signature-based and anomaly-based detection techniques have also emerged to address both known and novel threats in cloud-based systems. Such approaches, tested on datasets like UNSW-NB15 and NSL-KDD, have demonstrated high detection rates, showcasing their potential for securing cloud infrastructures. Additionally, researchers have explored the intricacies of encryption, honeypots, and advanced detection mechanisms to address security concerns in cloud environments. These methodologies collectively provide a framework for enhancing security measures against vulnerabilities like Log4j while paving the way for future research and development in cybersecurity.

## III.PROPOSED SYSTEM :

The technique of our hypothetical secure rule-based system is that a deployed honeypot has little contact, notifies the security team, and examines the payload pattern to enable the creation of a firewall rule to stop future attacks of this type. The honeypot also finds SSH-based threat-incidents and notifies for any prospective SSH-based assaults. In addition to having a honeypot, this rule offers another level of security. The model suggests a method in which the honeypot scans the arriving web request, identifies the design of the log4j exploit, and crafts a rule in the firewall based on that pattern. The firewall rule that is set up prevents requests with patterns that are similar to those it has already seen from coming through in the future. The model also proposes a built-in scanner that can take in a unified resource link as input and will send a series of payloads that are JNDI based and the scanner will figure out whether the given unified resource link is vulnerable to the CVE-2021- 45046 or not. There are two modules which are working in conjunction. The SSH honeypot and the H0ney4Log, which is the scanner that will determine whether the website is vulnerable to the CVE. The H0ney4Log has various web application firewall payloads that gets into the network undetected. The payload has both JNDI, Java Naming Directory Interface, as well as DNS, Domain Name System compatibility.

**Figure 1: System Architecture of proposed system**


Incoming Web Request
Sent to Log4j Honeypot
Scan HTTP header fields for Log4j exploits
Alert the security team for potential Log4j attack
Yes exploits present in header field
No exploits in header field
Legitimate Request
Scan for SSH Intrusion
Send request to the system/firewall as it is safe
Yes SSH Intrusion detected
Alert the security team for potential SSH attack
No SSH Intrusion detected
Store the Log4j exploit pattern and IP
A Rule Based Secure Network System - Prevents Log4jshell and SSH Intrusions
M A Samid Arsalan (RA1911030010004)
S Suryaraman (RA1911030010037)
Batch ID: NWC037004
Send patterns to firewall
Rule created in Firewall to scan for this pattern
Firewall blocks such requests in future

## IV. MODULES :

Python requires various modules or libraries to run functions essential for specific projects. For creating a honeypot, several libraries are integral. **Argparse**, for instance, allows developers to build user-friendly command-line interfaces. It parses arguments, provides default values, generates help and error messages, and manages complex command-line arguments like subcommands and optional parameters, making it indispensable for robust command-line application development. **Threading** offers a mechanism to run concurrent operations efficiently. Unlike processes, threads execute faster and are lightweight. In the context of a honeypot, threading ensures rapid creation and severance of connections. With tools like locks and semaphores, it manages synchronization, preventing race conditions and ensuring smooth execution. **Socket** provides low-level network communication capabilities, enabling client-host interactions. Developers can use it to create sockets for transmitting and receiving data over networks. Its adaptability supports protocols like TCP and UDP, making it versatile for building network applications ranging from simple client-server programs to distributed systems. The **sys** module interacts directly with the Python interpreter, giving developers control over runtime environments. It offers access to essential functions like modifying module search paths, accessing command-line arguments, and handling garbage collection. This module is vital for troubleshooting and optimizing Python programs, especially for understanding the runtime context. The **os.path** module and its `isfile()` function facilitate file handling tasks such as path manipulation and verification. This module determines whether a specified file exists, enabling error-free file operations. Its system-specific behavior ensures compatibility across operating systems, a critical feature for seamless file management. **Traceback** provides tools to format and extract stack traces, mimicking an interpreter's error reporting. It aids debugging by offering detailed error reports with file names, line numbers, and function paths. Developers can manipulate stack traces for custom error handling, logging, and optimizing exception reporting. **Logging** manages the creation and maintenance of logs. It records crucial details like SSH versions, connection status, credentials, and error messages. With adjustable verbosity levels and contextual logging capabilities, this module simplifies debugging and performance optimization. The **JSON** module handles data exchange in JavaScript Object Notation, simplifying interactions between Python objects and JSON data. It works with `os.path` to process command files, offering robust options for encoding, decoding, and manipulating complex JSON data structures. Lastly, **Paramiko** is the backbone for secure remote connections using the SSH protocol. It supports authentication, command execution, and secure file transfers. With advanced features like SSH tunneling and custom encryption algorithms, it enhances the security and efficiency of the honeypot.

## V.RESULTS AND DISCUSSION :

The honeypot system effectively detected and logged intrusion attempts, showcasing its reliability. With Paramiko, secure SSH connections were simulated, and unauthorized access attempts were captured. Logging provided detailed insights into activities, aiding in monitoring and analysis. The use of Threading ensured smooth handling of multiple connections concurrently, enhancing performance. Modules like Socket and Argparse streamlined network communication and command-line interaction. Error handling with Traceback and sys ensured robustness, while JSON and os.path facilitated efficient data management. Overall, the integrated use of these modules resulted in a functional and secure honeypot system, highlighting their importance in cybersecurity applications.

## VI.CONCLUSION :

In conclusion, it successfully detected intrusion attempts and provided valuable insights for analyzing malicious activities. Its modular approach demonstrated the importance of integrating versatile Python libraries for enhanced cybersecurity.

REFERENCE :

1. Oxford Analytica. "US government targets open-source software flaws." *Emerald Expert Briefings* oxan-es (2022).Jones, Amanda. ”Security Posture: A Systematic Review of Cyber Threats and Proactive Security.” (2022).
2. Sopariwala, Shein, Enda Fallon, and Mamoona Naveed Asghar. "Log4jPot: Effective Log4Shell Vulnerability Detection System." 2022 33rd Irish Signals and Systems Conference (ISSC). IEEE, 2022.
3. Zhang, Feng, et al. "Honeypot: a supplemented active defense system for network security." Proceedings of the Fourth International Conference on Parallel and Distributed Computing, Applications and Technologies. IEEE, 2003.
4. Biswas, Saikat, et al. "A study on remote code execution vulnerability in web applications." International Conference on Cyber Security and Computer Science (ICONCS 2018). 2018.
5. Ramachandra, A. C., and Subhrajit Bhattacharya. "Literature survey on log-based anomaly detection framework in cloud." Computational Intelligence in Pattern Recognition: Proceedings of CIPR 2020 (2020): 143-153.
6. Kaushik, Keshav, Alpana Dass, and Ayush Dhankhar. "An approach for exploiting and mitigating Log4J using Log4Shell vulnerability." 2022 3rd International Conference on Computation, Automation and Knowledge Management (ICCAKM). IEEE, 2022.
7. Chen, Boyuan, and Zhen Ming Jiang. ”A survey of software log instrumentation.” ACM Computing Surveys (CSUR) 54.4 (2021): 1-34.
8. Koniaris, Ioannis, Georgios Papadimitriou, and Petros Nicopolitidis. "Analysis and visualization of SSH attacks using honeypots." Eurocon 2013. IEEE, 2013.
9. Patel, Krupa C., and Priyanka Sharma. ”A Review paper on pfsense-an Open source firewall introducing with different capabilities & customiza- tion.” IJARIIE 3 (2017): 2395-4396.
10. Srinivasa, Shreyas, Jens Myrup Pedersen, and Emmanouil Vasilomanolakis. "Deceptive directories and “vulnerable” logs: a honeypot study of the LDAP and log4j attack landscape." 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). IEEE, 2022.

11. Li, Hongxia, Junming Chen, and Xin Jin. ”An outlook on network honeypot.” 2011 International Conference on Computer Science and Service System (CSSS). IEEE, 2011.
12. Juvonen, Artturi, et al. "On Apache Log4j2 Exploitation in Aeronautical, Maritime, and Aerospace Communication." IEEE Access 10 (2022): 86542-86557.
13. Arifianto, Ridho Maulana, Parman Sukarno, and Erwid Musthofa Jadied. ”An SSH honeypot architecture using port knocking and intrusion detection system.” 2018 6th International Conference on Information and Communication Technology (ICoICT). IEEE, 2018.
14. Varlioglu, Said, et al. "The dangerous combo: Fileless malware and cryptojacking." SoutheastCon 2022 (2022): 125-132.
15. Kelly, Christopher, et al. "A comparative analysis of honeypots on different cloud platforms." *Sensors* 21.7 (2021): 2433.
16. Vashishtha, Lalit Kumar, Akhil Pratap Singh, and Kakali Chatterjee. "HIDM: A Hybrid Intrusion Detection Model for Cloud Based Systems." Wireless Personal Communications 128.4 (2023): 2637-2666.
17. Negi, Poorvika Singh, Aditya Garg, and Roshan Lal. "Intrusion detection and prevention using honeypot network for cloud security." *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. IEEE, 2020.
18. Khan, Minhaj Ahmad. "A survey of security issues for cloud computing." Journal of network and computer applications 71 (2016): 11-29.
19. Buzzio-Garcia, Jorge. "Creation of a High-Interaction Honeypot System based-on Docker containers." *2021 Fifth World Conference on Smart Trends in Systems Security and Sustainability (WorldS4)*. IEEE, 2021.
20. Albugmi, Ahmed, et al. "Data security in cloud computing." 2016 Fifth international conference on future generation communication technologies (FGCT). IEEE, 2016.
21. Kambow, Navneet, and Lavleen Kaur Passi. "Honeypots: The need of network security." *International Journal of Computer Science and Information Technologies* 5.5 (2014): 6098-6101.
22. Leandros Maglaras Doubleday, Harry, and Helge Janicke. "SSH honeypot: building, deploying and analysis." (2016).
23. Manzanares, Antonio Izquierdo, et al. "Attacks on port knocking authentication mechanism." *International Conference on Computational Science and Its Applications*. Springer, Berlin, Heidelberg, 2005.
24. Sharma, Manish, and Shivkumar Singh Tomar. "Attack Detection and Security in Remote Code Execution." International Journal of Computer Applications 114.14 (2015).
25. Ramsbrock, Daniel, Robin Berthier, and Michel Cukier. "Profiling attacker behavior following SSH compromises." *37th Annual IEEE/IFIP international conference on dependable systems and networks (DSN'07)*. IEEE, 2007.
26. Sadasivam, Gokul Kannan, Chittaranjan Hota, and Bhojan Anand. "Classification of SSH attacks using machine learning algorithms." 2016 6th International Conference on IT Convergence and Security (ICITCS). IEEE, 2016.
27. Park, Jeonghoon, et al. "Network log-based SSH brute-force attack detection model." CMC-Computers Materials & Continua 68.1 (2021): 887-901.
28. Choi, Brendan. "Python Network Automation Labs: SSH paramiko and netmiko." Introduction to Python Network Automation: The First Journey. Berkeley, CA: Apress, 2021. 583-628.
29. Adee, Rose, and Haralambos Mouratidis. "A dynamic four-step data security model for data in cloud computing based on cryptography and steganography." Sensors 22.3 (2022): 1109.
30. Alfaidi, Arij, and Sudhanshu Semwal. "The right to be forgotten privacy and security in blockchain with multi-authority based chameleon hash function MAP-ABCH solution." Advances in Information and Communication: Proceedings of the 2022 Future of Information and Communication Conference (FICC), Volume 1. Cham: Springer International Publishing, 2022.