Indian Journal of Computer Science and Technology
https://www.doi.org/10.59256/indjcst.20260502031
Volume 5, Issue2 (May-August 2026), PP: 270-275.
www.indjcst.com
ISSN No: 2583-5300

Ransomware Threat Detection and Automated Response System Using Wazuh and Python

Mohan S[1], Pradeep G[2], Madhuravel P[3], Deepakumaar R[4], J. Dhivya[5]
[1,2,3,4] Department of Computer Science and Engineering (Cyber Security), United Institute of Technology, Coimbatore, Tamil Nadu, India.
[5]Assistant Professor, United Institute of Technology, Coimbatore, Tamil Nadu, India.

To Cite this Article: *Mohan S[1], Pradeep G[2], Madhuravel P[3], Deepakumaar R[4], J. Dhivya [5], "Ransomware Threat Detection and Automated Response System Using Wazuh and Python", Indian Journal of Computer Science and Technology, Volume 05, Issue 02 (May-August 2026), PP: 270-275.*

Abstract: *The increasing evolution of ransomware attacks has created significant challenges for traditional cybersecurity systems that rely primarily on signature-based detection and manual incident response mechanisms. According to the National Institute of Standards and Technology (NIST) Special Publication 1800-26, maintaining data integrity through timely detection and response is critical for minimizing organizational downtime and preventing data corruption caused by ransomware and other destructive events. Inspired by the NIST data integrity and ransomware response framework, this paper presents an intelligent and automated Ransomware Threat Detection and Response System designed to identify ransomware behaviour in real time and minimize data loss through rapid containment actions. The proposed framework integrates Wazuh for continuous system monitoring, behavioural analysis, log correlation, and threat detection with Python-based automated response mechanisms. The system continuously monitors critical indicators such as abnormal file modifications, suspicious process execution, unusual CPU and memory utilization, and shadow copy deletion attempts. Upon detecting ransomware-like activity, automated response actions including malicious process termination, network isolation, alert generation, and system containment are executed instantly to prevent further encryption and propagation. The framework is deployed in a virtualized environment consisting of Ubuntu Server, Windows client systems, and Kali Linux attack simulation machines to evaluate real-world attack scenarios safely. Experimental results demonstrate that the proposed behaviour-based approach effectively detects ransomware activity at early stages with faster response times and reduced system impact compared to conventional reactive security solutions. The proposed system offers a scalable, lightweight, and cost-effective cybersecurity solution suitable for academic, enterprise, and small-scale organizational environments.*

Key Word: *Ransomware Detection, Behavioural Analysis, Automated Response System, Wazuh, Threat Monitoring, Incident Response, Malware Detection, Python Automation, Network Isolation, Virtual Machine Security, Real-Time Threat Detection.*

I. INTRODUCTION

The rapid growth of digital technologies and interconnected systems has increased the risk of ransomware attacks. Ransomware encrypts critical files and demands payment for recovery, causing data loss and operational disruption. Traditional security solutions mainly rely on signature-based detection and are ineffective against modern and zero-day ransomware attacks.

This work proposes a Ransomware Threat Detection and Automated Response System that uses behaviour-based monitoring and automated response mechanisms. The system continuously monitors suspicious activities such as abnormal file modifications, malicious processes, and unusual system behaviour using Wazuh and Python automation scripts. Upon detection, the system automatically performs actions like process termination, network isolation, and alert generation.

The framework is implemented in a virtualized environment using Ubuntu Server, Windows, and Kali Linux for safe ransomware simulation and testing. The proposed solution provides a lightweight, scalable, and cost-effective approach for ransomware protection.

1.1 Research Contributions

The principal contributions of this research are as follows:

- Development of a behaviour-based ransomware detection framework capable of identifying suspicious activities such as abnormal file modifications, unauthorized process execution, unusual CPU and memory utilization, and shadow copy deletion attempts in real time.
- Integration of Wazuh for centralized log collection, endpoint monitoring, rule-based correlation, and real-time threat analysis across distributed systems.
- Design and implementation of automated incident response mechanisms using Python scripts to perform malicious process termination, system isolation, alert generation, and rapid containment during ransomware attacks.

- Creation of a secure virtualized cybersecurity environment consisting of Ubuntu Server, Windows client systems, and Kali Linux attack simulation machines for realistic ransomware behaviour analysis and testing.
- Real-time monitoring and analysis of ransomware indicators including mass file encryption activity, suspicious privilege escalation, abnormal resource consumption, and backup deletion behaviour.
- Implementation of automated alerting and response workflows that significantly reduce ransomware propagation, encryption impact, and manual response delays compared to traditional security approaches.

II.LITERATURE SURVEY

Cybersecurity threats have evolved rapidly, with ransomware emerging as one of the most destructive malware types targeting modern systems. Traditional security mechanisms such as antivirus software and signature-based intrusion detection systems rely on known attack patterns, making them ineffective against zero-day and polymorphic ransomware that continuously change their behaviour.

Behaviour-based ransomware detection has gained attention as an effective alternative, as it monitors system activities like file modifications, process execution, registry changes, and unusual encryption behaviour. Studies show that behavioural analysis improves detection of unknown threats by focusing on malicious patterns rather than static signatures. Modern SIEM and endpoint monitoring platforms such as Wazuh enhance this approach by enabling centralized log collection, file integrity monitoring, rule-based detection, and real-time alerting across distributed systems.

Automated incident response techniques are increasingly important due to the fast execution speed of ransomware attacks. Delayed manual response often leads to severe data loss, whereas automation enables immediate actions such as process termination, network isolation, and alert generation. Virtualized environments like Ubuntu, Windows, and Kali Linux are widely used for safe ransomware simulation and analysis, helping researchers study attack behaviour and test defence mechanisms without affecting real systems.

Despite these advancements, a key research gap exists in integrating real-time behavioural monitoring, centralized detection, and automated response into a single lightweight system. Most existing solutions focus either on detection or require complex infrastructure. This work addresses this gap by proposing an integrated ransomware detection and automated response system using Wazuh and Python to enable early detection, rapid containment, and reduced data loss.

III.PROBLEM STATEMENT AND MOTIVATION

3.1 Limitations of Traditional Security Systems

Traditional security solutions such as antivirus software and firewalls mainly rely on signature-based detection, making them ineffective against modern zero-day and polymorphic ransomware attacks. As a result, ransomware can encrypt files before detection occurs.

3.2 Delayed Detection and Manual Response

Ransomware spreads rapidly, while many existing systems depend on manual monitoring and response. Delays in identifying threats, isolating infected systems, and stopping malicious processes increase data loss and operational downtime.

3.3 Lack of Real-Time Behavioural Analysis

Most traditional tools focus on known malware signatures instead of monitoring system behaviour. Suspicious activities such as mass file modifications, abnormal CPU usage, malicious process execution, and shadow copy deletion may remain undetected during early attack stages.

3.4 Absence of Automated Containment

Many security solutions only generate alerts without taking immediate action. Without automated responses like process termination and network isolation, ransomware can continue spreading across systems and networks.

3.5 Motivation of the Proposed Work

The proposed system aims to provide early ransomware detection and rapid automated response using Wazuh and Python-based automation. The framework focuses on real-time behavioural monitoring, automatic containment, and cost-effective ransomware protection.

IV.PROPOSED METHODOLOGY

The proposed system implements a continuous ransomware detection and automated response pipeline using Wazuh for monitoring, Python for automation, and virtualized environments for attack simulation. The architecture is organized into six functional stages:

Stage 1: System & Log Data Acquisition

The data acquisition layer is handled by the Wazuh Agent installed on monitored endpoints (Windows user system and Linux server). It continuously collects system logs, file integrity monitoring data, process execution logs, and authentication activities. These logs are forwarded to the Wazuh Manager for centralized analysis in real time.

Stage2: Feature Extraction and Activity Monitoring
From collected logs, key behavioural indicators are extracted such as:

- File modification frequency
- Unusual process creation (e.g., unknown executables)
- High volume file rename or encryption-like activity
- CPU and disk usage spikes
- Shadow copy deletion attempts

These features represent ransomware-like behavioural patterns used for detection.

Stage 3: Behavioural Anomaly Detection (Wazuh Rules Engine)

Wazuh decoders and correlation rules analyze incoming logs to detect abnormal behaviour. Rule-based logic identifies ransomware indicators such as rapid file encryption patterns, mass file changes, and privilege escalation attempts. When thresholds are exceeded, the system flags the activity as suspicious or malicious without relying solely on known malware signatures.

Stage 4: Signature and Rule-Based Threat Correlation

Detected events are correlated with predefined Wazuh security rules and MITRE ATT&CK mappings. This stage classifies threats such as:

- File encryption behaviour (ransomware execution pattern)
- Unauthorized system modification
- Suspicious script execution (PowerShell / Bash abuse)
- Lateral movement attempts

This enhances detection accuracy by combining behavioural and signature-based intelligence.

Stage 5: Automated Response Execution (Python Integration)

Once a ransomware-like activity is detected, Python scripts are triggered automatically through Wazuh Active Response. The system performs containment actions such as:

- Terminating malicious processes
- Blocking attacker IP addresses via firewall rules
- Isolating the affected endpoint from the network
- Stopping file encryption processes
- Initiating backup protection workflows

This ensures immediate mitigation and minimizes data loss.

Stage 6: Alerting, Logging, and Reporting

All detected events and responses are logged in the Wazuh dashboard. Real-time alerts are sent via email and Telegram bot notifications. Alerts include:

- Severity level (Low / Medium / High / Critical)
- Source system details
- Type of ransomware behaviour detected
- Action taken by automated response system

This enables continuous monitoring, forensic analysis, and post-incident review.

V.SYSTEM ARCHITECTURE

The Ransomware Threat Detection and Automated Response System architecture is organized into multiple functional modules operating through a centralized Wazuh Manager and a Python-based automation engine. The system continuously monitors endpoint activities, analyzes logs, and triggers automated responses upon detecting ransomware-like behaviour.

Module	Function	Technology Used
Monitoring	Log & activity tracking	Wazuh
File Integrity Management	Detect file changes	Wazuh FIM
Detection	Identify threats	Wazuh Rules
Analysis	Behaviour monitoring	Wazuh & Python
Response	Alert & auto actions	Wazuh & Python

TABLE I: Ransomware Detection System Modules And Technologies

5.1 Behavioural Detection and Rule-Based Analysis Engine

The detection engine relies on Wazuh rule-based correlation combined with behavioural indicators of ransomware activity. Key monitored parameters include:

- Rapid file modifications or encryption patterns
- High frequency of file rename operations
- Suspicious process execution (e.g., unknown binaries)
- Unauthorized access to sensitive directories
- Shadow copy deletion attempts

Rules are defined using Wazuh decoders and custom detection logic. Alerts are triggered when predefined thresholds are exceeded.

5.2 File Integrity Monitoring (FIM) Architecture

The File Integrity Monitoring system continuously tracks changes in critical system directories.

Monitored activities include:

- File creation
- File modification
- File deletion
- Permission changes

FIM operates in real-time and compares system state against a trusted baseline. Any deviation is flagged as suspicious and forwarded to the Wazuh Manager for analysis.

5.3 Automated Response Engine

The automated response engine is implemented using Python scripts integrated with Wazuh Active Response.

When ransomware activity is detected, the system performs:

- Termination of malicious processes
- Blocking of suspicious user sessions
- Network isolation of compromised endpoints
- Execution of system lockdown scripts
- Triggering backup protection mechanisms

The response workflow is executed instantly to minimize encryption impact.

5.4 Alerting and Notification System

The alert engine provides multi-channel notification support including:

- Email alerts via SMTP
- Telegram bot notifications
- Wazuh dashboard alerts

Each alert contains:

- Event type
- Severity level
- Affected host
- Timestamp
- Suggested mitigation action

VI.IMPLEMENTATION

6.1 Development Environment

- **Operating System:** Ubuntu Server 24.04 LTS
- **Security Platform:** Wazuh SIEM (Manager + Agent)
- **Language:** Python 3.10+ (automation & response scripts)
- **Database:** Wazuh Indexer
- **Monitoring Tool:** Wazuh Dashboard (OpenSearch/Kibana)
- **Simulation Environment:** Kali Linux + Windows VM
- **Automation Layer:** Python
- **Deployment:** system service (auto-start on boot)

6.2 Wazuh Integration and Log Processing

The system integrates Wazuh Agents across endpoints to continuously forward security logs to the Wazuh Manager. The log processing pipeline includes:

- Log decoding
- Rule-based classification

- Event correlation
- Severity scoring
- Alert generation

This ensures real-time visibility of system activity and ransomware behaviour detection.

6.3 Attack Simulation Module

The attack simulation module replicates ransomware-like behaviour for testing detection capabilities. Simulated scenarios include:

- Mass file encryption behaviour
- Rapid file renaming attacks
- Process injection simulation
- Privilege escalation attempts
- Shadow copy deletion simulation

These scenarios are executed using Python scripts and Kali Linux-based tools.

6.4 Response Automation Workflow

The response workflow follows a structured pipeline:

1. Event detected by Wazuh rule engine
2. Alert generated with severity classification
3. Python automation script triggered
4. Active response executed on endpoint
5. System state restored or isolated
6. Incident logged for analysis

This ensures minimal delay between detection and mitigation.

6.5 Deployment and Accessibility

The system is deployed as a centralized security monitoring solution using Wazuh architecture.

Deployment features include:

- Auto-start Wazuh Manager via system
- Agent-based endpoint monitoring
- Cross-platform support (Windows & Linux agents)
- Web-based dashboard for SOC monitoring
- Scalable architecture for multiple endpoints

VII.RESULTS AND DISCUSSION

The proposed ransomware detection and automated response system successfully identified ransomware-like behaviour during testing in a virtualized environment. The system was able to detect abnormal activities such as rapid file modifications, unauthorized process execution, and suspicious encryption patterns in real time using Wazuh monitoring and rule-based detection. Once malicious behaviour was detected, the automated response mechanism was triggered immediately. The system successfully terminated harmful processes, generated real-time alerts, and performed network isolation to prevent further spread of the attack. This significantly reduced potential data loss compared to traditional manual response methods.

The behaviour-based detection approach proved more effective than signature-based methods, as it was able to identify unknown and simulated ransomware attacks. Integration of Python-based automation improved response speed and ensured quick mitigation without human intervention.

Overall, the system demonstrated high effectiveness in early detection, fast containment, and automated mitigation of ransomware threats. However, further improvements such as machine learning-based detection and cloud integration could enhance adaptability and scalability in real-world environments.

VIII.CONCLUSION

This paper presented an Integrated Ransomware Threat Detection and Automated Response System that addresses the limitations of traditional signature-based security mechanisms through behaviour-based monitoring, real-time threat detection, and automated incident response using Wazuh and Python.

The system effectively detected ransomware-like activities such as rapid file encryption, abnormal process execution, and suspicious system behaviour during simulated attack scenarios in a controlled virtual environment. The integration of real-time monitoring, rule-based correlation, and automated response actions such as process termination and network isolation significantly reduced response time and potential data loss.

The proposed approach demonstrated improved detection efficiency for both known and unknown ransomware threats compared to conventional antivirus systems. The use of virtualization environments (Ubuntu, Kali Linux, and Windows) ensured safe testing and validation of attack scenarios without impacting real systems. Overall, the system provides a lightweight, scalable, and effective solution for early ransomware detection and automated mitigation.

IX.ACKNOWLEDGMENT

The authors express sincere gratitude to Assistant Professor Ms. J. Dhivya, Project Guide, Department of Computer Science and Engineering (Cyber Security), for her valuable guidance, continuous support, and encouragement throughout the development of this project.

The authors also acknowledge the Department of Computer Science and Engineering (Cyber Security), for providing the necessary laboratory facilities, virtual machine infrastructure, and research environment required for successful completion of this work.

REFERENCES

1. M. Roesch, "Snort: Lightweight intrusion detection for networks," Proc. USENIX LISA, 1999, pp. 229–238.
2. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD Cup 99 dataset," in Proc. IEEE CISDA, 2009.
3. N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in Proc. MilCIS, 2015.
4. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset," in Proc. ICISSP, 2018, pp. 108–116.
5. R. Lippmann et al., "Evaluating intrusion detection systems: The 1998 DARPA evaluation," in Proc. DARPA Information Survivability Conference, 2000.
6. A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," IEEE Commun. Surveys Tuts., vol. 18, no. 2, pp. 1153–1176, 2016.
7. R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in Proc. IEEE Symp. Security and Privacy, 2010.
8. G. Kim, S. Lee, and S. Kim, "A hybrid intrusion detection method integrating anomaly and misuse detection," Expert Syst. Appl., vol. 41, no. 4, pp. 1690–1700, 2014.
9. A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in Proc. EAI SecureComm, 2016.
10. A. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," J. Netw. Comput. Appl., vol. 60, pp. 19–31, 2016.
11. E. M. Hutchins, M. J. Cloppert, and R. M. Amin, "Intelligence-driven computer network defense using the kill chain model," Lockheed Martin White Paper, 2011.
12. MITRE Corporation, "MITRE ATT&CK Framework," 2024.
13. Wazuh Inc., "Wazuh: Security Information and Event Management (SIEM) platform documentation," 2025.
14. S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Chalmers University Technical Report, 2000.
15. C. Modi, D. Patel, B. Borisaniya, H. Patel, M. Rajarajan, and A. Patel, "A survey of intrusion detection techniques in cloud," J. Netw. Comput. Appl., 2013.
16. C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and other botnets," IEEE Computer, vol. 50, no. 7, pp. 80–84, 2017.
17. A. O. Arabo, I. Brown, and A. El-Moussa, "Ransomware: Issues and defense mechanisms," in Proc. IEEE Cybersecurity Conf., 2017.
18. P. Garnaeva et al., "Ransomware evolution and detection techniques," Kaspersky Security Bulletin, 2023.
19. J. S. Smith and K. R. Johnson, "Behavior-based anomaly detection for endpoint security systems," IEEE Access, vol. 8, pp. 123456–123467, 2020.
20. H. Hindy, D. Brosset, E. Bayne, et al., "A taxonomy of ransomware threats and mitigation strategies," IEEE Access, vol. 9, pp. 147792–147809, 2021.