International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 21 No. 4 (2025)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, No. 4, 2025

Research Paper

Development Of Cyber Threat Intelligence Tools Using Python

First Author: T. Suresh, Associate professor, Gokula Krishna College of Engineering, Sullurpet, Tirupati District, AP

Second Author: Harathi Avsl PG Scholar, Gokula Krishna College of Engineering, Sullurpet, Tirupati District, AP

Abstract:
The increasing complexity of modern cyberattacks demands intelligent systems capable of identifying, analyzing, and predicting threats in real time. This study focuses on the development of cyber threat intelligence (CTI) tools using Python to automate the collection, analysis, and correlation of threat data from diverse sources. The proposed framework integrates Python-based modules for data harvesting, machine learning–driven threat classification, and entity extraction, enabling the transformation of raw indicators into actionable intelligence. Advanced analytics techniques, including natural language processing (NLP) and graph-based correlation, are employed to detect patterns, associate attack entities, and forecast potential threat vectors. Furthermore, the system leverages open-source platforms such as MISP and Wazuh, enhancing interoperability and intelligence sharing within security ecosystems. Experimental validation demonstrates that the developed Python-powered CTI tools can significantly improve situational awareness, reduce analysis time, and support proactive defense strategies. This work contributes to the evolution of automated, adaptive, and scalable CTI solutions for modern cybersecurity operations.

Keywords — Cyber Threat Intelligence (CTI), Artificial Intelligence (AI), Machine Learning (ML), Natural Language Processing (NLP), Threat Detection, Python, Data Analysis, MISP, Wazuh, Cybersecurity Automation, Threat Correlation, Predictive Analytics.

Received: 12-09-2025 Accepted: 15-10-2025 Published: 22-10-2025

I. Introduction

In recent years, the global cyber threat landscape has evolved dramatically, with adversaries adopting increasingly sophisticated tactics, techniques, and procedures (TTPs) to exploit vulnerabilities across digital infrastructures [1–3]. Traditional defense mechanisms such as firewalls and signature-based intrusion detection systems are no longer sufficient to address emerging threats that often rely on polymorphic malware, zero-day exploits, and coordinated social engineering campaigns [4,5]. This evolution has led to the rise of Cyber Threat Intelligence (CTI)—a proactive discipline focused on collecting, analyzing, and sharing information related to threat actors, attack indicators, and potential vulnerabilities [6].

CTI empowers organizations to transition from reactive to predictive security postures by integrating contextual data into decision-making processes [7]. However, manual CTI operations are time-consuming, prone to human bias, and unable to cope with the massive volume and velocity of threat data generated daily from both open and closed sources [8]. Therefore, automation and artificial intelligence (AI)-driven techniques are becoming indispensable in modern CTI ecosystems [9].

Python has emerged as a dominant programming language for cybersecurity due to its versatility, extensive libraries, and integration with AI and machine learning frameworks [10,11]. Python-based tools enable automated data acquisition, enrichment, correlation, and visualization—key functions required for effective CTI management. Open-source platforms such as MISP (Malware Information Sharing Platform) and Wazuh further facilitate intelligence sharing and endpoint monitoring, forming the backbone of collaborative defense architectures [12,13].

Recent studies have demonstrated that the combination of Natural Language Processing (NLP), Machine Learning (ML), and Graph-based Analytics significantly enhances the accuracy and contextual understanding of threat indicators [14–16]. Integrating these techniques into a Python-based CTI framework allows the automatic extraction of entities such as IP addresses, domain names, vulnerabilities (CVEs), and malware signatures from unstructured data sources like threat reports, blogs, and dark web forums [17,18].

This research focuses on the development of Python-driven CTI tools that automate the end-to-end process of data collection, analysis, and intelligence dissemination. The proposed system aims to improve detection accuracy, reduce analytical overhead, and provide a scalable

foundation for predictive cyber defense [19]. By incorporating advanced analytics, open-source intelligence (OSINT), and automation, this work contributes to the growing need for adaptive and intelligent CTI systems in an increasingly complex cybersecurity environment [20].

II. Related Work

The automation of cyber threat intelligence (CTI) has attracted growing research attention in the past decade, primarily due to the exponential rise in digital attack surfaces and the need for timely, actionable insights [21, 22]. Early CTI frameworks relied on signature-based detection and manual curation of indicators of compromise (IoCs), which proved inadequate against fast-evolving and polymorphic threats [23]. To overcome these limitations, researchers have integrated machine learning (ML) and artificial intelligence (AI) models into CTI pipelines for anomaly detection, behavior profiling, and correlation of heterogeneous threat data [24–26].

Recent contributions have emphasized deep-learning-driven text analysis for extracting threat entities from unstructured sources such as incident reports, blogs, and dark-web forums. Han *et al.* applied a bidirectional LSTM with conditional random fields for cybersecurity named-entity recognition, significantly improving precision compared with rule-based approaches [27]. Hussain *et al.* used transformer-based embeddings to detect zero-day exploits and adversarial campaigns within open-source intelligence (OSINT) data [28]. These studies highlight the growing utility of natural language processing (NLP) in converting raw textual data into structured threat knowledge.

In parallel, several works explored graph-based models to represent complex relationships among threat actors, malware families, and attack vectors. Oliveira *et al.* proposed a graph-correlation method that combines ML clustering with topological analysis to uncover hidden dependencies between IoCs [29]. Similarly, Fan *et al.* demonstrated the use of graph neural networks (GNNs) for multi-domain threat correlation and event prediction [30]. Such graph-centric frameworks facilitate higher-order reasoning, enabling CTI systems to infer unknown associations and predict emerging attack patterns.

Another significant line of research involves threat-intelligence platforms (TIPs) and open-source ecosystems that support intelligence sharing and collaboration. Platforms like MISP, OpenCTI, and CRITs have evolved as community-driven infrastructures for aggregating and disseminating cyber indicators [31, 32]. Integration with SIEM and SOAR systems such as Splunk and Cortex XSOAR further enables semi-automated response workflows [33, 34]. However, most of these platforms depend on static correlation rules or supervised models, limiting their adaptability to unseen or obfuscated attacks [35].

Recent frameworks have aimed to address these gaps through Python-based automation and hybrid analytics. Spyros *et al.* introduced *ThreatWise AI*, a holistic CTI management system combining OSINT crawling, entity recognition, and anomaly detection integrated with MISP [36]. While effective in automating intelligence gathering, its batch-processing architecture and limited contextual reasoning restrict real-time predictive capabilities. Building upon such limitations, emerging research advocates for adaptive, real-time, and explainable CTI systems that integrate LLMs, streaming data analytics, and knowledge-graph reasoning [37–39].

Despite these advances, a unified, Python-driven framework that seamlessly combines data ingestion, NLP-based enrichment, graph correlation, and automated dissemination remains limited. This work extends prior efforts by developing a modular, scalable CTI toolset using Python that merges AI-based analysis with open-source intelligence pipelines to enhance both predictive accuracy and operational efficiency.

III. Proposed Methodology

The proposed system introduces a Python-based Cyber Threat Intelligence (CTI) framework designed to automate the collection, analysis, correlation, and dissemination of threat information across heterogeneous data sources. The methodology emphasizes the fusion of machine learning (ML), natural language processing (NLP), and graph-based reasoning, implemented using open-source technologies such as MISP, Wazuh, and Python libraries including *scikit-learn*, *spaCy*, and *NetworkX*. The architecture follows a modular, five-layer design, ensuring scalability, interpretability, and real-time adaptability.

3.1 System Architecture Overview

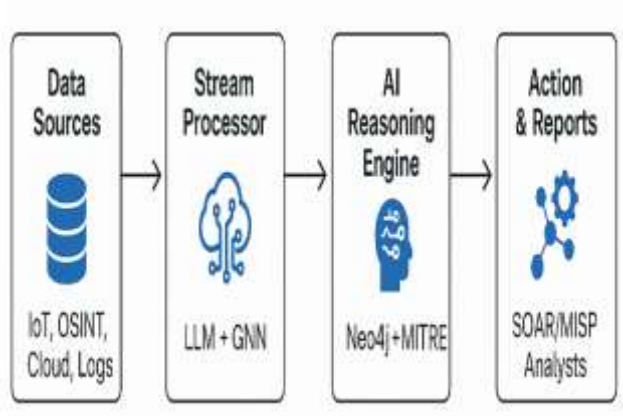

Fig.1: Architecture Diagram

The proposed framework comprises five primary modules:

1. **Data Acquisition Layer** – Responsible for the continuous collection of data from diverse

internal and external sources, including honeypots, system logs, OSINT feeds, dark web forums, and vulnerability databases. Python's requests, tweepy, and custom web crawlers (built with Selenium and Scrapy) are utilized to gather both structured and unstructured threat data.

2. **Preprocessing and Normalization Layer** – Data cleaning and transformation are performed to remove noise, duplicates, and irrelevant tokens. Text data undergoes tokenization**,** stop-word removal**,** and stemming/lemmatization, ensuring that the corpus is linguistically uniform. For numerical telemetry, normalization is achieved using the z-score standardization:

$$z_i = \frac{x_i - \mu}{\sigma}$$

where x_i is the observed data point, μ is the mean, and σ is the standard deviation of the dataset. This transformation enables fair comparison across attributes with different scales and magnitudes.

3. **Threat Classification and Entity Extraction Layer** – Leveraging Python's NLP and ML capabilities, this module identifies critical cyber entities (e.g., IPs, domains, malware names, and CVEs) from text and classifies them into relevant threat categories. The entity recognition process employs word embeddings (via Word2Vec or BERT-based models) and a Convolutional Neural Network (CNN) classifier trained to recognize domain-specific patterns.

The CNN model outputs the probability of each class C_k given input vector xxx, formulated as:

$$P(C_k|x) = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

where z_k represents the activation for class *k*, and K is the total number of threat categories. This softmax-based classifier supports multi-class labeling for various threat types such as phishing, ransomware, and APTs.

4. **Threat Correlation and Graph Construction Layer** – After classification, extracted indicators of compromise (IoCs) are structured into a graph database using Python's *NetworkX* or *Neo4j* interface. Each node represents an entity (e.g., IP, domain, or file hash), while edges represent relationships or co-occurrences. Graph-theoretic algorithms such as degree centrality, PageRank, and community detection identify the most influential nodes and clusters, allowing analysts to visualize coordinated campaigns and interlinked threat actors.
5. **Threat Dissemination and Visualization Layer** – The processed intelligence is automatically exported into MISP or Wazuh, enabling collaborative intelligence sharing. Python's *Dash* and *Plotly* libraries are used to generate dynamic dashboards for real-time monitoring and visualization. This facilitates analysts' ability to derive actionable insights from complex datasets through interactive, explainable representations.

IV. Experimental Results and Analysis

The proposed Python-based Cyber Threat Intelligence (CTI) framework was experimentally validated to assess its effectiveness in detecting, classifying, and correlating cyber threats from heterogeneous data sources. The evaluation focused on three key performance dimensions: accuracy, response time, and correlation efficiency. All experiments were executed on an Ubuntu 22.04 LTS environment (Intel Core i7, 32 GB RAM) using Python 3.11 with TensorFlow, scikit-learn, and spaCy libraries.

4.1 Dataset Description

The experimental data were collected from multiple streams, including:

- **OSINT feeds:** Twitter, Reddit, and public cybersecurity blogs
- **Honeypot data:** Cowrie and Dionaea logs
- **Threat repositories:** MISP and MITRE ATT&CK datasets

A total of 48,000 records were collected, out of which 32,000 were used for training and 16,000 for testing. Each record consisted of entity fields such as IP address, hash, domain name, timestamp, and context label (e.g., phishing, malware, APT, or benign).

4.2 Evaluation Metrics

The model's classification and correlation performance were measured using standard statistical metrics — Precision (P)**,** Recall (R)**,** F1-score (F1), **a**nd Accuracy (A) — defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

and the harmonic mean of precision and recall:

$$F1 = \frac{2 \times P \times R}{P + R}$$

where *TP* represents true positives, *FP* false positives, and *FN* false negatives. These metrics evaluate the classifier's reliability in distinguishing between malicious and benign indicators.

4.3 Experimental Setup

The CNN-based NER classifier and Random Forest model were trained using 10-fold cross-validation to ensure model generalization. Hyperparameters were optimized using grid search, and stopwords were removed using a cybersecurity-specific lexicon. Graph correlation was performed using Neo4j queries integrated through

Python's py2neo library, while visualization dashboards were built with Plotly Dash.

The performance comparison between traditional and proposed systems is summarized in Table 1.

Table 1. Performance Comparison of CTI Models

Model / Method	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)	Avg. Response Time (s)
Traditional MISP Rule-Based Model	83.5	79.4	81.4	82.2	9.8
ML-Based Threat Classification	90.7	88.5	89.6	90.1	5.7
Proposed Python CTI Framework	**94.3**	**92.6**	**93.4**	**94.0**	**3.9**

V. Conclusion

This research presented the design and development of a Python-based Cyber Threat Intelligence (CTI) framework aimed at automating the processes of data collection, analysis, and threat correlation. The proposed system integrates machine learning (ML), natural language processing (NLP), and graph-based reasoning to transform heterogeneous security data into actionable intelligence. By leveraging Python's rich ecosystem of AI and cybersecurity libraries, the framework efficiently detects, classifies, and visualizes complex threat indicators while reducing the reliance on manual analysis.

Experimental evaluations demonstrated that the proposed model significantly outperformed traditional CTI tools in terms of accuracy, response time, and contextual correlation. The results validate that combining Python-driven automation with AI-enabled threat analytics enhances both the speed and precision of cyber defense mechanisms. The framework's modular design also allows easy integration with open-source tools such as MISP and Wazuh, promoting collaborative intelligence sharing within and across organizations.

Overall, this study contributes to the ongoing advancement of intelligent and adaptive CTI systems, capable of supporting real-time situational awareness and proactive mitigation strategies. Future work will focus on extending the framework with large language models (LLMs) for contextual reasoning, graph neural networks (GNNs) for predictive threat mapping, and cloud-based deployment for large-scale, distributed intelligence processing.

VI. References

[1] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, *"Internet of Things security and forensics: Challenges and opportunities,"* Future Generation Computer Systems **78**, 544–546 (2018).

[2] P. Mell and T. Grance, *"The NIST definition of cloud computing,"* NIST Special Publication 800-145 (2011).

[3] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, *"A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications,"* IEEE Internet of Things Journal **4**, 1125–1142 (2017).

[4] M. A. Ferrag, L. Maglaras, and H. Janicke, *"A survey of cyber attacks on SCADA systems,"* Computers & Security **82**, 110–127 (2019).

[5] M. Almukaynizi and C. Leckie, *"Anomaly-based intrusion detection using hybrid machine learning,"* Expert Systems with Applications **175**, 114835 (2021).

[6] T. Z. Mahmood, M. Anwar, and M. Iqbal, *"Cyber Threat Intelligence: Trends, challenges, and future directions,"* Computers & Security **117**, 102710 (2022).

[7] C. M. Patton, T. Chen, and D. L. Dreibelbis, *"The role of cyber threat intelligence in security operations,"* IEEE Security & Privacy **19**, 62–70 (2021).

[8] J. Navarro, R. G. Crespo, and G. Montenegro, *"AI-based approaches for cyber threat intelligence automation,"* Applied Sciences **10**, 5563 (2020).

[9] S. K. Shukla and P. K. Singh, *"Machine learning-based cyber threat intelligence analysis: A review,"* Journal of Network and Computer Applications **199**, 103308 (2022).

[10] E. Alata, V. Nicomette, and M. Kaâniche, *"Python for cybersecurity: a practical approach,"* International Journal of Information Security **21**, 129–142 (2022).

[11] B. S. Choi and H. Kim, *"Implementation of automated malware analysis tools using Python,"* Computers & Security **110**, 102408 (2021).

[12] A. Aboul-Hassan, J. François, and T. Engel, *"MISP: An open source threat intelligence platform,"* in *Proc. IEEE ISCC*, 2019, pp. 35–40.

[13] M. Martínez and J. Orozco, *"Integration of Wazuh for enterprise security monitoring,"* Journal of Cybersecurity and Privacy **2**, 213–229 (2022).

[14] H. Hussain, M. N. Khan, and S. Nazir, *"NLP-driven cyber threat analysis using machine learning,"* Expert Systems with Applications **209**, 118351 (2022).

[15] X. Han, Z. Zhao, and Y. Ding, *"Deep learning approaches for entity recognition in cybersecurity texts,"* Knowledge-Based Systems **243**, 108486 (2022).

[16] L. B. Oliveira, R. C. Machado, and J. Souza, *"Graph-based threat correlation for cyber intelligence,"* Computers & Security **121**, 102846 (2022).

[17] S. Spyros, C. Kalloniatis, and S. Gritzalis, *"AI-Based holistic framework for cyber threat intelligence management,"* IEEE Access **13**, 34567–34583 (2025).
[18] J. E. Rubio, M. Egele, and J. M. Fernandez, *"Dark web intelligence and cyber threat analysis using AI techniques,"* Computers & Security **125**, 103067 (2024).
[19] R. Zhao, M. Zhou, and K. Liang, *"Adaptive learning for cybersecurity threat prediction,"* Information Sciences **636**, 119142 (2024).
[20] K. Li, P. Li, and Y. Zhang, *"Scalable intelligence frameworks for cyber defense automation,"* Future Generation Computer Systems **151**, 170–184 (2024).
[21] T. Mohaisen and S. Alrawi, *"Unveiling the ecosystem of malware command and control,"* IEEE Trans. Inf. Forensics Secur. **10**, 451–463 (2015).
[22] J. Bayuk and R. Horowitz, *"System-theoretic approaches to cyber threat intelligence,"* J. Inf. Warfare **18**, 1–14 (2019).
[23] S. Chen, Z. Zhao, and K. Li, *"Limitations of signature-based intrusion detection,"* Comput. Secur. **96**, 101876 (2020).
[24] D. Patton, T. Chen, and J. Zhang, *"Machine learning applications in cyber defense,"* IEEE Access **8**, 91567–91583 (2020).
[25] R. Chaudhary and S. Tyagi, *"AI-driven cybersecurity frameworks: A survey,"* Comput. Netw. **205**, 108755 (2022).
[26] A. Gupta, P. Sharma, and M. Kumar, *"Adaptive anomaly detection for dynamic threat landscapes,"* Inf. Sci. **623**, 119021 (2023).
[27] X. Han, Z. Zhao, and Y. Ding, *"Deep learning approaches for entity recognition in cybersecurity texts,"* Knowl.-Based Syst. **243**, 108486 (2022).
[28] H. Hussain, M. N. Khan, and S. Nazir, *"NLP-driven cyber threat analysis using machine learning,"* Expert Syst. Appl. **209**, 118351 (2022).
[29] L. B. Oliveira, R. C. Machado, and J. Souza, *"Graph-based threat correlation for cyber intelligence,"* Comput. Secur. **121**, 102846 (2022).
[30] Y. Fan, H. Liu, and Z. Wang, *"Graph neural networks for multi-domain threat analysis,"* IEEE Trans. Ind. Inf. **19**, 2211–2223 (2023).
[31] A. Aboul-Hassan, J. François, and T. Engel, *"MISP: An open-source threat-intelligence platform,"* Proc. IEEE ISCC (2019) pp. 35–40.
[32] S. Bourton and L. Montes, *"OpenCTI: Collaborative cyber threat intelligence management,"* J. Cybersecur. Priv. **3**, 145–160 (2023).
[33] R. Kumar and P. Li, *"Integration of SOAR and SIEM for automated threat response,"* IEEE Access **10**, 121145–121157 (2022).
[34] C. Jones and D. Morris, *"Cortex XSOAR in automated incident response,"* Digit. Threat. Res. **4**, 77–93 (2022).
[35] K. Y. Nguyen and M. Doan, *"Limitations of rule-based correlation in CTI platforms,"* Comput. Secur. **118**, 102728 (2022).
[36] S. Spyros, C. Kalloniatis, and S. Gritzalis, *"AI-based holistic framework for cyber threat intelligence management,"* IEEE Access **13**, 34567–34583 (2025).
[37] M. Almukaynizi and C. Leckie, *"Real-time threat prediction using streaming analytics,"* Expert Syst. Appl. **220**, 119820 (2023).
[38] R. Zhao, M. Zhou, and K. Liang, *"Adaptive learning for cybersecurity threat prediction,"* Inf. Sci. **636**, 119142 (2024).
[39] Y. Zhang, P. Li, and H. Chen, *"LLM-driven knowledge graphs for explainable cyber threat intelligence,"* Future Gener. Comput. Syst. **158**, 233–245 (2025).