CyberHolmes: Interactive Cyber Threat Intelligence Monitoring System

Muhammad Ariff Ridzlan Mohd Faudzi1, a), Shih Yin Ooi1, 2, b), Evita Herawaty Othman3, c), Ying Han Pang1, 2, d) and Yee Jian Chew1, 2, e)

*1Faculty of Information Science and Technology (FIST), Multimedia University, Jalan Ayer Keroh Lama, Melaka, 75450, Malaysia*

*2Centre for Advanced Analytics (CAA), COE for Artificial Intelligence, Multimedia University, Jalan Ayer Keroh Lama, Melaka, 75450, Malaysia*

*3Cybersecurity Risk Management (CRM) Services, TM ONE CYDEC, TM Annexe 2, Jalan Pantai Jaya, 59200 Kuala Lumpur, Malaysia*

*b) Corresponding author: syooi@mmu.edu.my*

*a) 1221301131@student.mmu.edu.my*

*c)evita.othman@tm.com.my*

*d)yhpang@mmu.edu.my*

*e)chewyeejian@mmu.edu.my*

**Abstract.** Today’s cybersecurity landscape demands robust cyber threat intelligence (CTI) systems to identify, analyze, and address emerging threats promptly. This paper introduces CyberHolmes, a CTI platform designed to simplify the entire intelligence process, from data collection to analysis of key information. Unlike conventional systems, CyberHolmes focuses on managing the full CTI workflow by planning, gathering, and analyzing threat data from sources such as normal websites and dark web forums. The platform also integrates with key frameworks, such as CVSS for scoring vulnerabilities and MITRE ATT&CK for mapping attacker techniques through cosine similarity calculations on SecBERT embeddings. Although CyberHolmes utilizes tools such as web crawling and AI, its primary objective is to generate actionable intelligence that enables security teams to respond to threats more effectively and efficiently. Results show that CyberHolmes successfully produces meaningful insight for threat intelligence, making it a strong solution for improving an organization’s cybersecurity.

# INTRODUCTION

In the current cybersecurity environment, individuals and organizations must find new ways to protect themselves against emerging threats. The rising incidences and complexity of cyberattacks, which constantly breach traditional security measures, have led to a rise in threats. Intelligent systems with rapid threat detection and defensive capabilities are essential today, especially with the rising cyberattack cases. The recent development of artificial intelligence (AI) enables the processing and interpretation of large amounts of data at speeds far exceeding human capabilities, making it beneficial to recognize anomalies and patterns quickly.

The current conventional threat hunting system faces a number of limitations, including inefficient data samples, unstructured data analysis, and lack of contextual information and actionable insights[1]. Due to the fact that data acquisition is tedious, slow, and error-prone, most of the threat hunting system facing challenges to handle large volumes of data effectively. Thus, the primary objectives of the CyberHolmes system are to automate the collection of cyber threat data through advanced web crawling. It also integrated with several AI techniques, including pattern recognition, natural language processing, and sentiment analysis to detect and analyze the threat automatically.

This paper is presenting and discussing the entire deployment of CyberHolmes. It is a dashboard and system integrate and incorporate several AI approaches throughout the entire cyber threat intelligence (CTI) pipeline. The result shown that by combining web crawling, data processing, and machine learning models into a single system, it is able to detect threat faster and more accurately.

# LITERATURE REVIEW

## Cyber Threat Intelligence

The process of gathering, evaluating, and classifying data from multiple online sources in order to detect possible cybersecurity risks is known as threat intelligence (TI). TI assists organizations in comprehending the dangers of external attacks, including zero-day vulnerabilities and advanced persistent threats. Additionally, it offers information about the potential attacker's identity, objectives, abilities, and indicators of a breach, enabling organizations to make better decisions about their own security.

### Components and Features of CTI

As cyber threats continue to grow quickly, having up-to-date information is important for organizations to protect themselves. This information can be gathered from OSINT sources, such as social media, news articles, blogs, and dark web forums, where threat actors often communicate [2]. CTI platforms utilize techniques such as web crawling and web scraping to collect large amounts of data automatically, thereby reducing the need for manual labor and accelerating the process. Thus, it can be said that web crawling and scraping are automated methods for extracting data from websites into formats such as HTML, JSON, and XML [3]. Many tools available today can collect information from both the dark and clear webs. One study, for instance, used Python libraries like PyMongo to store data in a MongoDB database, Requests to send HTTP requests, and Scrapy to crawl [4]. Additionally, they created a customized crawler for the Tor network that supports CTI efforts by allowing the tracking of illicit activity on the dark web using word clouds and histograms.

Threat analysis is necessary for organizations to be able to see patterns, look at unusual activity, and see suspicious alarms. One of the assignments is known as threat analysis, wherein you take that raw data and turn it into something useful. CTI platforms are able to associate, examine, and forecast threats in real-time with sophisticated machine learning (ML) and artificial intelligence (AI) algorithms [2]. AI on these sites enables quicker, better-informed decisions in cyber security, higher accuracy and an even keener edge in detecting emerging threats. In [5], the authors built a machine learning model to classify and detect threats on hacker forums from a set of 150,000 posts. They compared six various ML classifiers such as, Support Vector Machine (SVM), k-Nearest Neighbors (KNN) and Random Forests. They used NLP algorithm for tonality classification and topic classification. Once the data had been cleaned and was in the tokenized and normalized form, the models were tested. Most effective results were delivered by Random Forest model that accompanied 0.89 accuracy of 0.91 precision, and F1-score of 0.87. AI technology within the CTI platform helps such systems to detect sneaky threats sooner, minimize false alarms, and resolve issues faster. A study demonstrated the way inTIME utilized NLP methods like dependency parsing, phrase matching, and Named Entity Recognition (NER) to retrieve security information more effectively [6]. To enhance threat detection and make CTI gathering more context-related, it also interacted with knowledge bases like MITRE ATT&CK and utilized semantic ranking to emphasize crucial intel. This translated into enhanced cybersecurity defense.

Lastly, all of the features of CTI are presented in a manner that enables it to be easily read and distributed by security professionals. Visual aids such as threat maps, graphs, and charts are used by the dashboard to make complicated data simple. Users can also set filters to customize their view so they are only seeing the most critical and latest threats. Real-time updates are also crucial; these allow analysts to track and respond to emerging threats in a timely fashion. A CTI dashboard typically shows summaries of cyber threats, data about the threat actors, and facts about their attacks that matter to organizations and industries. It also provides navigation controls for viewing suspicious IPs, stolen information, leaks, and recent attacks. Overall, the insights dashboard is important because it presents information, allows security teams to make faster and improved decisions, and improves response precision.

### Existing CTI Platforms

In total, five CTI platforms were analyzed to compare their key features. These platforms are ThreatConnect, Anomali ThreatStream, Mandiant Advantage, IBM X-Force Exchange, and Rapid7 Threat Command. The comparison focused on key capabilities, including web crawling, OSINT feeds, dark web monitoring, AI utilization, real-time alerts, customizable rules, and integration with other security tools. While all platforms support most of these features, there are some differences. For example, some platforms have limited web crawling capabilities or rule customization options. Overall, each platform provides robust threat intelligence support, offering various levels of integration and automation. The detailed comparison is shown in Table 1.

**TABLE 1.** Comparison table of CTI features and capabilities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CTI Platforms** | **Web Crawling & Scraping** | **OSINT Feeds** | **Dark Web Monitoring** | **AI** |
| ThreatConnect | Yes | Yes | Limited | Yes |
| Anomali ThreatStream | No | Yes | Yes | Yes |
| Mandiant Advantage | Limited | Yes | Yes | Yes |
| IBM X-Force Exchange | Yes | Yes | Yes | Yes |
| Rapid7 Threat Command | Yes | Yes | Yes | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| **CTI Platforms** | **Real-Time Alerts** | **Customizable Rules & Filters** | **Integration** |
| ThreatConnect | Yes | Yes | SIEM, SOAR, EDR, custom APIs |
| Anomali ThreatStream | Yes | Yes | SIEM, SOAR, EDR, TI platforms |
| Mandiant Advantage | Yes | Limited | EDR, XDR, and Google Cloud security tools |
| IBM X-Force Exchange | Yes | Yes | IBM security tools like SIEM and SOAR |
| Rapid7 Threat Command | Yes | Yes | SIEM, IDR, endpoint security tools |

# METHODOLOGY

For the development of this project, the iterative model was chosen as the software development approach. This model was suitable because it allows the system to be built step by step, starting with the basic features and improving it through several cycles [7]. It is useful for projects that may need changes or small updates during development. Each phase helps to break down complex tasks into smaller parts, making them easier to manage. The iterative model also helps reduce risks early and provides an opportunity to review and improve the system in every cycle. This way, the final product meets the project requirements and works as expected.

## Development Stages of CyberHolmes

The iterative model for developing CyberHolmes followed a clear flow through each phase. The planning phase included creating a detailed project plan and milestones. This phase also involved defining the project's goals, objectives, and scope. Preliminary research was conducted to gather requirements for a cyber threat intelligence platform. During the analysis phase, we identified key features such as data collection, threat analysis, actionable intelligence, and dashboard functionality. To ensure proper tracking, we created a requirements management plan.

In the design phase, we used data flow diagrams, database schemas, and wireframes to plan the system structure and user interface. We selected the best designs for implementation. The implementation and coding phase included developing elements like web crawling for data collection, AI models for threat detection and analysis, and an interactive dashboard to present results.

During the testing phase, the system prototype went through several test cases and usability tests. We resolved issues to improve functionality and ensure readiness for deployment.

In the end, this development aims to improve the cyber threat intelligence (CTI) process. It shows how we can combine existing technologies into a workable system that supports timely and effective decisions in cybersecurity operations.

## Ethical Considerations of Dark Web Crawling

For this project, only publicly accessible .onion websites that anyone can visit without paying or logging in were used to gather data. These websites were accessed through the Tor network, and there was no effort to track or identify users. Every piece of information collected was securely stored in a database and managed according to ethical and legal guidelines. No systems were compromised, and no private data was stolen for improper use or personal gain. These steps were taken to support threat intelligence research while ensuring that the data collection was ethical and responsible.

# SYSTEM DESIGN

## Architecture Overview

CyberHolmes (Figure 1) is a cyber threat intelligence system that utilizes modern web tools to collect and analyze online data, including data from the dark web. The front end is built with Next.js and TypeScript, where Next.js helps the website load faster, and TypeScript helps catch errors early, making the code more reliable. The system will provide key information, including the number of posts collected, threat severity from both the clear web and dark web, and trends, using dashboards, charts, and graphs. Meanwhile, the backend utilizes Django, a secure and scalable Python framework, along with Django REST Framework (DRF) to create APIs that enable seamless integration between the frontend and backend, as well as manage and handle data collection and AI prediction processes. CyberHolmes utilizes Scrapy to develop web crawlers and scrapers that extract data from both the clear web and the dark web, leveraging the Tor network. After collecting the data, machine learning with scikit-learn checks, whether the content poses a threat, and deep learning with PyTorch and models from Hugging Face helps to analyze the information from posts further. All processed data is saved in an SQLite database, which is simple and useful for development.

A diagram of a software

AI-generated content may be incorrect.

**FIGURE 1.** A high-level overview of the CyberHolmes system architecture

# IMPLEMENTATION

## Web Crawling and Scraping

The web crawling and scraping process, as illustrated in Figure 2, begins by selecting a target website and specifying a keyword to search for. The keyword is added to a search URL to begin the crawling process. After that, the system checks whether the website belongs to a different domain, such as a .onion site, which requires extra privacy when accessed. If it does, the system will use Tor to create a SOCKS5 proxy for anonymous browsing. This setup utilizes tools such as the Tor service and the request\_tor library to send requests securely. For normal websites, the system uses Python’s standard requests library and functionality without any special proxy settings. Once the connection is ready, the system sends a request to the search URL and downloads the page content. It utilizes Scrapy’s selectors to find all the post links related to the keyword from the generated result. These links are then compiled into a list, and the crawler visits each one to obtain more information. If the result page has multiple parts or pagination, the system will detect it and continue to the next page until all posts are collected.

Scrapy handles most of this work, like avoiding duplicate links and scheduling which pages to crawl and scrape next. After gathering all the necessary pages, the system begins extracting the actual data, including text, based on CSS or XPath rules. Scrapy’s item pipelines help to clean and save the data into a database. Throughout the process, the system switches between Tor and normal Internet based on the website type. In this way, it maintains a flexible, secure, and efficient workflow for both clear web and dark web sources.

A diagram of a flowchart

AI-generated content may be incorrect.

**FIGURE 2.** Flowchart illustrating the process of web crawling and scraping

## Artificial Intelligence

### Threat Prediction Using Machine Learning Model

Referring to the pipeline illustrated in Figure 3, the dataset for this project was collected from five different cybersecurity forums and dark web sources, including CrackingArena, Twitter (X), DreamMarket, Garage4hackers, as well as CrackingFire [8]. It consists of 9,470 posts in total, with 882 labeled as threats and 8,359 as non-threats, creating a clear class imbalance. The posts vary in length, typically ranging from 13 to 169 words. Each post is labeled as either “YES”, indicating that it is a threat, or “NO” for a normal one, while posts labeled “Undecided” were removed. By utilizing data from both the clear and dark web, the model can generalize more effectively across various types of online discussions.

A diagram of a process

AI-generated content may be incorrect.

**FIGURE 3.** Machine learning pipeline

Before training the model, the text data was cleaned and pre-processed. This included removing HTML tags, URLs, mentions, and special characters, expanding contractions like “don’t” to “do not,” and replacing numbers with a <NUM> token. Tokenization and part-of-speech (POS) tagging were done using the NLTK library, and important cybersecurity terms were kept even when removing common stop words. Lemmatization was used to reduce words to their root form based on their part of speech. To address the class imbalance, SMOTE Tomek was used, which oversampled the threat data and undersampled non-threat data, thereby improving the model’s performance on both classes. TF-IDF with n-grams was applied to turn text into numerical features. Hyperparameters were tuned using a grid search, and stratified 5-fold cross-validation was employed to ensure that both classes were well-represented during evaluation.

For model development, three types of machine learning models were trained: Naïve Bayes, serving as a baseline for performance, Support Vector Machine (SVM), and an ensemble model. The ensemble combined Logistic Regression, Stochastic Gradient Descent (SGD) Classifier, and XGBoost, using soft voting, where predictions from all models were averaged based on their probabilities.

### Threat Analysis Using Pre-Trained Deep Learning Model

In this stage, data that had previously been recognized as a threat was analyzed using a pre-trained deep learning model known as SecBERT. A variant of the BERT model, SecBERT is trained on cybersecurity-specific texts to improve its comprehension of terms like "phishing" and "malware." Hugging Face's transformers library provides easy access to the PyTorch-implemented model. With an emphasis on cybersecurity terms, SecBERT tokenizes the input sentence and processes it to produce a fixed-size numeric vector that captures the sentence's meaning.

()

Regarding the Common Vulnerability Scoring System (CVSS) v4 scoring, SecBERT embeddings are utilized to match parts of a vulnerability description with the corresponding CVSS metrics. By comparing the input embedding to reference embeddings using cosine similarity, as shown in Equation (1), the most similar metric values are selected. After that, the final CVSS v4 base score, along with its severity and vector string, is then generated using the Python cvss library. Cosine similarity measures the degree of similarity between two vectors by comparing the angle between them, with values closer to 1 indicating greater similarity. Similarly, SecBERT can help identify relevant MITRE ATT&CK techniques by comparing the vulnerability description’s embedding to stored technique embeddings using cosine similarity. The threat posts and MITRE ATT&CK technique descriptions are converted into vectors using SecBERT, and cosine similarity is used to determine which techniques best match the posts. The most similar techniques are then ranked. For information, the MITRE ATT&CK technique descriptions are retrieved using the Python attackcti library.

# RESULTS

## Artificial Intelligence

Table 2 compares three machine learning models: Naïve Bayes, SVM, and an ensemble model trained on a class-imbalanced dataset derived from cybersecurity forums and dark web sources. The ensemble model performed the best, achieving 93.78% accuracy, 81.37% precision, 84.36% recall, and an 82.77% F1-score. The ensemble model employed soft voting by combining Logistic Regression, SGD, and XGBoost, which helped it outperform the other models. SVM performed better than Naïve Bayes in all areas, particularly with a recall of 85.50% compared to 77.12%, which is crucial for detecting unique threats.

**TABLE 2.** Performance evaluation metrics on test dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ML Models** | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **Specificity (%)** | **F1-score (%)** |
| Naïve Bayes | 90.81 | 73.60 | 77.12 | 94.02 | 75.19 |
| Support Vector Machine | 92.16 | 76.97 | **85.50** | 93.72 | 80.40 |
| Ensemble | **93.78** | **81.37** | 84.36 | **96.00** | **82.77** |

Based on the results (see Figure 4), the CyberHolmes system is functioning as expected for both the front end and back end. The backend successfully performs web crawling [9][10] and scraping by connecting to both clear web and dark web sources using Scrapy and Tor. It collects posts based on keywords, handles pagination, and stores relevant information in a database. AI models trained on cybersecurity data are utilized to classify threats and perform tasks such as sentiment detection, keyword extraction, and entity recognition. The front end then displays this data through charts such as “Posts Scraped Over Time,” “Threat Posts by Source,” and “Severity Distribution,” making it easy to track trends. For example, the system recently scraped over a thousand posts, identified 227 threats, and detected approximately 300 posts with negative sentiment, confirming that web crawling, scraping, and threat classification are working effectively.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

|  |  |
| --- | --- |
| (a) | (b) |

**FIGURE 4.** (a) Dashboard page and (b) Analysis page of CyberHolmes

On the “Analysis” page, the system’s effectiveness is further confirmed. The backend uses SecBERT embeddings to analyze detected threats by assigning CVSS scores and matching potential MITRE ATT&CK techniques. It also evaluates sentiment and captures source information, like user profiles and content snippets. The front end displays all this information, including threat severity, confidence levels, CVSS details, sentiment scores, and links to the original posts. For example, the system identified a high-severity threat related to a Russian bulletproof hosting service, assigned a CVSS score of 7.7, calculated threat confidence of 75.8%, and traced it back to a Reddit user. This indicates that the deeper AI analysis, along with the features on the “Analysis” page, is functioning properly and that SecBERT, CVSS scoring and MITRE ATT&CK mapping are working as expected.

# CONCLUSION

In conclusion, CyberHolmes can help identify and respond to threats. It covers all stages of the CTI process, from collecting data to analyzing it and sharing useful information, turning raw data into clear, actionable insights. By utilizing frameworks such as MITRE ATT&CK and CVSS, CyberHolmes enables security experts to understand attack methods better and evaluate vulnerabilities. With AI and web crawling processes, CyberHolmes distinguishes itself by adhering to CTI best practices and enhancing data to make it more useful for informed decision-making. In the future, it is hoped that CyberHolmes could integrate with other cybersecurity tools and improve intelligence sharing. As cyber threats continue to grow, CyberHolmes provides a solid foundation for organizations to build a smarter, more proactive cybersecurity defense.

# References

1. “4 Key Challenges and Solutions in Threat Intelligence | CloudSEK,” (n.d.).
2. “What are the Key Components of Threat Intelligence? | CloudSEK,” (n.d.).
3. M. Khder, “Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application,” IJASCA **13**(3), 145–168 (2021).
4. Y. Xu, G. Chen, J. Wu, W. Xu, and Q. Liu, “Research on Dark Web Monitoring Crawler Based on TOR,” in *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, (IEEE, Chongqing, China, 2021), pp. 197–202.
5. S. Mambetov, I. Ilhe, V. Babenko, B. Kulambayev, O. Fridman, S. Joldasbayev, H. Doroshenko, O. Gurko, Y. Begimbayeva, and S. Neronov, “Detection and classification of threats and vulnerabilities on hacker forums based on machine learning,” EEJET **3**(9 (129)), 16–27 (2024).
6. P. Koloveas, T. Chantzios, S. Alevizopoulou, S. Skiadopoulos , and C. Tryfonopoulos , “inTIME: A Machine Learning-Based Framework for Gathering and Leveraging Web Data to Cyber-Threat Intelligence,” Electronics **10**(7), 818 (2021).
7. K. Rana, “Iterative Model - Features, Advantages & Disadvantages,” ArtOfTesting, (2020).
8. Andrei, “andlq/dataset\_hacker\_online\_communication,” (2021).
9. L.C. Xiang, O.S. Yin, and P.Y. Han, “Intelligent web crawler for file safety inspection,” in *2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, (IEEE, Kuala Lumpur, Malaysia, 2015), pp. 309–314.
10. Y.H. Tay, S.Y. Ooi, Y.H. Pang, Y.H. Gan, and S.L. Lew, “Ensuring Privacy and Security on Banking Websites in Malaysia: A Cookies Scanner Solution,” *Journal of Informatics and Web Engineering* **2**(2), 153–167 (2023).