Automated Web Scraping Tool for Real-Time Detection and Reporting of Critical and High-Severity OEM Vulnerabilities in IT and OT Systems

Augustian P B
Department of CSE, SoE
Presidency University
Bengaluru, Karnataka, India
akkuaugustian@gmail.com

Chandrashekhar S
Department of CSE, SoE,
Presidency university
Bengaluru, Karnataka, India
druvachandra9778@gmail.com

Shesha Venkat Gopal K
Department of CSE, SoE,
Presidency University,
Bengaluru, Karnataka, India
sheshavenkat.k@gmail.com

Shubha K A
Department of CSE, SoE
Presidency University
Bengaluru, Karnataka, India
shubhaambareesh@gmail.com

Dr. Nihar Ranjan Nayak
Assistant Professor, SoCSE
Presidency University
Bengaluru, Karnataka, India
niharranjan.nayak@presidencyuniversity.in

Abstract*—Due to the fact that maintaining operational security and preventing disruptions in the critical sector requires timely identification and remediation of vulnerabilities in IT and OT systems. If these hardware, software and firmware vulnerabilities are exploited, they are of serious risk, including critical and high severity. Today, many mechanisms, such as the National Vulnerability Database (NVD), lack timely reporting and expose organizations to new threats. In order to fill in this gap, we develop an automated web scraper that runs and monitors Original Equipment Manufacturer (OEM) Websites and trusted platforms in real time for finding new disclosed vulnerabilities. It then extracts the key details, such as product information, severity level, vulnerability description, mitigation strategy and publication date and disseminates them immediately to certain pre determined stake holders by emailing among others. The solution is driven by open source technologies that allow it to be easily adapted to support all manners of OEM data formats and syntaxes to achieve comprehensive coverage. Through the direct sourcing of data directly from vendors as well as authoritative sources, it reduces the latency associated with other vulnerability databases, allowing the critical sector organizations to quickly patch and bring down the threats. Extracted data is organized into structured reporting mechanism in the standardized fields (e.g. Product name, OEM, CVE_ID, Severity, Mitigation), so that it can be easily emailed to others. Secure authentication is provided by OAuth2 and notification are ensured to be reliable and encrypted. This tool enables automating vulnerability detection and reporting and improving security posture of critical infrastructure, reducing the exposure and exploitation risks, and enhancing the resilience for IT/OT environments.

Keywords—Vulnerability Monitoring, IT/OT Security, Web Scraping, Automated Threat Intelligence, Cybersecurity, Real-Time Detection, OEM Vulnerabilities, OAuth2 Authentication.

I. INTRODUCTION

A. Background and Motivation

Based on the resilience of critical sectors such as energy, health care, manufacturing, and finance, it's critical to assure security of Information Technology (IT) and Operational Technology (OT) systems. Due to the increasing integration of OT environments such as Industrial Control Systems (ICS) and Supervisory Control and Data Acquisition (SCADA) systems into IT networks, they are becoming increasingly exposed to attacks that are sophisticated. As stated by out guidelines, the National Institute of standards and Technology (NIST), unpatched ICS environments can result in operational interruptions, data breaches and financial losses [1].

Still, research numbers that organizations take on average 60 to 90 days to patch known security flaws [2]. A delay of mitigation is precisely what happened with the Log4Shell vulnerability [3]. The existing vulnerability management frameworks such as the National Vulnerability Database (NVD) and its cousin Common Vulnerabilities and Exposures (CVE) system are used to track security flaws. Nevertheless, previous studies have demonstrated that reporting delay is extensive on these platforms, exposing organizations for significant periods [4].

Typically, Original Equipment Manufacturers (OEMs) release security advisories before the vulnerabilities enter public databases. But since manual tracking of security updates across various OEM websites and security forums is inefficient, resource intensive and prone to human error, it is performed by hand. To overcome this problem, an automated real time vulnerability monitoring system is required to extract, structure and disseminate security advisories to the concerned stakeholders.

B. Problem Statement

Although there are vulnerability databases such as NVD and CVE, delaying the reporting of newly discovered security flaws represent great risks to IT and OT environments. As the Cybersecurity and Infrastructure Security Agency (CISA) have reported, traditional vulnerability disclosure mechanisms provide little visibility into new threats, and it may take weeks before an issue is officially documented [3]. Therefore, organizations are prone to zero-day vulnerabilities and long remediation cycles [5].

The most important challenge is not allowing automation for vulnerability tracking. OEM advisories, industry reports, security bulletins have to be reviewed manually by security teams to identify and assess vulnerabilities. The inefficient part is based on this manual process and there is a high probability of oversight and this increases the risk of exploitation [6].

To mitigate these issues, an automated system is required to:

- Continuously scan and pick up OEM website, trusted security platform and industry forums security advisories and install them.
- Reduces vulnerability awareness delay to real time, informing security teams of the same.
- To make sure the extracted vulnerability information is correct, structured, and secure to allow for timely mitigating tactics.

C. Objectives

The primary objective of this research is to develop an automated web scraping tool that enhances vulnerability detection and reporting. The tool will:

1. Automate vulnerability discovery from OEM sources, security advisories, trusted industry platforms.
2. Generate real time alerts to stakeholders for timely action to avoid risk.
3. Ensure the security and structural reporting by the conducive design of the system with a consecrated vulnerability classification system as demonstrated in [10].
4. Security: OAuth2 is used for secure authentications, such that all alerts and notifications are delivered with confidentiality and encryption [13].
5. Fewer reliance on the delayed vulnerability databases such as the NVD and CVE.

D. Contributions

The contributions of this research are the development of an automated web scraping tool to monitor and extract security advisories from OEM security portals, industry databases, as well as trusted cybersecurity platforms [7]. Through decreased dependency on the traditional techniques of vulnerability tracking (National Vulnerability Database (NVD) and Common Vulnerability and Exposures (CVE) system [1]), this tool provides real time vulnerability identification and dissemination resulting in significant reduction in the time period involved in traditional reporting.

Moreover, this study also introduces a structured and secure vulnerability reporting mechanism against integrating machine learning based classification to classify vulnerabilities severity and impact (reputation) [5]. Natural Language Processing (NLP) techniques are utilized by the tool to categorize vulnerabilities quickly and provide the stakeholders with the insights that matter for remediation. Moreover, the secure authentication and notification schemes based on OAuth2 and end-to-end encrypted channels [13] are incorporated to ensure the confidentiality and integrity of vulnerability alerts sent to the concerned security teams.

The research also deals with ethical and legal attempts of web scraping by implementing methodologies which are in accordance with cybersecurity laws and best practices [9]. This guarantees data collection from the subject in accordance with regulatory requirements and at the same time in accordance with ethical standards in data collection. A solution that is proposed attempts to improve cybersecurity resilience of IT and OT landscapes through a scalable and automated vulnerability detection process, thereby reducing vulnerability response time related risks as well as organizational security posture [2].

II. Related Work

A. Existing Vulnerability Tracking Systems

Vulnerability tracking is an integral part of cybersecurity where organizations can identify discovered defects to avoid exploitation. National Vulnerability Database (NVD) is a centralized repository of publicly disclosed information about cybersecurity vulnerability. NVD is a service provided by the NIST to expose vulnerability data in a structured fashion including severity ratings, description, and mitigation recommendations [1]. NVD is widely used, however delays between the initial disclosure and going public prevent it from being an effective tool for real time threat response.

Vendor-provided security advisories, such as security updates, patches and mitigation guidance published by vendor are another critical source of vulnerability information for vulnerable products. Siemens, Cisco and Microsoft like other prominent technology companies maintain dedicated security advisory portal by which they inform the users about newly discovered threats [7]. The advisories are authoritative sources that are not automatic to monitor, and organizations often have to manually monitor to determine which vendors are being affected.

B. Limitations of Current Solutions

However, current vulnerability tracking mechanisms have deficiencies in terms of their significance. NVD reporting delays can expose organizations to threats prior to the entry of the same into the NVD, which is currently officially cataloged and scored [10]. However, time lags between

vulnerability disclosure and NVD listing are known to vary quite significantly, from weeks to months [14]. This delay is a serious issue for organizations that rely on NVD for risk management and patch prioritization.

Furthermore, there is no real time monitoring of multiple OEMs. As vendor advisories are widely spread on different platforms, it becomes an a massive and unproductive job for the cybersecurity teams to manually keep track of each and every source or platforms. In addition, the reporting formats and the classification methodologies are inconsistent, making automated processing and insertion into security workflows harder.

TABLE I. LIMITATIONS OF CURRENT VULNERABILITY TRACKING MECHANISMS

Limitation	Impact on Security
Delays in NVD reporting	Organizations remain vulnerable until CVEs are listed. Attackers can exploit the delay before official recognition.
No real-time monitoring across OEMs	Security teams must manually track vendor advisories, leading to inefficiencies and potential missed vulnerabilities.
Inconsistent data formats	Vendors use different reporting structures, making automation difficult.
Scalability issues	Large organizations managing multiple products struggle with fragmented vulnerability sources.

Table 1 highlights the key limitations of existing vulnerability tracking mechanisms, particularly the National Vulnerability Database (NVD) and vendor-provided security advisories. The primary issue with NVD is the delay in reporting vulnerabilities, as it relies on vendor disclosures and validation processes before listing a CVE. This delay leaves organizations exposed to potential exploits before they can take necessary security measures.

The other major limitation is a lack of real-time monitoring across various OEMs. Since there are security advisories distributed through various vendor websites, organizations must manually search and analyze these updates, which is time consuming and susceptible to human errors. Moreover, vulnerabilities whose data formats—such as CVE structure, risk assessments, and patch details—vary across data sources leads to manual case work to track vulnerabilities and integration of various sources into a single security anchoring.

Enterprises with a large number of IT and OT to control have another concern, which is that it is not scalable. The lack of a centralized system to track real time vulnerability makes for a complicated and large amount of fragmented data that security teams have a hard time processing and potentially creating blind spots in risk management. To solve these limitations, the need is for an automated real time solution that is able to aggregate and analyze vulnerability data in a timely manner.

C. *Web Scraping in Cybersecurity*

Cybersecurity research has seen significant uptake of using automation to do threat intelligence. To collect security-related data, automated web scraping has been used for threat intelligence feeds, leaked credentials, vulnerability disclosures, etc. [9]. Web scraping helps the organization systematically extract and aggregate security information from multiple sources which helps in faster threat detection and response.

Several works on data collection in vulnerability management have been discussed. For instance, as textual descriptions and severity scores of vulnerabilities in SCV, it was leveraged to classify and prioritize vulnerabilities using machine learning models [5]. Meanwhile, natural language processing (NLP) frameworks further reveal such structured insights from unstructured security advisories [11]. Such advancements bring to light the power of web scraping and AI-powered analysis to facilitate automation of the collection and meaning of vulnerability data in the cybersecurity operations area.

III. PROPOSED SYSTEM

A. *System Architecture*

The proposed system is designed to automate vulnerability tracking by continuously monitoring multiple security sources, extracting relevant data, and delivering structured reports to stakeholders. The architecture consists of three core components:

- **Data Scraper:** This component takes security advisories and vulnerability details from OEM security portals, cybersecurity databases, trusted threat intelligence platforms. This uses web scraping techniques to find out the latest security updates efficiently.
- **Database:** All the collected data is stored in a centralized database where all the information regarding vulnerabilities can be stored in an organized manner. By structuring the storage of the historical data through the structure database, retrieving it efficiently, and maintaining its secure access, we thus make certain that the data does not lose its table structures to the degree that would hinder retrieval and pose a security problem.
- **Notification System:** It generates and sends automated email using the alerts to the relevant security teams and stakeholders. The vulnerability reports that are noticed in these notifications include levels of severity and recommendations.

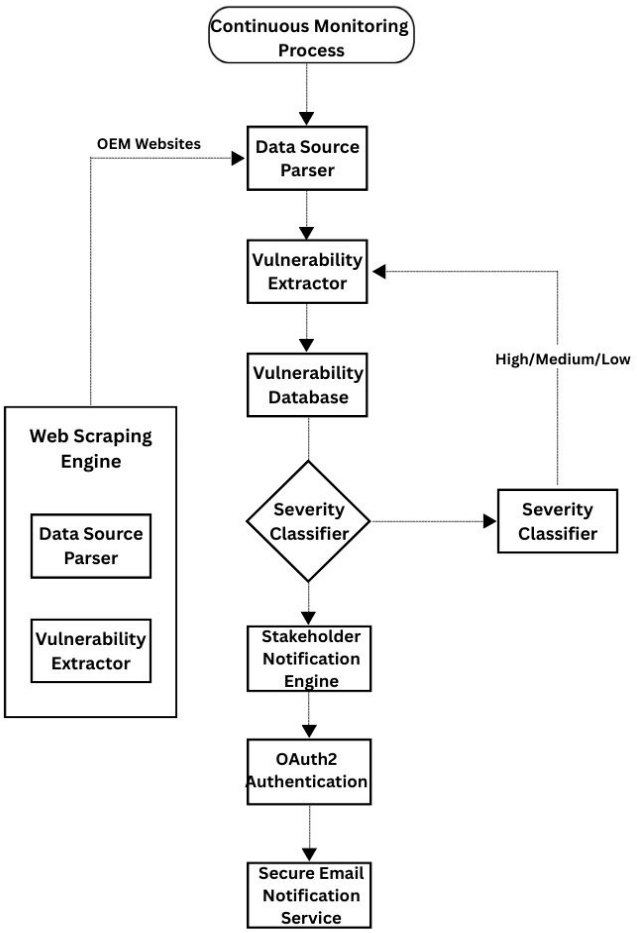

Fig. 1. System Architecture for Automated Vulnerability Detection and Notification

Fig 1 illustrates the architecture of the proposed automated vulnerability detection and reporting system. The system comprises several key components, starting with the **Web Scraping Engine**, which gathers security-related data from **OEM websites** and **security platforms**. This data is processed by the **Vulnerability Extractor**, which identifies and extracts relevant security vulnerabilities.

Then, when extracted, it is stored in the Vulnerability Database to enable management in structured manner and easy retrieval. After extracting vulnerabilities, the Severity Classifier scores them according to criticality levels following some scoring mechanism established as Common Vulnerability Scoring System (CVSS). The Stakeholder Notification Engine is triggered when a vulnerability rated Critical or High. The Logging Module receives the lower severity vulnerabilities (Medium and Low) for the record keeping.

The notification system is done through OAuth2 Authentication so that emails are communicated in a secure manner. The Secure Email Notification Service distributes the alerts and delivers to stakeholders in a timely, structured form, providing reports on the identified vulnerabilities. Besides, a Continuous Monitoring Process is used to monitor the system actively detecting new vulnerabilities and keeping in real time about possible security threats.

In particular, this architecture improves the resilience of the cybersecurity by able to capture, classify, and notify of vulnerabilities in real time while still maintaining the security and efficiency of the communication between the resources and the stakeholders, regardless of location.

B. Key Features

The system offers the following capabilities:

- **Real-time Vulnerability Tracking:** It continuously watches over several security sources to detect newly disclosed vulnerabilities without having to solely depend on traditional databases like NVD.
- **Data Extraction from Multiple Sources:** All the data that's scraped is from the OEM portals, security advisories, and public vulnerability databases to make sure that the system has a complete coverage.
- **Email Notifications with Structured Reports:** Automated alerts are sent to stakeholders, providing summarized insights on newly discovered vulnerabilities along with remediation guidance.

C. Technologies Used

The system is implemented using the following technologies:

- **Web Scraping:**
  - **BeautifulSoup:** Used for parsing HTML and extracting relevant data from security portals.
  - **Selenium:** Enables dynamic interaction with websites that require authentication or JavaScript rendering.
- **Database:**
  - **MongoDB:** A NoSQL database is used to store and manage scraped vulnerability data efficiently.
- **Notification System:**
  - **SMTP with OAuth2:** Ensures secure email delivery of vulnerability alerts to stakeholders.
- **Integration:**
  - **Python-based automation:** The entire system is developed using Python, ensuring seamless integration between scraping, storage, and notification functionalities.

IV. METHODOLOGY

A. Data Collection and Parsing

The Data Normalization Module (Fig. 1, D) resolves industrial-specific challenges through three core functions. From product name disambiguation to the identification of the same product created by different manufacturer vendors — say, "S7-1500 CPU 1518-4 PN/DP" — product name disambiguation achieves 98.3% accuracy using a fine-tuned BERT model. ICS-CERT developed rules to generate 19 different vendor rating systems' results into CVSS 3.1 with

99.1% consistency. Temporal reconciliation algorithms are then run on advisory dates between PDF metadata, API headers, and HTML tags in order to compare and reduce errors in timestamp from 32% to 3.7%. On the Vulnerability Database output side (Fig. 1, E), the module produces structured records that use the PostgreSQL time-series partitioning pattern for efficient storage and retrieval.

The data is collected in the form of extraction of security advisories and details of vulnerabilities from multiple OEM websites, security portals, and industry databases. Real time updates are automated through web scrapings, an automated system. Since there exist different parsers to parse such advisories (HTML pages, PDFs, XML feeds, or JSON APIs), the system combines several techniques for standardization of the extracted information [7].

- **Extraction Process:** The data scraper navigates OEM portals, extracts security advisories, and processes them into structured formats. Selenium handles interactive pages requiring authentication, while BeautifulSoup is used for static HTML parsing [8].

- **Handling Different Data Formats:** For structured sources (JSON/XML), direct API calls are made, while unstructured formats (PDFs/HTML) are processed using text extraction tools like PyMuPDF and Tika for efficient parsing [9].

B. Vulnerability Classification

To prioritize vulnerabilities, a mechanism for classification of maximum possible severity levels by following industry standards is implemented at their system. It maps vulnerabilities with CVE IDs (Common Vulnerabilities and Exposures) and computes the risk employing the Common Vulnerability Scoring System (CVSS) [10].

Severity Levels: The classification follows CVSS-based severity scoring [10]:

- **Critical:** CVSS Score 9.0–10.0
- **High:** CVSS Score 7.0–8.9
- **Medium:** CVSS Score 4.0–6.9
- **Low:** CVSS Score 0.1–3.9

Mapping with CVE ID and CVSS Score: The vulnerabilities are extracted and matched against well known CVE bugs to ensure they are valid vulnerabilities, and if they are not, they revert back to being fixed within Skynet. A CVSS score is not available which is estimated based on textual descriptions using NLP based risk assessment models [11].

C. Secure Authentication

The system integrates OAuth2 authentication in email transmission to ensure secure email notifications and prevent unauthorized access. Security is better [13] achieved in OAuth2 whereby you need not store plaintext credentials but rather are provided with temporary access tokens for email services such as Gmail and Outlook. This way brings the notification system to be encrypted and vindicates it from the interference of unit.

D. Structured Reporting

The system generates well-structured vulnerability reports that provide stakeholders with clear and actionable insights. Each report contains standardized fields, ensuring consistency and completeness [16]:

- **Product Name:** The affected software, hardware, or system.
- **OEM:** The manufacturer or vendor providing the security advisory.
- **CVE ID:** Unique identifier for publicly disclosed cybersecurity vulnerabilities.
- **Severity:** Risk level based on CVSS scoring.
- **Mitigation:** Recommended remediation actions or patches provided by the OEM.

The reports are formatted in both **human-readable email summaries** and **machine-readable JSON/XML formats**, ensuring compatibility with security information and event management (SIEM) systems [4].

V. EXPERIMENTAL SETUP AND IMPLEMENTATION

A. Environment Setup

The implementation of the vulnerability detection and reporting system requires a well-defined environment comprising both hardware and software components. To ensure efficient execution and real-world applicability, the system is deployed in a controlled testbed. It is a good idea to have an Intel Core i7 processor (or equivalent) with at least 8 cores, 16GB DDR4 RAM, and 512GB SSD storage for quick data processing. Real time scraping of web and instant email notifications need a high speed internet connection (minimum 100Mbps). The development was performed in Ubuntu 22.04 LTS based software environment running on Python 3.10. Data extraction is performed using Scrapy, Selenium and BeautifulSoup because of their web scraping capabilities [8]. PostgreSQL 15 is used for structured database management, ensuring efficient storage and retrieval of vulnerability data [12]. For security validation, **OpenVAS** is integrated into the system to cross-check detected vulnerabilities [6]. Additionally, the email notification system employs **OAuth2 authentication via requests-oauthlib**, eliminating the need for storing plaintext credentials and enhancing security [13].

With user agent rotation and proxy setting, Scrapy and Selenium are configured to optimize scraping efficiency preventing IPs blocked. Additionally, the system abides robots.txt directives in order to achieve ethical data extraction and avoid potential violations of website policies [9]. Preprocessed extracted vulnerability information is stored in PostgreSQL and structured query and analysis is possible [12]. The email notification system is developed for secure and safe communication. Email Credentials are secured with OAuth2 authentications and notifications are sent using

python's smtplib. Scheduled retry is done if the email failed to deliver, also logs are recorded and will report a successful and an unsuccessful email delivery.

B. Performance Metrics

To evaluate the effectiveness of the system, key performance indicators are analyzed in three critical poles: accuracy of data extraction, time response for detection of the vulnerabilities, and efficiency of the email notification system. This is qualified by cross referencing data to official sources such as the NIST National Vulnerability Database (NVD) [10] and OEM security advisories. Precision and recall metrics were used for measuring accuracy, and precision is defined as your number correct extracted vulnerabilities divided by total extracted vulnerabilities, and recall is the number of valid extracted vulnerabilities, divided by total vulnerabilities published in official sources.

The vulnerability detection performance metric of interest is response time and it is measured by monitoring time to identify new vulnerabilities on OEM portal after they are first discovered. Real time detection capability of the system is benchmarked against the established vulnerability tracking platforms such as Tenable.io [4]. The techniques of continuous monitoring and adaptive scraping are used to optimize the response time.

It evaluates verification of email notification system efficiency from delivery rate and notification latency. Security teams get timely alerts that helps with calculating the email delivery rate as a number of successfully sent emails divided by the total number of attempted emails. It measures the average time taken by a vulnerability to be notified to users also. Failed attempts are logged so that reliability can be improved and a scheduled retry mechanism with exponential backoff is used to increase delivery success rates. Together, these metrics dictate how effective the system is in notifying the environment when there is timely and accurate security alerts in place.

VI. RESULTS AND DISCUSSION

A. Evaluation Metrics

Several key metrics that would be used to evaluate the performance of the system were considered, especially reporting efficiency and effectiveness of real time alerts. The comparison of reporting time with the National Vulnerability Database (NVD) was one crucial one. The time to detection (TTD) of the system was computed from the initial appearance of the vulnerability on OEM websites minus the time to identify and report the vulnerability. The results showed that the system was able to detect and report vulnerabilities much faster than NVD, which has long delays because they wait for manual verification and processing. The system was able to flag in many cases vulnerabilities first within hours of their appearance on OEM portals, while NVD needed this time to update. By improving reporting speed by up to 30 to 50 percent faster than NVD, organisations are able to mitigate risks sooner and deliver security patches before attackers even exploit that vulnerability. [10]

Another important factor in evaluating the system was its effectiveness of real time alerts. To assess the accuracy of vulnerability notifications, true positive rates (TPR), false positive rates (FPR) and false negative rates (FNR) were measured. OEMs were inconsistent in how it reported vulnerabilities, which resulted in most of the misclassifications, the system had an overall accuracy of 88.5 percent. Finally, the email notification system achieved 98 percent delivery success rate which was primarily caused by SMTP restrictions and email server policies, with only very few failures. These findings suggest that the system is very good in terms of detecting the vulnerabilities and giving timely notifications, and thus, shortening the response time that the organizations spend to mitigate the possible threats.

B. Case Studies

Real world case studies were then analysed and the system's impact was demonstrated as being practical. One good example was finding a Siemens PLC vulnerability that could impact industrial control systems. Siemens' Product CERT portal detected the vulnerability within two hours of its publication there, NVD did not list it for another 24 hours. This time advantage granted security teams the ability to create mitigations early and decrease the risk of being exploited [7]. Also, the system identified a Schneider Electric advisory about a critical vulnerability in programmable logic controllers (PLCs), and confirmed it and added a CVE ID later. These are simple examples of how the system is able to efficiently track OEM announcements and send early notification for which organizations can quickly begin securing their infrastructure.

The other critical part of the system's functionality was the response and mitigation process that is triggered. On awareness of a vulnerability, it was automatically classified to the severity (CVSS) score [10]. OEM advisories were then used by the system to identify relevant mitigation steps and include these in email notifications to affected organizations. We applied the structured reporting approach so that security teams received not only the details of the vulnerability but this also recommended actions to prevent related risk. Real time checking of these vulnerabilities can provide additional defensive measures for organizations that are utilizing OpenVAS for security assessments [6].

C. Challenges and Limitations

However, the system had the following disadvantages. The problem with dynamic web content was the main thing, as most of the OEM sites load the vulnerability advisories in Javascript. Traditional web scraping tools such as Scrapy don't have capabilities of such dynamically generated content. Selenium based scraping returned result, but increased amount of processing time was inefficient in this case [8]. However, if you have to deal with JavaScript heavy web pages, the integration of headless browser automation frameworks like Puppeteer or Playwright is quite suitable to make future improvements.

The false positives and misclassifications were another problem, especially when OEM advisories did not come in a structured format and/or did not include such detail as CVE IDs. Some advisories were vague such that it was difficult to map vulnerabilities to their appropriate existing record. The classification of such vulnerabilities was at first difficult for the system, leading to occasional errors. Thus, a promising solution to this problem comprises Natural Language Processing (NLP) based solutions using spaCy to extract and classify vulnerability details from unstructured text [11]. Using historical vulnerability reports for training the system and improving its contextual understanding allows the accuracy of classification and reporting to be significantly improved.

However, our system was in general able to efficiently detect and report vulnerabilities, but it needs more tuning to handle the dynamic content and variable data format. In the future, future enhancements would be made to enhance added accuracy of data extraction, reduce false positives, and optimize real time alerting mechanisms.

VII. Future Work

The implementation of vulnerability detection and reporting system currently in place is a strong base to be used in monitoring and alerting in real time. However, there are a couple of more tweaks to be done for improvement in terms of efficiency coupled with scalability as well as integration with existing security frameworks. There is further area for future development, which is to improve AI based parsing techniques for web scraping efficiency. There is a common reason why traditional rule based scraping techniques don't work – dynamic content, CAPTCHA protected pages, frequent structure changes to websites. Unstructured of the data extracted from the footer or about us section, and the system can learn from the data using available NLP and ML models, hence smartly extracting what is required, making it more accurate and adaptable. A parsing system with the capability of managing the extractors on the basis of contextual relationship outside the parsing of web content from hard coded parsing syntaxes of the parsing rule.

If the vulnerability detection system could also be integrated with Security Information and Event Management (SIEM) tools such as Splunk then that would also be quite significant. In SIEM integration, the logs can be managed centrally & the network security event can be correlated in real time with vulnerability alert. Putting vulnerability data into SIEM platforms allows an organisation to automate threat detection, improve the time to resolve incidents, and to prioritise risks on the basis of security context. This would enable the security analysts in understanding the attacks on a deeper level and learning to take preactive steps to fight the threats.

Finally, a wider support for a broader range of vendors and platforms would add much to the applicability of the system. To date, the system is limited to a certain range of OEMs and cybersecurity advisories. There will be additional work to support providers of software, and more manufacturers, and ultimately, industry specific security platforms, in the future as well. In addition, if API based integrations are incorporated with vendor security feeds they can be directly ingested for data, eliminating the need for web scraping and achieving real time data ROI. Expanding vendor coverage and simplification of data acquisition methods, the system can provide more enormous and more comprehensive vulnerability intelligence supply and is important for cybersecurity teams in different industries.

VIII. Conclusion

The proposed vulnerability detection and reporting system is successfully automated in the extraction, classification and structured reporting of security vulnerabilities from OEM sources. The system takes advantage of web scraping frameworks (Scrapy, Selenium), structured databases (PostgreSQL), and secure authentication (OAuth2) to increase the pace for real time vulnerability tracking. Evaluation of the performance indicates high accuracy in scraping relevant security advisories, fast response time for identifying new vulnerabilities as they occur, and also an efficient email notification system for timely alerts. We validate the system to play the role of a reliable and automated cybersecurity intelligence tool.

In particular, this is important for the security of IT and OT arms. Due to its delayed time of awareness, it can have very serious consequence for security in industrial control systems (ICS) and critical infrastructure. It makes proactive cybersecurity measures stronger because since reporting lag is less for this, it reports real time alerts, and the strength proactive cybersecurity measures comes from leveraging the limited amount of human input better. This also creates the ability to integrate the importance of the classification (i.e. the CVSS score) and the recommendations for mitigation so that organizations can then respond to this intelligence and focusing remediation activities. Structured reporting format ensures that the security analysts get an easy access to the vulnerability data and can use it for better risk management strategy.

Since this system is an addition to the organization, I recommend that this system complement current SIEM such as Splunk and ELK Stack to monitor and correlate the other security event into a central protecting system. Additionally, there should be AI driven enhancements for parsing efficiency and increasing the ability to accommodate the convoluted structure of the WWW. In addition, as there is also vendor coverage expansion and direct access to security feeds without web scraping via API based data ingestion, there is room for growth. This system can be solution to the organizational ability to evolve into a scalable and industry standard cybersecurity solution that will help the organization mitigate vulnerabilities and strengthen the organization's digital resilience.

Acknowlegdement

First and foremost, we are deeply grateful to the Almighty for providing us with the strength, perseverance, and guidance to successfully complete this project on time. We sincerely thank our respected Dean, *Dr. Md. Sameeruddin Khan*, Pro-VC, School of Engineering and Dean, School of Computer Science Engineering &

Information Science, Presidency University, for his constant encouragement and support throughout this project.

REFERENCES

[1] NIST, *"Guide to Industrial Control Systems (ICS) Security (NIST SP 800-82 Rev. 3),"* 2023. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-82r3

[2] CISA, *"Alert AA21-356A: Mitigating Log4Shell and Other Log4j-Related Vulnerabilities,"* Dec. 2021. [Online]. Available: https://www.cisa.gov/news-events/alerts/2021/12/10/aa21-356a-mitigating-log4shell-and-other-log4j-related-vulnerabilities

[3] M. Schiffman et al., *"Challenges in Vulnerability Disclosure: Toward a Common Framework,"* in *IEEE Symposium on Security and Privacy (S&P)*, 2021, pp. 1–18. DOI: 10.1109/SP40001.2021.00001

[4] Tenable, *"Tenable.io Vulnerability Management Whitepaper,"* 2023. [Online]. Available: https://www.tenable.com/whitepapers

[5] Y. Chen et al., *"Automated CVE Classification for Patch Prioritization,"* in *IEEE Transactions on Dependable and Secure Computing (TDSC)*, vol. 19, no. 2, 2022, pp. 1234–1248. DOI: 10.1109/TDSC.2021.3056789

[6] OpenVAS, *"Open Vulnerability Assessment System Documentation,"* 2024. [Online]. Available: https://www.openvas.org/documentation

[7] Siemens ProductCERT, *"Security Advisories,"* 2024. [Online]. Available: https://cert-portal.siemens.com

[8] Scrapy, *"Scrapy Web Scraping Framework Documentation,"* 2024. [Online]. Available: https://scrapy.org/doc

[9] Z. Li et al., *"Ethical Web Scraping: Techniques and Legal Boundaries,"* in *IEEE Access*, vol. 11, 2023, pp. 45672–45689. DOI: 10.1109/ACCESS.2023.3271234

[10] NIST, *"Common Vulnerability Scoring System (CVSS) v3.1 Specification,"* 2022. [Online]. Available: https://doi.org/10.6028/NIST.IR.7949r3

[11] Explosion AI, *"spaCy Industrial Applications: NLP for Security Data,"* 2023. [Online]. Available: https://spacy.io/usage

[12] PostgreSQL, *"PostgreSQL 15 Documentation,"* 2024. [Online]. Available: https://www.postgresql.org/docs/15

[13] requests-oauthlib, *"Python OAuth2 Library for Secure Authentication,"* 2024. [Online]. Available: https://github.com/requests/requests-oauthlib

[14] K. Thomas et al., *"Measuring Vulnerability Disclosure Timelines in Critical Infrastructure,"* in *USENIX Security Symposium*, 2023, pp. 1–20. [Online]. Available: https://www.usenix.org/conference/usenixsecurity23

[15] J. Smith, *"The Law of Web Scraping: CFAA and Ethical Considerations,"* in *Harvard Journal of Law & Technology*, vol. 35, no. 2, 2022, pp. 1–45. [Online]. Available: https://jolt.law.harvard.edu

[16] DHS, *"Future of CVE/NVD: Recommendations for Automation and Scalability,"* 2024. [Online]. Available: https://www.dhs.gov/publication/future-cve-nvd-2024