Journal of Intelligent Decision Support System (IDSS) 8 (1) (2025), pp. 44-53

Published by: Institute of Computer Science (IOCS)

Journal of Intelligent Decision Support System (IDSS)

Journal homepage: www.idss.iocspublisher.org

The development of a data lakehouse system for the integration and management of cyber threat intelligence data in XYZ unit

Ricky Chan[1], Rendi Hanif Dhaifullah[2], Hondor Saragih[3], Nadiza Lediwara[4], Rochedi Idul Adha[5]

[1,2,3,4,5] *Informatika, Universitas Pertahanan Republik Indonesia, Bogor, Indonesia*

Article Info

Article history:

Received Mar 06, 2025
Revised Mar 29, 2025
Accepted Mar 28, 2025

Keywords:

Apache Spark;
Cyber Threat Intelligence;
Data Lakehouse;
Dremio;
MinIO.

ABSTRACT

Cybersecurity systems are evolving to deal with increasingly complex digital threats. One of the main challenges in this field is integrating and managing Cyber Threat Intelligence (CTI) efficiently. This research aims to design and implement Data Lakehouse as a solution to manage CTI data in XYZ Unit. The system was built using Apache Spark, MinIO, Dremio, Nessie, and Apache Iceberg with a containerization approach using Docker to ensure flexibility and ease of implementation. The implementation results show that the system successfully integrates various CTI data sources and improves efficiency in data storage, processing, and analysis. MinIO is used as the primary storage, Apache Spark processes data at scale, Dremio enables real-time data analysis, and Nessie manages data version control to maintain its integrity. Blackbox testing proves that the system can work optimally, with results showing improved data integration and efficiency in managing cyber threat information. Thus, the developed Data Lakehouse can be an effective solution in supporting threat detection and strategic decision-making in XYZ Unit.

Corresponding Author:

Ricky Chan,
Program Studi Informatika,
Universitas Pertahanan Republik Indonesia,
Kawasan IPSC Sentul, Sukahati, Kec. Citeureup, Kabupaten Bogor, Jawa Barat, 16810, Indonesia
Email: chan.ricky500@gmail.com

Introduction

Nowadays, cyber threats are increasingly complex, encompassing various forms of attacks such as hacking, malware, and data leakage (Mallick & Nath, 2024). Cybersecurity refers to the practice of protecting systems, networks, and data from digital threats (Le et al., 2019). A typical cybersecurity system consists of a network security system and a computer security system, which includes firewalls, antivirus software, and intrusion detection systems. (Xin et al., 2018).

One of the challenges in cybersecurity is ensuring the CIA triad (Confidentiality, Integrity, and Availability), which is the foundation of information security (Harahap et al., 2023). Confidentiality focuses on ensuring that information can only be accessed by those with proper authority or permission (Mitchell & Osazuwa, 2023). Integrity emphasizes the importance of maintaining the accuracy and consistency of data throughout its lifecycle (Yin et al., 2020). Meanwhile, Availability refers to the system's ability to provide access to information for authorized users whenever necessary. (Helmiawan et al., 2024).

To meet cybersecurity challenges, organizations rely on Cyber Threat Intelligence (CTI), an approach that focuses on collecting, analyzing and utilizing threat-related information. Two legislative acts that enforce this strategy are the Network and Information Security (NIS) Directive and the General

Data Protection Regulation (GDPR), which require organizations to share cyber incident information as well as data breach reports with relevant authorities (Spiga et al., 2022). The CTI community aims to utilize this information, understand the situation through analysis, and share the intelligence gained with the community, organizations, and the public (Schaberreiter et al., 2019). Additionally, OpenIOC, developed by Mandiant, is an extensible XML schema that contains technical characteristics that identify known threats, attacker methodologies, or other indicators of compromise (Zhao et al., 2020).

A major concern in cybersecurity is data breaches, which occur when sensitive or confidential information is accessed, stolen or leaked without authorization (Seh et al., 2020). Data leaks usually result from improper encryption and stolen credentials, which is one of the simplest and most common ways for cyber attackers to hack or infiltrate a system when the system uses passwords that are easy to guess and decrypt (Wang & Johnson, 2018). In the past decade, several high-profile data leaks have raised awareness of the consequences of such leaks. One of the most prominent incidents was the Target Corporation network leak in 2013, which led to the theft of approximately 40 million credit card data and 70 million identifiable customer data, with losses estimated at USD 248 million. Similarly, in 2016, Yahoo reported that at least 500 million accounts had been stolen in a 2014 data breach, which was allegedly backed by a country (Molitor et al., 2023).

To enhance CTI implementation, organizations need a system that efficiently manages and analyzes large amounts of threat intelligence data (Kayode-Ajala, 2023). The Data Lakehouse architecture, which combines the advantages of Data Lake and Data Warehouse, enables large-scale data storage with high flexibility while supporting efficient data analysis (Lavrentyeva & Sherstnev, 2022). The Data Lake is a centralized repository that enables storage of all types of data, including structured data from relational databases (rows and columns), semi-structured data (CSV, logs, XML, JSON), unstructured data (emails, documents, PDFs), and binary data (images, audio, video) at any scale (Andriyani et al., 2023). James Dixon first introduced the term “data lake” in 2010, comparing a data warehouse to a water bottle that represents cleaned and structured data, while a data lake is described as a large body of water in a more natural state that contains raw and unprocessed data (Giebler et al., 2019). In the big data era, new ideas and techniques for storing and processing diverse and evolving data are essential (Khine & Wang, 2018).

Previous studies have explored the advantages of Data Lakes, Data Warehouses, and Data Marts for data management (Al-Okaily et al., 2023). Data Warehouses are structured for analytical processing but struggle with handling semi-structured and unstructured data (Janssen, 2022). Meanwhile, Data Lakes provide scalable storage but often suffer from performance issues in handling diverse workloads (Mehmood et al., 2019). To overcome these challenges, a third-generation architecture called Data Lakehouse was introduced in 2020, combining the cost-effective storage and flexibility of a Data Lake with the analytical capabilities of a Data Warehouse (Janssen, 2022). Another advantage of Data Lakehouse is its ability to support real-time streaming use cases (Armbrust et al., 2021). This architecture also enables real-time data streaming and processing, leveraging technologies such as Apache Spark and Kafka (Orescanin & Hlupic, 2021). However, research on Data Lakehouse implementation specifically for CTI data integration and management remains limited, highlighting the gap that this study aims to address.

Although Data Lakehouse architectures have been implemented in various sectors such as finance, healthcare, and retail, existing research primarily focuses on their use for business intelligence, customer analytics, and large-scale data storage optimization (Harby & Zulkernine, 2024). For instance, in the financial industry, Data Lakehouses have been used to enhance fraud detection by enabling real-time analysis of transactional data and customer behavior patterns. In the healthcare sector, Data Lakehouses facilitate predictive analytics by integrating diverse data types, such as electronic medical records and genomic data, to provide comprehensive patient insights (Begoli et al., 2021). Meanwhile, in the retail industry, Data Lakehouses improve supply chain efficiency by consolidating data from multiple sources, allowing for more accurate demand forecasting and inventory management (Harby & Zulkernine, 2024).

However, research on the implementation of Data Lakehouse specifically for cybersecurity and Cyber Threat Intelligence (CTI) remains scarce. Unlike its applications in finance and healthcare, which primarily deal with structured and transactional data, CTI involves highly dynamic, semi-structured, and

unstructured threat intelligence data, necessitating real-time processing and version control (Begoli et al., 2021). In CTI systems, data from various sources such as Telegram, hacking forums, and other cyber intelligence feeds must be collected, stored, and analyzed efficiently to enhance the accuracy and speed of threat detection. The ability to process such complex data types in real time is crucial for improving cyber threat mitigation strategies and decision-making.

To address these challenges, this research proposes the development of a Data Lakehouse architecture optimized for CTI data integration and management. The system is designed to support data ingestion from multiple cybersecurity sources, facilitate structured storage, and enable efficient query execution for rapid threat analysis. Unlike conventional Data Lakehouse implementations that focus on structured datasets, this approach emphasizes scalability, adaptability, and real-time intelligence processing, ensuring enhanced cyber defense capabilities. By bridging the gap between traditional data management frameworks and cybersecurity-specific requirements, this study contributes to advancing CTI processing using Data Lakehouse technology.

Method

This research uses the SDLC Waterfall method in developing Data Lakehouse to integrate and manage Cyber Threat Intelligence (CTI) data in XYZ Unit. The Waterfall model was chosen because it provides a structured, systematic, and sequential approach, which ensures that each stage must be completed before moving on to the next stage (Haerani et al., 2023). The stages in the Waterfall method can be seen in Figure 1, including Requirement Analysis, System Design, Implementation, Integration & Testing (Heriyanti & Ishak, 2020).

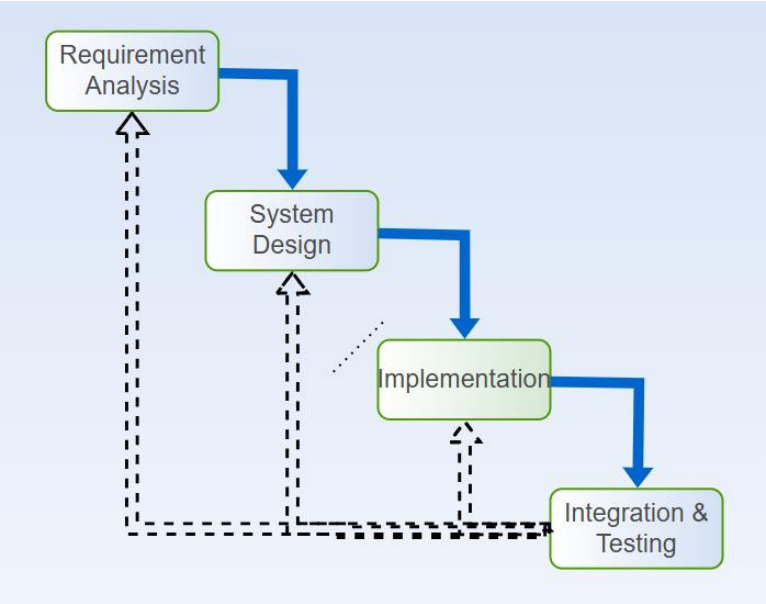

Figure 1 Flowchart of Waterfall Model

Requirement Analysis

At this stage, we identify and observe what is needed by the XYZ Unit to help the team related to this CTI in carrying out data security tasks. In the implementation of the existing CTI in the XYZ Unit, especially in handling data leaks, we found that credential data that leaked every day from various sources was not integrated into a system. Then, data collection related to system needs and analysis was also carried out through interviews and consultations with clients. They said that the current leaked credential data is still not integrated, which makes it a little difficult to handle the credential data.

Therefore, the client said that there must be a system that can integrate all the data, then suggested making a data lakehouse that not only integrates all the data, but can also make it easier for the XYZ Unit to analyze or query the data that has been leaked.

Then, the source of the leaked credential data we collected in this project came from telegram channels. The reason for choosing telegram channels as a source of project data is because telegram is a source of leak data that is being widely used today after breach forums and twitter. In addition, a lot of these data are open source or shared at no cost. The results of identifying hardware and software requirements as shown in Table 1 and Table 2.

Table 1. Identification of Hardware Requirements

Components	System Specification
Processor	AMD 8GHz
RAM	16 GB (bigger is better for data-intensive processes)
Storage	SSD 1 TB
Network	Minimum 1 Gbps Ethernet
Operating System	Linux (Ubuntu 20.04 LTS)

Table 2. Identification of Software Requirements

Software	Function
Docker	Create isolated and manageable container environments for each service in the data lakehouse
Apache Spark (v33.x)	Distributed data processing, analysis, and large-scale data transformation
MinIO (Version RELEASE.2024-07-10T18-41-49Z-cpuv1)	S3-based object storage platform used to store raw data and processing results
Dremio	Platform for interactive SQL querying and data analysis with high performance
Nessie (Version 0.76.0)	Stores metadata and enables data version management within the data lakehouse
Apache Iceberg	A table format that supports distributed schemas and is used to organize data files in the form of large tables that can be accessed efficiently.

System Design

This stage includes designing a Data Lakehouse architecture that aims to integrate and manage Cyber Threat Intelligence (CTI) data in the XYZ Unit environment. This design considers the need for flexible storage, fast data processing, and the ability to query and analyze data efficiently.

Use Case Diagram

Use case diagram is used to model the interaction between actors and the system, assisting in describing how the designed system will function from the user's point of view. In this project, the Data Lakehouse system is designed to manage and process Cyber Threat Intelligence (CTI) data collected from various sources, including Telegram. This diagram displays the relationship between the two main actors in the system, namely Admins and Verified Users, as well as the various key functions they use in the Data Lakehouse ecosystem built using MinIO, Apache Spark, Dremio, and Nessie technologies.

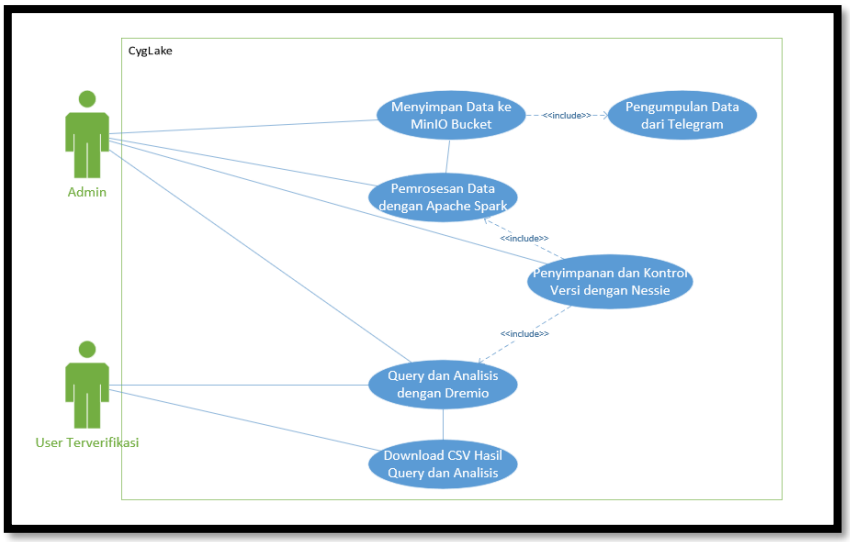

Figure 2 Use Case Diagram

Data Flow

(1) Data Collection; data is collected from CTI sources such as Telegram using a web scraper or Telegram API. The collected data in JSON or TXT format is sent to MinIO as the initial storage. (2) Data Processing; Apache Spark retrieves data from MinIO, then performs cleaning, parsing, and transformation. The processed data is stored in Iceberg tables for easy management and analysis. Nessie records any changes to the dataset made by Apache Spark. (3) Data Analysis and Query; Users with access can run queries against CTI data using Dremio. Dremio pulls data from Iceberg managed by Nessie to ensure the version of data used is always valid. The query results can be exported in CSV or JSON format for further needs.

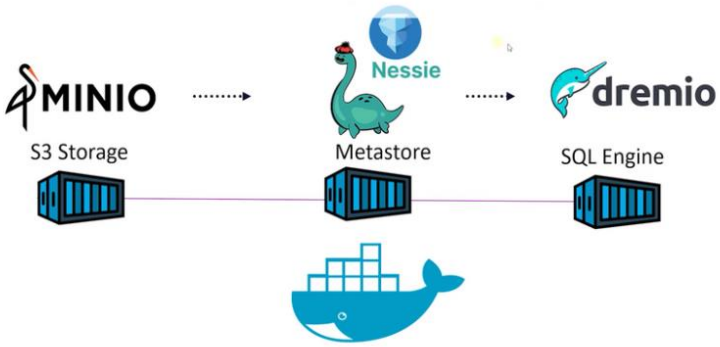

Figure 3 Data Flow Data Lakehouse System

Implementation

The implementation stage in the SDLC Waterfall model is the realization process of the system design that has been made. At this stage, the Data Lakehouse system begins to be built by implementing the previously designed technical configuration. The implementation is carried out in stages by integrating all the main components used in the system, namely MinIO, Apache Spark, Dremio, Nessie, and Apache Iceberg.

The system uses Docker Compose, which allows each service to run in an isolated and well-integrated environment. The first step in the implementation is to prepare a Docker Compose File, which

includes configurations for each major component. MinIO is used as an object store to hold raw data collected from CTI sources, Apache Spark serves as a data processing engine to perform data cleaning, transformation, and processing, while Dremio is used as a query engine that allows users to explore data using SQL. In addition, Nessie acts as a version control system to ensure data integrity in Data Lakehouse, while Apache Iceberg is used as a table format that supports large-scale data management.

Integration & Testing

After all services run using Docker, the next step is to integrate the components so that they can work synchronously. The integration is done by connecting MinIO as the main storage with Apache Spark, which is in charge of reading and processing data from MinIO. Spark processing results are then stored in Iceberg tables to support large-scale data processing efficiency. Nessie is used to log changes to the dataset and allow rollback if needed, while Dremio is configured to read data from Iceberg and Nessie to allow users to run SQL queries for analysis of processed CTI data.

After the system is integrated, initial testing is carried out to ensure that each component works properly. At this stage, the test used is Blackbox Testing, which is a testing method that focuses on validating the functionality of the system without seeing how the system works internally. This test is carried out by providing certain inputs and evaluating whether the resulting output is as expected.

Testing was conducted in several key scenarios. First, functional testing was conducted to evaluate whether each component worked according to its specifications. For example, tests were conducted to ensure that Apache Spark could read, process and store data into Iceberg, and that MinIO was able to store and manage data properly. Dremio was tested to ensure that SQL queries could be executed correctly, while Nessie was tested to ensure that data version changes could be controlled properly. Next, integration testing was conducted to ensure that all components can communicate and exchange data properly. At this stage, it is tested whether the data processed by Apache Spark can be accessed through Dremio, whether changes to the Iceberg table can be controlled by Nessie, and whether Dremio can read and manage version metadata stored in Nessie. This test aims to ensure that the system can work in a coordinated manner without losing data or experiencing inconsistencies in metadata management.

Results and Discussions

The first result obtained is the successful integration of the main components in the Docker environment. Each service can run well and communicate with each other according to the architecture design. MinIO is successfully used as the main storage of CTI data, while Apache Spark is able to read, process, and convert data into a more structured format before being stored in Iceberg. Dremio was able to read and execute SQL queries against the data stored in Iceberg, and Nessie was able to manage data versions and log changes that occurred during each processing process. This integration proves that the system can run in a coordinated manner and is ready to be used for cyber threat data analysis.

In figure 4, we see the MinIO that has been prepared and can be directly used as storage for CTI breach data or metadata. We have also tested that the entire MinIO storage process runs smoothly. After the data processing from Apache Spark is saved into MinIO, a file titled '_SUCCESS' with the processed csv file will appear in the folder.

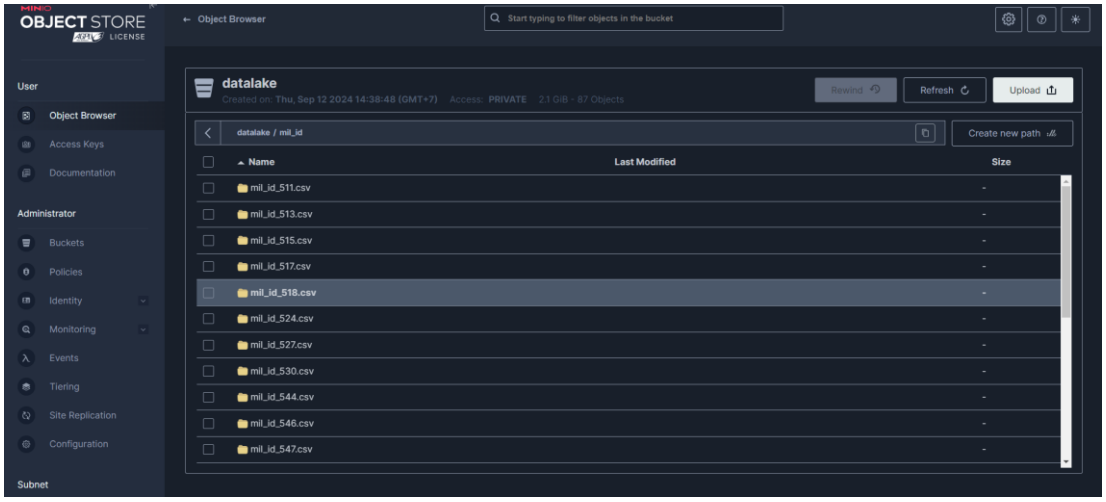

Figure 4 MinIO View

In Figure 5, a transformation script was developed to convert unstructured raw text files collected from Telegram channels into structured CSV format using Apache Iceberg. The integration of Nessie ensures that every metadata change or data catalog update is tracked, providing robust version control. Furthermore, the processed data can be used for advanced analysis, such as machine learning or AI-based threat detection models integrated within Apache Spark. This capability expands the system's use case from simple data management to automated threat intelligence processing.

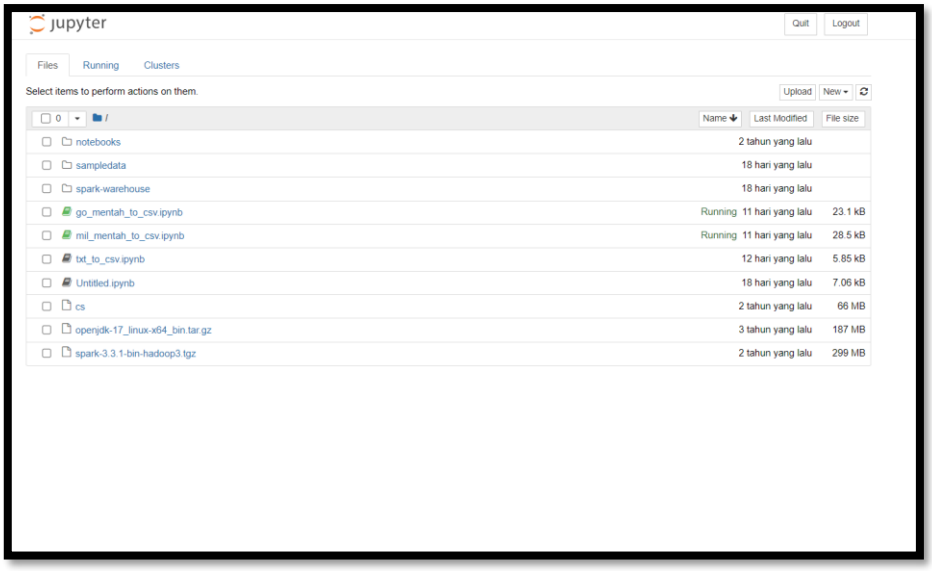

Figure 5 Apache Spark View

The version control functionality of Nessie, as shown in Figure 6, provides historical tracking of data catalog changes made by Apache Spark and Dremio. Users can roll back to previous versions, create data branches, and merge modifications, maintaining data integrity across the system. This Git-like approach ensures that analysts and security professionals have access to well-organized and auditable threat intelligence data, improving traceability and forensic investigation capabilities.

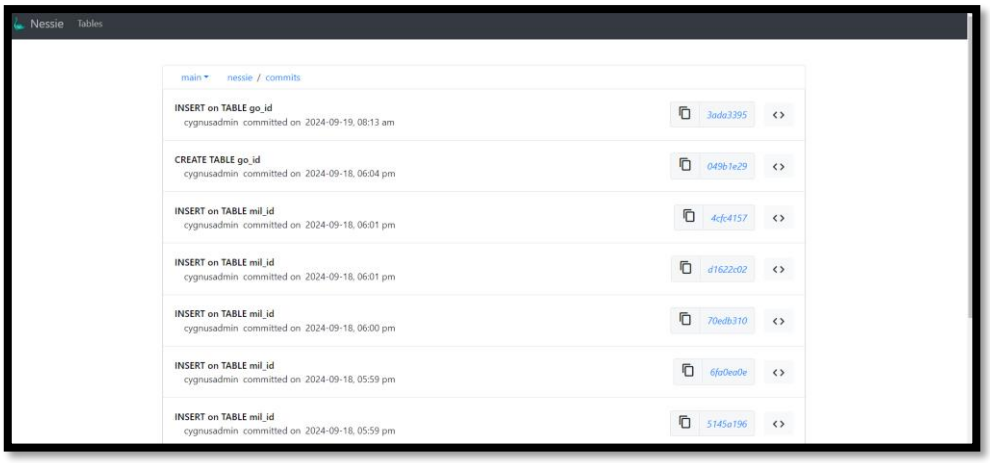

Figure 6 Nessie View

The Dremio interface, depicted in Figure 7, plays a central role in querying and analyzing CTI data. This interface allows users to perform real-time SQL queries on Nessie-managed datasets, integrating credential breach data categorized by domain. By leveraging Dremio's high-performance query engine, analysts can quickly extract actionable insights to support cyber threat intelligence operations. This confirms that the Data Lakehouse framework not only facilitates efficient data storage and processing but also enhances data-driven decision-making in cybersecurity.

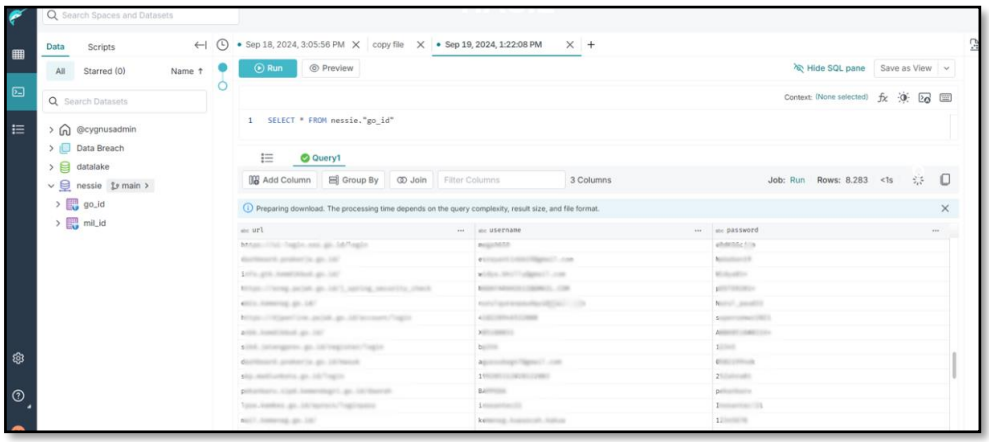

Figure 7 Dremio View

One of the significant benefits of this Data Lakehouse implementation is its scalability and adaptability in handling large-scale cybersecurity data. Unlike traditional data management systems, which struggle with integrating structured and unstructured threat intelligence data, this architecture ensures that CTI data is efficiently stored, processed, and analyzed in a single framework. The combination of Apache Iceberg and Nessie allows for structured storage while maintaining flexibility in tracking changes and managing multiple data versions, providing cybersecurity professionals with reliable access to historical and current threat intelligence.

Additionally, the system's ability to support real-time data processing and querying enables faster threat detection and response. The integration of Apache Spark allows for advanced analytics, such as anomaly detection and predictive threat modeling, which can significantly enhance cybersecurity operations. Future enhancements could include the incorporation of machine learning

models for automated threat classification, further improving the efficiency and accuracy of cyber intelligence analysis.

Conclusions

This research advances the field of Cyber Threat Intelligence (CTI) data management by proposing a Data Lakehouse architecture that integrates Apache Spark, MinIO, Dremio, and Nessie within a Docker-based environment. The primary objective of this study was to design a system that enhances data integration, structured storage, and real-time analysis, addressing existing limitations in cybersecurity data management. By incorporating version control through Nessie and utilizing Iceberg for structured storage, this research bridges the gap between conventional data warehouses and cybersecurity intelligence systems. The findings contribute to improving the scalability and efficiency of CTI data processing, ensuring that threat intelligence data can be managed dynamically while maintaining integrity and accessibility. This study highlights the potential for Data Lakehouse architectures to be further developed in cybersecurity applications, offering a structured and adaptive approach to handling evolving cyber threats. Future research should explore the integration of automated threat intelligence processing using machine learning models, enhancing real-time anomaly detection and predictive analytics. Additionally, expanding the system to support distributed multi-node computing would allow for greater scalability in managing large-scale CTI datasets. Further experimentation with real-time stream processing technologies, such as Apache Flink or Kafka, could enhance the system's ability to process and respond to emerging cyber threats dynamically. These advancements would position Data Lakehouse architectures as a core component of future cybersecurity infrastructures, supporting more proactive and adaptive cyber defense strategies.

References

Al-Okaily, A., Al-Okaily, M., Teoh, A. P., & Al-Debei, M. M. (2023). An empirical study on data warehouse systems effectiveness: the case of Jordanian banks in the business intelligence era. *EuroMed Journal of Business*, *18*(4), 489–510. https://doi.org/10.1108/EMJB-01-2022-0011

Andriyani, W., Dawis, A. M., & Purnomo, R. (2023). DATA LAKE INSIGHTS. In *Widina Media Utama* (First Edit). Widina Media Utama. http://scioteca.caf.com/bitstream/handle/123456789/1091/RED2017-Eng-8ene.pdf?sequence=12&isAllowed=y%0Ahttp://dx.doi.org/10.1016/j.regsciurbeco.2008.06.005%0Ahttps://www.researchgate.net/publication/305320484_SISTEM_PEMBETUNGAN_TERPUSAT_STRATEGI_MELESTARI

Armbrust, M., Ghodsi, A., Xin, R., & Zaharia, M. (2021). Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics. *11th Annual Conference on Innovative Data Systems Research, CIDR 2021*.

Begoli, E., Goethert, I., & Knight, K. (2021). A Lakehouse Architecture for the Management and Analysis of Heterogeneous Data for Biomedical Research and Mega-biobanks. *Proceedings - 2021 IEEE International Conference on Big Data, Big Data 2021*, *December 2021*, 4643–4651. https://doi.org/10.1109/BigData52589.2021.9671534

Giebler, C., Gröger, C., Hoos, E., Schwarz, H., & Mitschang, B. (2019). Leveraging the Data Lake: Current State and Challenges. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11708 LNCS*(DaWaK), 179–188. https://doi.org/10.1007/978-3-030-27520-4_13

Haerani, R., Hendriyati, P., Nugroho, P. A., & Lukman, M. (2023). Waterfall Model Implementation in Information Systems Web Based Goods Delivery Service. *JURTEKSI (Jurnal Teknologi Dan Sistem Informasi)*, *9*(3), 501–508. https://doi.org/10.33330/jurteksi.v9i3.2267

Harahap, A. H., Difa Andani, C., Christie, A., Nurhaliza, D., & Fauzi, A. (2023). Pentingnya Peranan CIA Triad Dalam Keamanan Informasi dan Data Untuk Pemangku Kepentingan atau Stakholder. *Jurnal Manajemen Dan Pemasaran Digital*, *1*(2), 73–83.

Harby, A. A., & Zulkernine, F. (2024). Data Lakehouse: A Survey and Experimental Study. *Information Systems*, *00*(July 2024), 1–23. https://doi.org/10.1016/j.is.2024.102460

Helmiawan, M. A., Akbar, Y. H., & Mahardika, F. (2024). Keamanan Teknologi Informa: Teori, Risiko, dan Strategi Pertahanan di Era Digital. In U. Press (Ed.), *UNSAP Press 2024*. UNSAP Press. http://scioteca.caf.com/bitstream/handle/123456789/1091/RED2017-Eng-

8ene.pdf?sequence=12&isAllowed=y%0Ahttp://dx.doi.org/10.1016/j.regsciurbeco.2008.06.005%0Ahttps://www.researchgate.net/publication/305320484_SISTEM_PEMBETUNGAN_TERPUSAT_STRATEGI_MELESTARI

Heriyanti, F., & Ishak, A. (2020). Design of logistics information system in the finished product warehouse with the waterfall method: Review literature. *IOP Conference Series: Materials Science and Engineering*, *801*(1). https://doi.org/10.1088/1757-899X/801/1/012100

Janssen, N. E. (2022). *The Evolution of Data Storage Architectures : Examining the Value of the Data Lakehouse*. 133.

Kayode-Ajala, O. (2023). Applications of Cyber Threat Intelligence (CTI) in Financial Institutions and Challenges in Its Adoption. *Applied Research of Artificial Intelligence and Cloud Computing*, *6*(1), 1–21. https://researchberg.com/index.php/araic/article/view/159

Khine, P. P., & Wang, Z. S. (2018). Data lake: a new ideology in big data era. *ITM Web of Conferences*, *17*, 03025. https://doi.org/10.1051/itmconf/20181703025

Lavrentyeva, Y., & Sherstnev, A. (2022). *The Definitive Guide to Data Warehouse vs. Data Lake vs. Data Lakehouse — ITRex*. https://itrexgroup.com/blog/data-warehouse-vs-data-lake-vs-data-lakehouse-differences-use-cases-tips/#

Le, D., Kumar, R., Mishra, B. K., Khari, M., & Chatterjee, J. M. (2019). Cyber Security in Parallel and Distributed Computing. *Cyber Security in Parallel and Distributed Computing*. https://doi.org/10.1002/9781119488330

Mallick, A. I., & Nath, R. (2024). Navigating the Cyber security Landscape: A Comprehensive Review of Cyber-Attacks, Emerging Trends, and Recent Developments. *World Scientific News: An International Scientific Journal*, *190*(1), 1–69. www.worldscientificnews.com

Mehmood, H., Gilman, E., Cortes, M., Kostakos, P., Byrne, A., Valta, K., Tekes, S., & Riekki, J. (2019). Implementing big data lake for heterogeneous data sources. *Proceedings - 2019 IEEE 35th International Conference on Data Engineering Workshops, ICDEW 2019*, 37–44. https://doi.org/10.1109/ICDEW.2019.00-37

Mitchell, O., & Osazuwa, C. (2023). Confidentiality, Integrity, and Availability in Network Systems: A Review of Related Literature. *International Journal of Innovative Science and Research Technology*, *8*(12). https://doi.org/10.5281/zenodo.10464076

Molitor, D., Raghupathi, W., Saharia, A., & Raghupathi, V. (2023). Exploring Key Issues in Cybersecurity Data Breaches: Analyzing Data Breach Litigation with ML-Based Text Analytics. *Information (Switzerland)*, *14*(11). https://doi.org/10.3390/info14110600

Orescanin, D., & Hlupic, T. (2021). Data Lakehouse - a Novel Step in Analytics Architecture. *2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO)*, 1242–1246. https://doi.org/10.23919/MIPRO52101.2021.9597091

Schaberreiter, T., Kupfersberger, V., Rantos, K., Spyros, A., Papanikolaou, A., Ilioudis, C., & Quirchmayr, G. (2019). A quantitative evaluation of trust in the quality of cyber threat intelligence sources. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3339252.3342112

Seh, A. H., Zarour, M., Alenezi, M., Sarkar, A. K., Agrawal, A., Kumar, R., & Khan, R. A. (2020). Healthcare data breaches: Insights and implications. *Healthcare (Switzerland)*, *8*(2). https://doi.org/10.3390/healthcare8020133

Spiga, D., Ciangottini, D., Costantini, A., Cutini, S., Duma, C., Gasparetto, J., Lubrano, P., Martelli, B., Ronchieri, E., Salomoni, D., Sergi, G., Storchi, L., & Tracolli, M. (2022). Open-source and cloud-native solutions for managing and analyzing heterogeneous and sensitive clinical Data. *Proceedings of Science*, *415*(Isgc), 21–25. https://doi.org/10.22323/1.415.0022

Wang, P., & Johnson, C. (2018). Cybersecurity Incident Handling: a Case Study of the Equifax Data Breach. *Issues In Information Systems*, *19*(3), 150–159. https://doi.org/10.48009/3_iis_2018_150-159

Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H., & Wang, C. (2018). Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access*, *6*, 35365–35381. https://doi.org/10.1109/ACCESS.2018.2836950

Yin, L., Fang, B., Guo, Y., Sun, Z., & Tian, Z. (2020). Hierarchically defining Internet of Things security: From CIA to CACA. *International Journal of Distributed Sensor Networks*, *16*(1). https://doi.org/10.1177/1550147719899374

Zhao, J., Yan, Q., Li, J., Shao, M., He, Z., & Li, B. (2020). TIMiner : Automatically extracting and analyzing categorize d cyb er threat intelligence from social data. *Computers & Security Journal*, *95*. https://doi.org/10.1016/j.cose.2020.101867