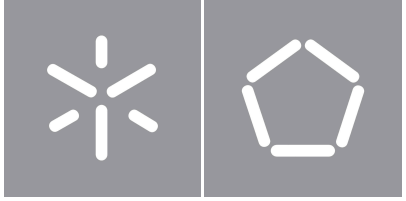
University of Minho
School of Engineering

Ricardo Alves Oliveira

Cyber Threat Intelligence Alert System

december 2025



University of Minho
School of Engineering

Ricardo Alves Oliveira

Cyber Threat Intelligence Alert System

Master's Dissertation in Informatics Engineering

Dissertation supervised by
João Marco Cardoso Silva

december 2025

Copyright and Terms of Use for Third Party Work

License granted to users of this work:

Acknowledgements

I would like to express my sincere gratitude to my supervisor, João Silva, for his continuous guidance, insightful feedback, and support throughout this dissertation. His expertise and observations were essential to the successful completion of this work.

I also extend my appreciation to Eurotux for providing the opportunity to undertake this project and to learn within a professional environment that values collaboration and innovation. In particular, I thank Mr. Fernando Gomes for his confidence in my capabilities and potential, my mentor Paulo Silva, and my team for their encouragement and support throughout this journey.

My deepest thanks go to my parents, whose unconditional support, patience, and sacrifices have provided me with every opportunity to pursue my goals. None of this would have been possible without their love, encouragement, and understanding.

I would also like to thank my girlfriend for her constant encouragement and motivation throughout this journey. Her belief in me and my work has been a continuous source of strength and inspiration.

Finally, I wish to thank my friends, both near and far, for their unwavering support and companionship during this challenging yet rewarding endeavour. I especially want to thank Bruno, Diogo, Gabriel, and Rodrigo for always being there through thick and thin, walking this path alongside me.

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

University of Minho, Braga, december 2025

Ricardo Alves Oliveira

Abstract

The rapid evolution of cyber threats, coupled with the exponential increase in available data, such as system logs and Cyber Threat Intelligence (CTI), poses considerable challenges for the enhancement and refinement of cybersecurity strategies. These data sources are frequently underutilised by current threat detection systems due to their inability to directly signal pertinent risks or attacks relevant to each environment, thereby constraining the identification of emergent threats. The increasing adoption of structured frameworks such as Structured Threat Information eXpression (STIX) has facilitated the dissemination of CTI, including Indicator of Compromise (IOC)s and existing attack patterns. Nonetheless, the effective utilisation of this information to refine cybersecurity workflows remains insufficiently explored by the scientific community.

This dissertation bridges this gap by designing and implementing a CTI-based alert system, capable of dynamically filtering, correlating and contextualising CTI to produce contextualised security alerts. By integrating intelligence feeds and parsing system data, such as running processes and ongoing network connections, the system complements current threat detection practices, providing timely and contextualised alerts for both current and emerging threats. This document further assesses the applicability of CTI in augmenting threat identification and response processes, constituting a significant advance in the way Cyber Threat Intelligence is addressed in the detection of cyber threats, tackling the challenges of its utilisation within a proactive cybersecurity framework.

Keywords Cyber Threat Intelligence, STIX, Cybersecurity, TAXII

Resumo

A rápida evolução das ciberameaças, juntamente com o aumento exponencial dos dados disponíveis, como logs de sistema e Cyber Threat Intelligence (CTI), impõe desafios consideráveis à melhoria e refinamento das estratégias de cibersegurança. Estas fontes de dados são frequentemente subutilizadas pelos sistemas atuais de deteção de ameaças devido à sua incapacidade de sinalizar diretamente riscos ou ataques pertinentes a cada ambiente, limitando, assim, a identificação de ameaças emergentes. A crescente adoção de estruturas padronizadas, como o Structured Threat Information eXpression (STIX), tem facilitado a disseminação de CTI, incluindo Indicator of Compromise (IOC)s e padrões de ataque existentes. No entanto, a utilização eficaz dessa informação para o aperfeiçoamento dos fluxos de trabalho de cibersegurança permanece insuficientemente explorada pela comunidade científica.

Esta dissertação colmata essa lacuna através da conceção e implementação de um sistema de alertas baseado em CTI, capaz de filtrar, correlacionar e contextualizar dinamicamente CTI para produzir alertas de segurança contextualizados. Ao integrar feeds de inteligência e analisar dados de sistema, como processos em execução e ligações de rede a decorrer, o sistema complementa as práticas atuais de deteção de ameaças, fornecendo alertas oportunos e contextualizados tanto para ameaças atuais como emergentes. Este documento avalia ainda a aplicabilidade da CTI no reforço dos processos de identificação e resposta a ameaças, representando um avanço significativo na forma como a Cyber Threat Intelligence é abordada na deteção de ciberameaças, enfrentando os desafios da sua utilização no âmbito de uma abordagem proativa de cibersegurança.

Palavras-chave Cyber Threat Intelligence, STIX, Cibersegurança, TAXII

Contents

List of Figures

List of Tables

Acronyms

0-Day Zero-Day Vulnerabilities.

AI Artificial Intelligence.

API Application Programming Interface.

APT Advanced Persistent Threat.

CA Certificate Authority.

CEF Common Event Format.

CERT Computer Emergency Response Team.

CLF Common Log Format.

CRUD Create, Read, Update, Delete.

CSAF Common Security Advisory Framework.

CSIRT Computer Security Incident Response Team.

CTI Cyber Threat Intelligence.

CVE Common Vulnerabilities and Exposures.

CVRF Common Vulnerability Reporting Framework.

CVSS Common Vulnerability Scoring System.

CyBOX Cyber Observable eXpression.

IDS Intrusion Detection System.

IOC Indicator of Compromise.

IP Internet Protocol.

JSON JavaScript Object Notation.

MISP Malware Information Sharing Platform.

ML Machine Learning.

NIST National Institute of Standards and Technology.

OSINT Open Source Intelligence.

OTX Open Threat Exchange.

POC Proof of Concept.

RBAC Role-Based Access Control.

SDO STIX Domain Object.

SIEM Security Information and Event Management.

SOAR Security Orchestration, Automation, and Response.

SOC Security Operation Center.

SQL Structured Query Language.

STIX Structured Threat Information eXpression.

TAXII Trusted Automated eXchange of Indicator Information.

TLP Traffic Light Protocol.

TLS Transport Layer Security.

TTE Time-to-Exploit.

TTP Tactics, Techniques, and Procedures.

UI User Interface.

URL Uniform Resource Locator.

XML eXtensible Markup Language.

Chapter 1

Introduction

The rise of the digital era has sparked remarkable innovations, yet it has simultaneously triggered a worrying surge in cyber threats that affect individuals, organisations, and governments worldwide. The increasing frequency and sophistication of cyberattacks, exemplified by incidents such as CVE-2024-3094 (The MITRE Corporation, 2024) and extensive ransomware campaigns targeting critical infrastructure, underscore the need for robust and proactive cybersecurity strategies. As cybercriminals continuously evolve their tactics, organisations must adapt their security systems to protect sensitive information and maintain operational stability.

1.1 Context and Motivation

Cyber Threat Intelligence (CTI) refers to both information and insights that were gathered, analysed, and then shared within a community, be it restricted or not, in order to better contextualise and prepare defense strategies against current and future cyber threats. These threats can themselves be defined as malicious acts that affect the confidentiality, integrity, or availability of computer systems, aiming to steal, damage or disrupt their digital wellbeing.

As such, CTI has quickly become a key component of contemporary cybersecurity, providing structured insights on both potential and active threats. The main goal is to enable individual users and organisations to communicate and collaboratively exchange details about various threats, adversaries, malicious systems, and a wide range of related issues. Frameworks such as Structured Threat Information eXpression (STIX) (Barnum, 2012) and Trusted Automated eXchange of Indicator Information (TAXII) (Connolly et al., 2014) illustrate the potential for standardisation and dissemination of Cyber Threat Intelligence across entities.

STIX is designed to represent and share CTI in a structured and consistent manner. Developed to facilitate the exchange of actionable information, STIX enables organisations to describe a wide range of

threat-related data. Its flexible design allows users to employ only the components relevant to their specific needs, promoting interoperability among various tools and systems. By providing a common language for CTI objects, STIX enhances organisations' ability to communicate, analyse, and respond to evolving cyber threats effectively.

TAXII plays a crucial role in augmenting STIX by acting as a transport protocol that ensures secure and efficient sharing of STIX-structured data. As a standardised protocol, TAXII promotes interoperability and cooperation across a wide array of systems and tools. It accommodates various modes of data exchange, including point-to-point sharing, hub-and-spoke architectures, and peer-to-peer networks. This versatility ensures that organisations can tailor data exchanges to meet their specific needs. Although not the only frameworks found for the standardisation and sharing of intelligence, these two, STIX and TAXII, are suitable examples of the crucial role CTI plays in the cybersecurity landscape. By automating the dissemination of threat intelligence, TAXII reduces the dependence on manual processes, enabling the timely and consistent delivery of actionable insights. This makes it easier for organisations to respond quickly to emerging threats and improve their overall security posture.

Security teams can take advantage from utilising CTI to foresee attacker Tactics, Techniques, and Procedures (TTP), recognise indicators of compromise, and strengthen their security defences against emerging threats. A prominent illustration of this advantage is displayed by the numerous Computer Security Incident Response Team (CSIRT) worldwide. Often, these teams consist of collaborations among several companies, aiming to enhance their overall cybersecurity framework. Moreover, certain nations, including Portugal[1], have established national CSIRTs to promote improved cyber environments and practices.

1.2 Problem Statement

Despite the significant potential of Cyber Threat Intelligence and the numerous efforts that have been made, it remains considerably underutilised within the broader scope. Current alert systems, including both Intrusion Detection System (IDS) and Security Information and Event Management (SIEM) tools, frequently do not integrate CTI sources into their detection mechanisms or are somewhat restricted to utilising trusted Indicator of Compromise (IOC) databases.

In addition, these systems frequently produce a substantial volume of alerts, a portion of which are inevitably false positives, thereby inundating analysts and diminishing their capacity to accurately identify and respond to genuine threats. Coupled with the absence of advanced capabilities to dynamically in-

[1] "Rede Nacional de CSIRT", Centro Nacional de Cibersegurança Portugal, https://www.cncs.gov.pt/pt/csirt/, Accessed: 2025-10-29

tegrate CTI and offer contextual insights, organisations may encounter considerable deficiencies in their security posture that could otherwise be mitigated.

This limited adoption of current CTI becomes even more apparent when examining the ongoing efforts to address the emerging threats. Recently, there has been a notable emphasis within the scientific community on leveraging the capabilities of Artificial Intelligence (AI) to assist in identifying both established and emerging threats. Although these efforts are commendable, it is important to recognize that AI-based systems face numerous challenges that remain to be thoroughly addressed, such as the scarcity of high-quality datasets for model training and evaluation and the applicablity of developed models to real world scenarios.

Furthermore, the traditional incident response method is inherently reactive, which means that a security alert is addressed only after its activation. Consequently, alerts must initially be governed by predetermined rules to enable their activation. Given that these rules are predominantly static, the integration of Cyber Threat Intelligence into such systems is rendered more challenging, though possible.

Comprehending the present condition of CTI requires acknowledging these core components within the cybersecurity domain. Although significant research efforts have been dedicated to both the standardisation and dissemination of Cyber Threat Intelligence, the comprehensive benefits that its application can provide are yet to be fully realised. When effectively harnessed, CTI has the potential to enhance threat detection capabilities, aid incident response and contextualisation, and ultimately fortify an organisation's overall security posture.

Advancing towards a more robust cybersecurity environment depends on deciphering how relevant data can be identified, autonomously processed, and used to acquire credible and pertinent insights for both detection and response. This necessitates a new approach that transcends traditional methods, fostering new research and development efforts, shifting from perceiving CTI as a mostly supplementary resource to recognising it as a fundamental element in the proactive identification and mitigation of emerging cyber threats.

1.3 Objectives

This dissertation seeks to demonstrate the feasibility of an autonomous alerting system rooted in Cyber Threat Intelligence. The overarching aim is to close the divide between raw CTI data and usable insights by not only detecting threats using information gathered from various sources, but also providing context for improved clarity in the understanding and decision-making process.

To achieve its effectiveness, the system must dynamically filter, analyse, and provide context for Cyber Threat Intelligence. This process produces alerts that are not only precise, but also representative of the factors that led to the detection. By automating the ingestion and processing of CTI feeds and integrating them with local information, such as running processes and network connections, the system provides organisations and individuals with a powerful tool to adeptly detect both current and emerging threats.

The objective of this dissertation extends beyond developing a theoretical framework, focussing on the implementation of a functional solution that merges staticly defined data collectors with dynamic detection and contextual analysis to enhance alert context and relevance. Its goal is to create a reproducible and adaptable system that can assess the feasibility of CTI-driven alerting. Taking this approach not only allows for the development of a system that enhances the overall cybersecurity posture of organisations, but also underscores the significance of Cyber Threat Intelligence.

The fundamental goal is to address the usage of CTI as a central component in the detection of threats, equipping organisations with a framework that enhances their defensive potential while simultaneously progressing the research aspects of CTI in the wider field of cybersecurity. Concurrently, it is crucial to clearly identify the insights gained regarding the potential of CTI, exploring how it can be used in proactive and reactive strategies.

1.4 Contributions of this Dissertation

Through this work, the main objective is to make a significant contribution to the cybersecurity domain by addressing the crucial gaps in the usage of CTI while providing the basis for a scalable, efficient, and effective solution. The main contributions are as follows:

- **A conceptual and technical architecture for CTI-driven detection**

  The dissertation proposes and implements a modular architecture for a CTI-driven detection system that autonomously ingestes, normalises, and correlates data from both external feeds and the monitored environment. This contribution establishes a design blueprint for transforming Cyber Threat Intelligence from a mostly passive source of intelligence into an active operational component of the detection pipeline.

- **A complement to existing security infrastructures**

  The proposed system complements, rather than replaces, existing tools such as SIEM or IDSs platforms. Its focus on interoperability through the use of standards (e.g., STIX) and modular components allows for future integration into existing cybersecurity infrastructures, allowing its capabilities

to be evaluated in realistic settings, without impacting current processes.

- **Hybrid automation framework balancing autonomy and analyst oversight**

  The solution introduces a balance between automation and human oversight by incorporating a hybrid framework that maintains analysts as the main decision makers, while automating processes such as CTI triage and correlation. This structure contributes to a practical detection framework that enhances repetitive tasks efficiency without removing human interpretability.

- **Empirical demonstration of feasibility and foundation for future research**

  Through controlled experimentation within a reproducible set of scenarios, the Proof of Concept (POC) validates that autonomous CTI-driven detection is technically feasible. The dissertation details the encountered challenges and design choices, providing a methodological foundation and clear roadmap for subsequent quantitative evaluation and scalability studies.

- **Contribution to the understanding and future operationalisation of CTI**

  Beyond the technical prototype, the dissertation contributes conceptual insight into the evolving role of CTI. The document illustrates the potential of CTI as both a operational and strategic asset in cybersecurity, advocating for its recognition within modern detection ecosystems. This work therefore establishes itself as a point of departure for further academic inquiry and practical implementations aiming to fully explore CTI's capabilities.

Collectively, these contributions bridge the gap between theoretical CTI research and practical, deployable implementations, advancing the discourse on CTI-driven cybersecurity practices and providing an academically grounded foundation for research in autonomous detection systems.

1.5 Document Layout

The document opens with a detailed introduction to the thesis in Chapter 1, followed by an overview of the entire work. This chapter establishes the research context, defines the objectives, and introduces the contributions made throughout this dissertation.

Chapter 2 presents an extensive examination of the current literature that forms the foundation of this research. The study of related work provides insight into existing initiatives and approaches, while also identifying less explored areas and persistent challenges that motivate the need for the proposed solution.

As the document unfolds, Chapter 3 details the core research hypothesis and the proposed approach to address it. It highlights the key functionalities of the system and the architectural principles that underpin

its development, forming the conceptual basis for subsequent implementation.

Building upon this design, Chapter 4 defines the technologies adopted and describes the final framework in detail. This chapter analyses each implemented service that composes the Proof of Concept, outlining their components and interactions to provide a clear overview of the complete system.

In Chapter 5, the experimental setup and testing methodology are presented. Each defined requirement is evaluated, demonstrating both the correctness of the developed system and its effectiveness in supporting the overarching hypothesis of the research.

The dissertation culminates in Chapter 6, which discusses the broader implications of the findings, reflecting on the results obtained and their importance to the overall goals. It reflects on the technical achievements, research contributions, and potential avenues that result from this work.

Finally, the dissertation concludes with a comprehensive assessment of the system and a summary of the contributions introduced to the cybersecurity domain. This closing chapter also reflects on the challenges encountered, the limitations of the current implementation, and potential directions for future research and development aimed at enhancing and extending this work.

Chapter 2

Related Work

The increasing complexity and sophistication of cyber threats require an ongoing evolution of cybersecurity strategies. Central to these strategies is CTI, which provides a systematic approach to understanding and addressing potential risks. CTI not only identifies the IOCs and TTPs of adversaries, but also empowers organisations to implement proactive measures through the sharing and analysis of threat data. Frameworks such as STIX and TAXII have been pivotal in standardising and disseminating threat intelligence, enabling interoperability across varied security systems.

This chapter delves into current practices and related work in the field of CTI, examining existing frameworks, tools, and methodologies that aim to improve cybersecurity capabilities. It also highlights the limitations of current approaches and identifies areas where further research and development are needed. By understanding the current landscape, this work seeks to lay the foundation for the development of the proposed alert system that bridges the gap between raw intelligence and actionable insights.

2.1 Background on Cyber Threat Intelligence

In the field of cybersecurity, threat intelligence refers to the collection, analysis, and dissemination of information about potential or active cyber threats. However, clearly defining CTI involves some complexities. In (Planque, 2017), the author proposes the following definition for CTI:

"The result of the process that combines information to create an overview of an adversary and their intent, tactics, techniques, and procedures."

This concept originates from an overview of not only the data being shared, but also the whole process depicted as the '*intelligence cycle*'. This cycle aims to divide threat intelligence into specific functions so that specific aspects can be individually improved, thus optimising the entire system. However, the parts proposed in this paper (Chismon and Ruks, 2015) are not strict divisions of the cycle. As the cyber

landscape evolves, the cycle itself needs to adapt to overcome new challenges and, above all, to be able to quickly help prevent new threats.

In Frini and Bourey-Brisset (2011), various approaches were studied in order to understand the different alternatives and how the intelligence cycle could be improved moving forward. Focussing mainly on the jump from relying on a single source of intelligence to an "*All-Source intelligence process*" [2]. At the time, the model, despite its potential, faced some challenges, namely due to the restriction on intelligence sharing between organisations and systems resulting from elements like:

- Private networks, repositories and formats make it harder to draw insights as fundamental pieces of information may not be available to all interested parties;
- The lack of effective communication coupled with the challenges associated with discovering and retrieving information, limit the interconnectivity of stakeholders.

Since then, efforts have been made to address these issues through the establishment of various formats and mechanisms designed to enhance the standardisation, sharing, and discovery processes associated with CTI.

2.2 Standardisation Formats

A critical aspect of the success of CTI is the adoption of standardised formats and frameworks that facilitate the organisation, representation, and sharing of threat intelligence. These guidelines, while crucial, tend to vary depending on their objectives, from very domain-specific data formats to attempts of representing the various aspects that compose the wider field.

The variety in formats is often rooted in the inherent nature of threat intelligence, which can originate from various sources, such as alerts, logs, and network events. Although some of these data conform to standardised formats like the Common Event Format (CEF) (ArcSight, 2009) and the Common Log Format (CLF) (Consortium, 1995), deviations are still common. Both CEF and CLF represent early efforts to standardise cyber information between different systems and organisations. However, as time has passed, sources have increasingly relied on variations, reflecting the evolving complexity and diversity of threat intelligence and systems.

[2] "All-source intelligence is the products, organisations, and activities that incorporate all sources of information and intelligence, including Open Source Intelligence (OSINT), in the production of intelligence. All-source intelligence is both a separate intelligence discipline and the name of the process used to produce intelligence from multiple intelligence or information sources. (US Army doctrine for intelligence)", Frini and Bourey-Brisset (2011)

In the article Ramsdale et al. (2020), the diversity and adoption of formats and standards in the realm of CTI is explored. This analysis highlights the difference between the CTI obtained from different alert systems and information feeds, clearly identifying the formats used. From the study, it can be observed that the adoption of these standards has been inconsistent, with many organisations opting for custom or legacy formats that better fit their operational needs, ranging from simple blocklists to more complex representations of IOCs.

Among the most widely used standards, the Structured Threat Information eXpression (STIX) framework (Barnum, 2012) stands out as one of the most complex, yet flexible, and robust languages for representing threat-related information. Developed to provide a common vocabulary to describe cyber threats, STIX enables organisations to share information on indicators, observed attack patterns, threat actors, and response strategies. For example, researchers have used STIX to model real scenarios without compromising data privacy (Sadique et al., 2018a). This aspect is vital because it seeks to establish its efficacy in facilitating secure and reliable information sharing.

Furthermore, the modular design behind this standard allows users to adopt only the components that are relevant to their needs, fostering interoperability between tools and systems. Figure 1[3] illustrates an example of how the STIX format effectively represents the association between a *file hash* and its connection to a specific malware.

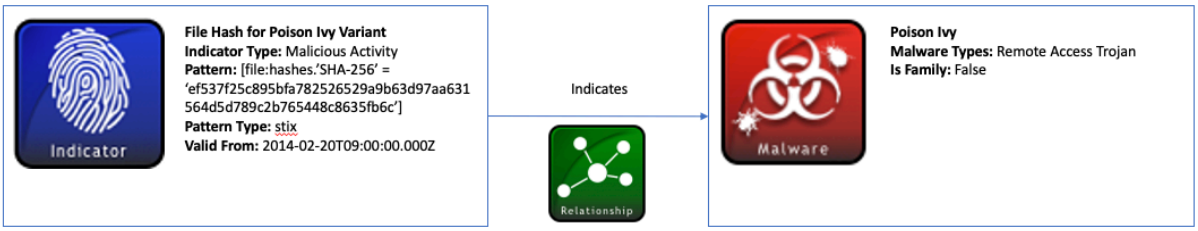


Figure 1: Malware Indicator for File Hash

The aforementioned relationship can be effectively characterised by a fundamental rule in the logical format "*If A, then B.*" Specifically, in the event that the hash is identified within a given system, an alert should be issued to indicate the potential presence of malware within that system. However, the area in which STIX truly excels is in its ability to depict intricate relationships that can encompass more than just a pair of objects.

[3] "Malware Indicator for File Hash",OASIS Open,https://oasis-open.github.io/cti-documentation/examples/malware-indicator-for-file-hash, Accessed: 2025-04-05.

In Figure 2, OASIS Open[4] demonstrates a graph generated using Graphviz[5], depicting all of the possible possible relationships between the various SDOs defined within the STIX standard.

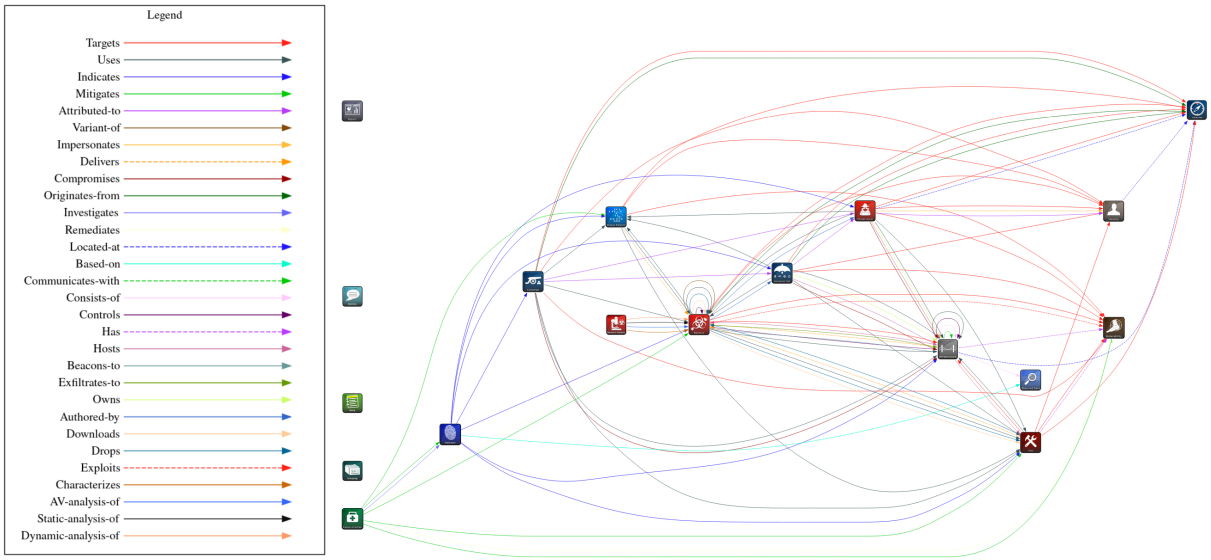


Figure 2: Visualized STIX Domain Object (SDO) Relationships (Edited for clarity)

This adaptability has allowed the development of various frameworks for the generation of rich CTI (Iqbal et al., 2018) (Marchiori et al., 2023), the enrichment of existing information (Chen et al., 2024) and its integration in real-world scenarios (Czekster et al., 2022).

Being in version 2.1 (Jordan et al., 2021), an import addition that was made to the STIX standard was the inclusion of Cyber Observable eXpression (CyBOX) (Barnum et al., 2012). Originally developed as a standalone language, CyBOX is now integrated into STIX and focusses on describing the observable properties of threats, such as file hashes, IP addresses, and domain names. By allowing granular descriptions of technical artefacts, CyBOX provides a foundation for detailed analysis and correlation of cyber events. Similarly, the Common Vulnerability Reporting Framework (CVRF) (Hagen, 2017) and its successor, the Common Security Advisory Framework (CSAF) (Rock et al., 2024), are specialised standards used to describe vulnerabilities and security advisories in a structured and machine-readable format, enabling efficient dissemination and integration of these data points into organisational workflows.

In summary, a variety of frameworks and standards are instrumental in structuring and enabling the exchange of CTI. General purpose standards such as STIX, characterised by its modular architecture, offer tools for the characterisation and detailing of a wide array of threat-related information, ranging from technical artefacts to strategic insights. This methodology provides the required flexibility and specificity

4 "Visualized SDO Relationships",OASIS Open,https://oasis-open.github.io/cti-documentation/img/relationships/SDO_Relationships_Graphviz.svg, Accessed: 2025-10-29.

5 Graphviz, The Graphviz Authors, https://graphviz.org/, Accessed: 2025-04-05.

inherent to CTI. In contrast, domain-specific frameworks like CSAF focus on presenting efficient formats, meticulously crafted to excel in domain-specific tasks. These frameworks underscore the equilibrium between flexibility, detail, and standardisation, thereby facilitating more effective and interoperable practices for information sharing across the cybersecurity domain.

2.3 Information Sharing Platforms and Channels

Effective information sharing is a cornerstone for CTI, enabling organisations to collaboratively identify and prevent cyber threats by exchanging insights, IOCs, and tactical data. Information sharing frameworks aim to bridge gaps between stakeholders by fostering interoperability, standardisation, and secure communication. These frameworks ensure that CTI can be shared between various systems and organisations, allowing faster detection and response to current and emerging threats.

The Trusted Automated eXchange of Indicator Information (TAXII) standard provides the mechanisms to share and consume STIX information seamlessly between entities, enhancing their collective defence capabilities. This framework comprises a set of technical specifications that aim to standardise the secure exchange of CTI across organisational and technological boundaries. Its primary objective is to enable the automated dissemination of actionable threat indicators, thereby empowering stakeholders to collaborate efficiently and respond to threats in real-time.

TAXII by design enables multiple exchange models, including peertopeer, hubandspoke, and publish - subscribe configurations. In TAXII's introductory webpage, (Open, 2025), the Figure 3 is used to exemplify the two models used in this process: Collections and Channels.

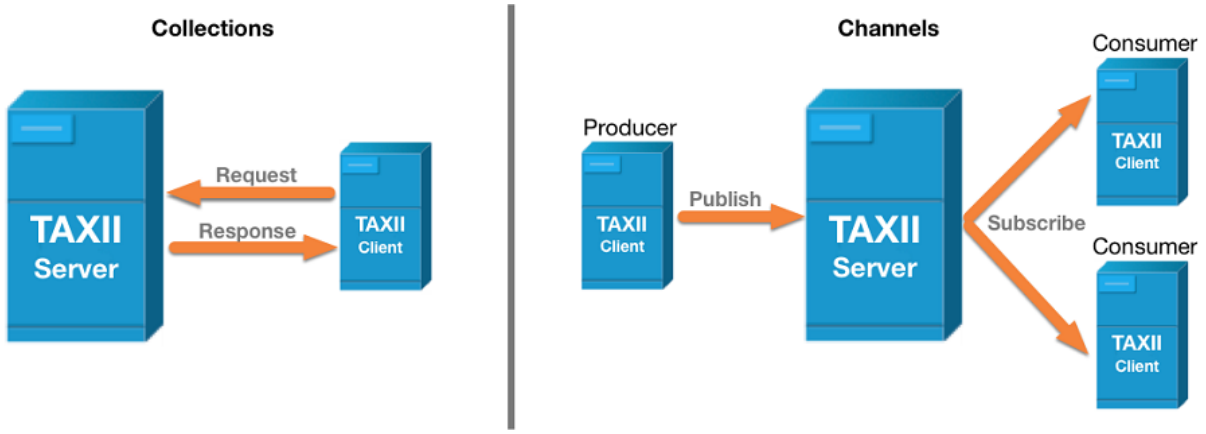


Figure 3: Overview of TAXII exchange models (Open, 2025)

A *Collection* serves as a repository for CTI objects that are maintained on a TAXII server. This setup allows to effectively store and disseminate trusted CTI data with users able to request these data using a

common request-response model. In contrast, a *Channel* operates on a publish-subscribe model, facilitating the distribution of data from sources to multiple subscribed consumers while also allowing consumers to obtain data from a variety of producers. This flexibility ensures the ability to accommodate diverse operational requirements. Using well-established protocols such as HTTPS for secure communication, TAXII guarantees the integrity and authenticity of the data exchanged.

Although the TAXII standard establishes a structured framework for the exchange of CTI, it also has limitations. While TAXII accommodates various exchange models, its efficacy is largely dependent on the trustworthiness and reliability of the data providers. Any security breach or failure by a crucial provider can cause a ripple effect across the consumer network, which could result in the spread of incorrect or harmful information. This dependency on a select group of trusted sources presents issues regarding the system's overall resilience.

Alongside the TAXII standard, Kampanakis (2014) reinforces the value of CTI sharing and automation by presenting multiple threat sharing platforms. These include initiatives such as the Malware Information Sharing Platform (MISP) (The MISP Project, 2025) and the Open Threat Exchange (OTX) (LevelBlue, 2025). These platforms arise to further facilitate the discovery and sharing of CTI providing both feeds and databases.

MISP is an open source threat intelligence platform designed to improve the collection, storage, and distribution of security indicators such as malware samples, phishing URLs, and IOCs (Wagner et al., 2016). It enables organisations to create and share rich, structured data while supporting automation through clear taxonomies. Another prominent initiative is AlienVault OTX, a community-driven platform that enables stakeholders to share and consume threat intelligence in real-time.

Unlike more formalised frameworks, OTX allows a collaborative approach, encouraging users to share IOCs, threat indicators, and attack methodologies with other participants. Consequently, users are required to make an additional effort to discern both valuable information and reliable sources. This then raises questions on the suitability of public feeds' data when integrating them into cybersecurity workflows, an issue that is deeper analysed by Allegretta et al. (2023).

Both MISP and OTX present challenges similar to those of the TAXII feeds previously discussed. The open source nature of MISP can result in discrepancies in data quality, as contributors offer varying levels of detail and accuracy when it comes to shared data. This variability can complicate the task of distinguishing irrelevant information from valuable intelligence, potentially inundating analysts with unrelated data. Similarly, the community-driven model of OTX, while promoting collaboration, raises concerns regarding the credibility and reliability of the shared data.

In addition to larger platforms and frameworks, smaller domain-specific standards play a crucial role in the current CTI landscape by focussing on specific aspects of cybersecurity. For example, the Common Vulnerabilities and Exposures (CVE) system provides a standardised identifier for publicly known vulnerabilities, allowing users to easily refer and share information on specific security flaws (Martin et al., 2002). Likewise, the Common Vulnerability Scoring System (CVSS) complements CVE by offering a consistent methodology to assess the severity of vulnerabilities, allowing organisations to prioritise and remediate security risks (Wirtz and Heisel, 2019).

These and other standards are lightweight and often serve as foundational building blocks for larger platforms, ensuring that even highly specific threat data can be seamlessly integrated into broader CTI ecosystems. By addressing niche areas, these domain-specific standards improve the precision and granularity of threat intelligence, contributing to a more comprehensive cybersecurity posture. Meanwhile, STIX aims to help automate the sharing of CTI within the larger ecosystem. Platforms like MISP and AlienVault OTX provide a middle term between global automation of CTI and domain-specific public intelligence.

2.4 The Role of CTI in Incident Response

The integration of CTI into incident response workflows significantly improves an organisation's ability to detect, analyse, and mitigate threats. By providing real-time information on emerging risks, CTI enables faster and more accurate responses, helping minimise potential damage and reduce recovery time.

Schlette et al. (2021b) emphasise its importance in incident response, arguing that it plays a critical role in improving decision making and fostering collaboration among stakeholders. Real-time intelligence ensures that incident response teams are equipped with up-to-date information on IOCs and adversary TTPs enabling them to address threats more effectively. However, integration of CTI into incident response requires tools and frameworks that support real-time processing and seamless collaboration.

In order to evaluate the various systems, Schlette et al. (2021b), also propose a comparative framework for this purpose. It focusses on their ability to share intelligence efficiently and facilitate timely responses to threats. This highlights the importance of interoperability, accentuating that organisations must adopt standardised protocols and formats to ensure that CTI can be effectively shared and applied for incident response.

Schlette et al. (2021c) further emphasises the critical role that CTI plays in improving the maturity and effectiveness of a Security Operation Center (SOC) and incident response capabilities through their CTI-SOC2M2 model. The model facilitates the integration of CTI with SOCs by employing a maturity framework

of capabilities to assess the extent of intelligence-driven operations. A notable aspect of the CTI-SOC2M2 framework is its methodical approach to integrate CTI formats, such as STIX and OpenIOC, with SOC services, including threat detection, vulnerability management, and incident analysis. A core contribution of the model is its emphasis on CTI's proactive role in each stage of the incident response lifecycle. Using a tiered capability maturity framework, this model effectively illustrates the systematic integration of CTI into the organisational processes of preparation, detection, and containment. This methodology promotes collaborative efforts and enables SOCs to access a more extensive array of intelligence resources while assessing their own methodologies related to the incident response process.

The research carried out by He et al. (2022) underscores the substantial influence of threat intelligence in informing and improving the capabilities to respond to malware incidents. By incorporating CTI into the incident response framework established by National Institute of Standards and Technology (NIST), the proposed methodology effectively integrates threat intelligence throughout the critical stages of preparation, detection, containment, eradication, recovery, and post-incident analysis. Furthermore, the cryptojacking case study illustrates the practical implementation of this CTI-enhanced approach, demonstrating its ability to reduce response times, limit organisational impact, and improve overall resilience against evolving threats. This methodology then attempts to continuously improve the defence mechanisms while fostering collaboration amongst stakeholders.

In addition to proposed frameworks from the scientific community, incident response platforms have been evolving toward automatic incorporation of threat intelligence. Tools such as OpenCTI (Filigran, 2025) and TheHive (StrangeBee, 2025) attempt to enhance the incident response capabilities of stakeholders by integrating CTI directly into their workflows. TheHive functions as a comprehensive Security Case Management Platform, providing SOCs, CSIRTs, and Computer Emergency Response Team (CERT) with the ability to handle incidents with increased efficiency. The platform automates the aggregation of alerts sourced from a variety of security tools. Subsequently, through the utilisation of more than 200 integrated analysers, it enables the rapid identification of malicious observables, thereby augmenting incident cases with pertinent threat intelligence. This integration supports real-time collaboration between teams, allowing them to derive advantages from CTI in their decision-making processes.

On the other hand, OpenCTI focusses on providing a structured approach to manage intelligence. Designed with CTI as one of its core priorities, the platform architecture presents a framework that facilitates incident analysis and dissemination of threat intelligence.

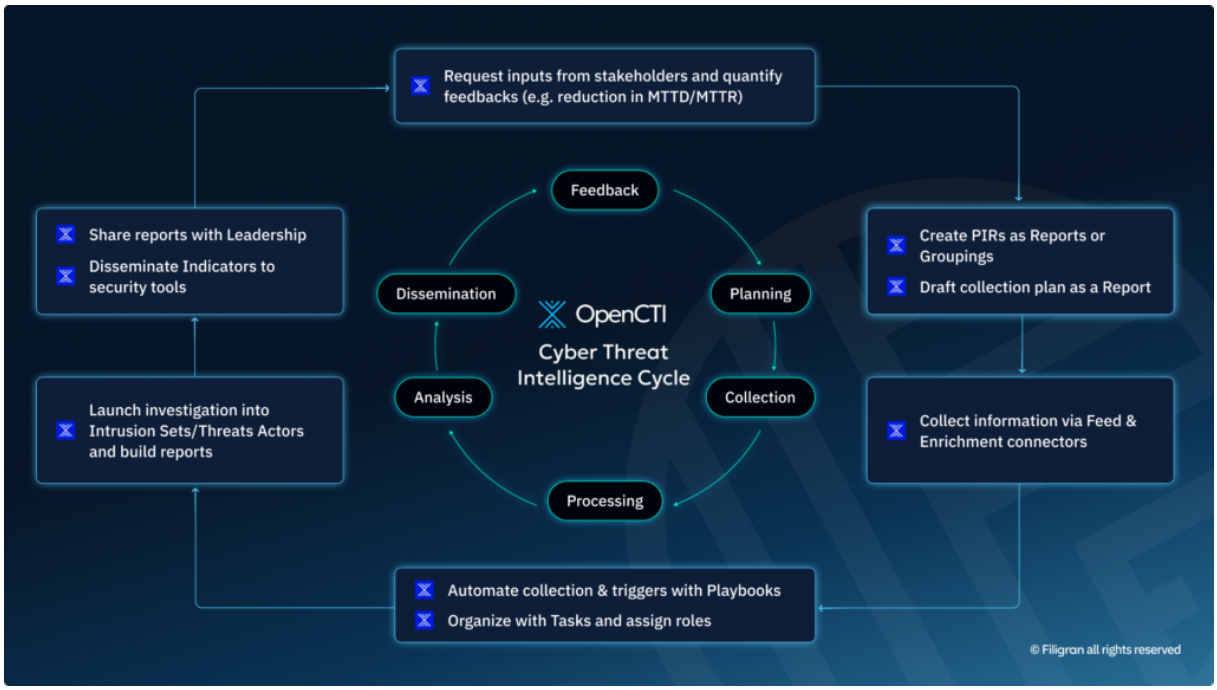


Figure 4: Cyber Threat Intelligence Lifecycle in OpenCTI Filigran (2024)

As depicted in Figure 4 and further elaborated in the associated post by Filigran[6], the features of OpenCTI are underscored for their importance in the multiple stages that constitute the CTI lifecycle, ranging from the planning phase to the feedback phase. It allows organisations to collect, analyse, and share threat data effectively, which can be directly fed into incident response processes. By utilising the MITRE ATT&CK framework and integrating with platforms like MISP and AlienVault OTX, OpenCTI ensures that threat intelligence is not only accessible but also relevant to the threat being analysed.

Together, these platforms and frameworks enable organisations to respond to incidents with greater speed and accuracy, ultimately aiming to improve their overall security posture. The synergy between incident response and threat intelligence fosters a proactive security environment, enabling teams to anticipate threats and mitigate risks before they escalate. However, in Section 2.6 the challenges associated with the effective application of these strategies will be addressed.

2.5 Current Challenges

In cybersecurity, the challenges faced tend to shift and evolve rapidly. This creates the need for prompt detection and response to new threats that may affect the monitored environment. Articles show that in recent years the amount of published CVE's is increasing rapidly, with an increase of 38% from 2023 to

6 "Understanding the cyber threat intelligence lifecycle", Filigran, https://filigran.io/ understanding-cyber-threat-intelligence-lifecycle/, Accessed: 2025-01-27.

2024 (YesWeHack, 2025).

This progression can be further elucidated by examining the weekly publication frequency of vulnerabilities. Figure 5 illustrates this upward trend as depicted by CVE.ICU. It is noteworthy that, despite initial spikes during the early 2000s, there has been a marked increase in weekly growth post-2015.

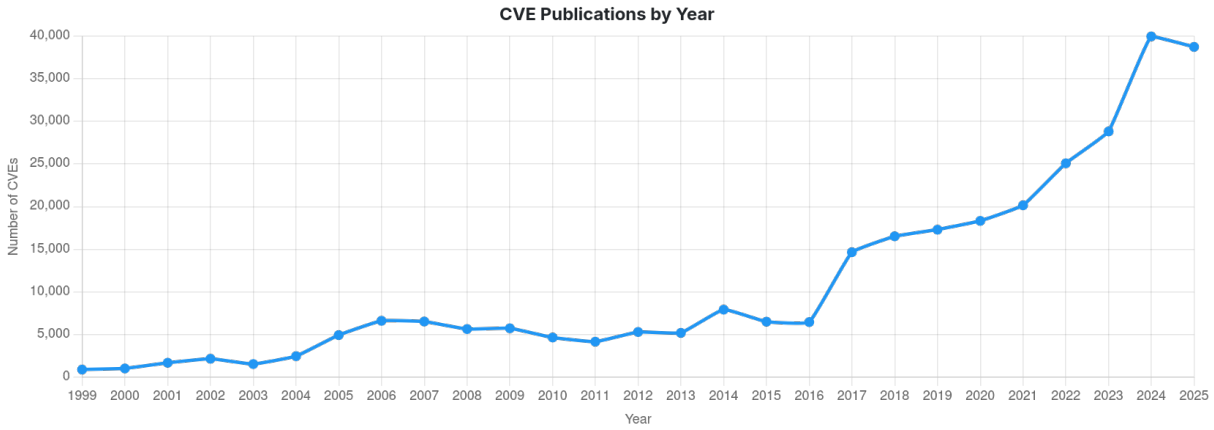


Figure 5: CVE Publications by Year, CVE.ICU (2025)

The more significant issue, however, hinges on undisclosed or unpublished vulnerabilities, often referred to as a Zero-Day Vulnerabilities or, in the short form, 0-Day. As defined by IBM, this type of '*exploit is a cyberattack vector that takes advantage of an unknown or unaddressed security flaw in computer software, hardware or firmware. "Zero day" refers to the fact that the software or device vendor has zero days to fix the flaw because malicious actors can already use it to access vulnerable systems.*'.

These threats are difficult to identify, mainly due to insufficient information. Moreover, even after their publication, the dissemination of mitigating measures requires significant time and effort, thus potentially prolonging the period of exposure. Bilge and Dumitraş (2012) discuss this issue in their paper concerning 0-Day attacks, providing an illustrative example timeline to demonstrate this concept, Figure 6, underscoring that not all phases occur necessarily in the order presented.

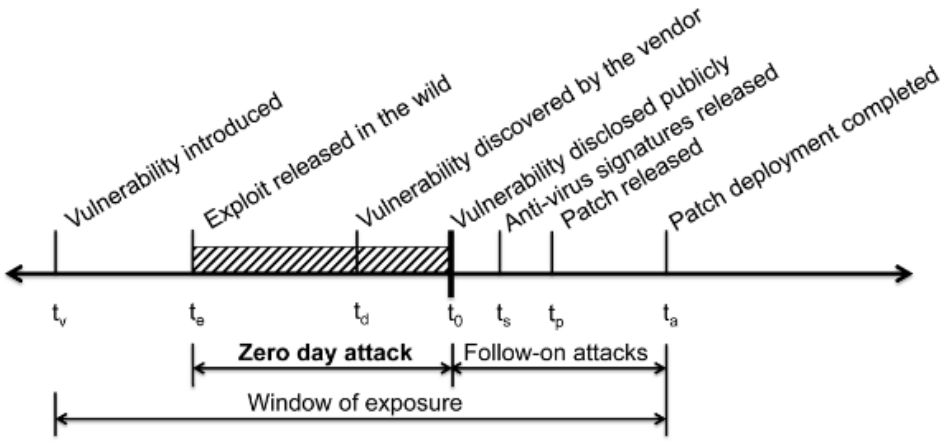


Figure 6: Attack Timeline, Bilge and Dumitraş (2012)

To give a better understanding of this timeline, it is essential to grasp how long do these attacks truly last. In the book "*Zero days, thousands of nights: The life and times of zero-day vulnerabilities and their exploits*", Ablon and Bogart (2017) explore the lifecycle of these attacks. In the research conducted they conclude two very important points:

- Exploits tend to have an average life expectancy of almost 7 years ("*specifically, 2,521 days*" according to Ablon and Bogart (2017));
- For the majority of vulnerabilities, once they are identified, it takes less than a month to develop a fully working exploit.

Further supporting these points, cybermindr (2025) claims that the average Time-to-Exploit (TTE) was reducded from the previously acknowledged 32 days, to just 5 days in 2024. While these revelations underscore the significance of a prompt response, they raise the question of how to address threats that remain undisclosed to the general public. In recent times, the number one attempt to answer this question has been Artificial Intelligence (AI)/Machine Learning (ML), with multiple researchers proposing frameworks to identify these threats.

Zoppi et al. (2021a) conducted a comprehensive study on various unsupervised algorithms for the detection of Zero-Day Vulnerabilities that showed promising results. Among the approaches studied, the ones that yielded the best results relied on meta-learning techniques, a topic discussed in-depth in the paper (Zoppi et al., 2021b).

In alignment with these initiatives, Saurabh et al. (2025) pursued a hybrid methodology that integrates the use of threat intelligence with unsupervised learning techniques to develop a "Hybrid IDS". The system in question, while it claims extraordinary results, requires further rigorous testing due to the ambiguity

surrounding its effectiveness in detecting Zero-Day Vulnerabilities. In addition to the limitations posed by the datasets, the authors frequently conflated 0-Day with unknown attacks in their findings. Although not entirely different, the notion of an unknown attack is broad and could represent anything from the 0-Day exploits to a simple variation when exploiting known vulnerabilities.

Another study by Guo (2023) took a broader approach, comparing the various efforts in the ML field to detect 0-Day exploits. This study identified that, while most detection methods present favourable results, most of them either had varying accuracy or limited test cases. This does not undermine the efforts made in enhancing detection capabilities. The paper rather emphasises the necessity for "Multi-front efforts [...] to address the challenges in designing effective ML-based zero-day attack detection schemes".

Even after the previously identified difficulties have been addressed, the challenges associated with AI driven approaches persist at a deeper level. As Bécue et al. (2021) categorises them into three areas:

- *Technical Challenges*: For instance, the necessity to initially collect and store substantial volumes of data, along with the intricate nature of developing and testing these systems;
- *Operational Challenges*: These encompass the substantial financial and risk-related implications inherent in AI systems, the expertise requisite for deploying such solutions, and the necessity to adhere to regulatory requirements;
- *Security Challenges*: The unpredictability of AI systems poses notable security concerns, contributing to uncertainty in anomaly detection. Additionally, there exists a risk associated with the possibility that adversaries may exploit inherent vulnerabilities within ML models themselves.

Taken together, these factors underscore the need for a solution that is rapid in action and stable over time, thus capable of effectively addressing continuously emerging threats. However, the proposed solution may diverge from those described in this chapter, as although AI may foster innovation in the detection of anomalous activities, it simultaneously introduces greater instability and technical challenges that can impede immediate progress.

2.6 Research Gaps

Despite advances in CTI, several challenges persist, preventing its full potential in cybersecurity. One of the most promising, yet underutilised, developments is the shift from traditional atomic IOCs to behavioural IOCs, which offer more durable and actionable intelligence. Villalón-Huerta et al. (2022) highlight that current standards, such as STIX, lack the necessary capabilities to effectively detect, represent and

share behavioural IOCs, leaving this valuable resource largely untapped. Addressing these gaps requires enhanced frameworks and methodologies that can integrate behaviour-orientated indicators into security workflows, providing a more comprehensive and proactive approach to threat detection.

In the article, the proposed alternative is based on SIGMA (SigmaHQ, 2025), a rule-based framework designed to share and detect dangerous behaviour by integrating with multiple SIEMs. However, it also acknowledges its flaws, such as the reduced adaptability and the lack of broader acceptance by the community. The latter point is the biggest barrier to CTI. Being already highly dependent on the interchange of information, systems that have more niche use cases/communities become less suitable for a common standard that is used across the larger cybersecurity landscape.

Another pressing issue, independent of the formats used during the sharing process, is the lack of robust methods to assess the quality of CTI. Schlette et al. (2021a) argue that poor quality CTI, often marked by inaccuracies, redundancies, and outdated information, diminishes its effectiveness in operational contexts. They propose quality assessment models that incorporate metrics such as timeliness, relevancy, and consistency. However, the integration of these models into real-world systems remains limited, which calls for more research into practical implementations that can further improve the reliability and trustworthiness of CTI, as well as how it can be better measured, in organisational scenarios.

The conceptual clarity and standardisation themselves are also areas that require significant improvement. In the article by Sahrom Abu et al. (2018), inconsistencies are identified in the definitions and frameworks used between organisations, reducing effective communication and collaboration. They emphasise the importance of qualified analysts who can transform raw threat data into actionable intelligence, bridging the gap between data collection and operational application. Standardised definitions and training courses are essential to address this fundamental issue, ensuring that CTI is understood and used consistently, not only by security systems but also adopted by professionals in the field.

Even then, Wagner et al. (2019) emphasise the need for automated mechanisms for CTI as automation and privacy in CTI sharing and processing are also current critical challenges. This effort aims to reduce the dependency on manual processes and improve efficiency. In terms of fostering trust in CTI sharing ecosystems, they highlight the importance of trust management systems and legal frameworks to enable effective collaboration between organisations. Without these mechanisms, the dissemination of sensitive threat data can raise concerns about data misuse or exposure to liabilities. Developing protocols that improve trust, ensure data security and promote transparency will be vital to advance CTI sharing and collaboration.

Moreover, the work of Sadique et al. (2018b) provides a comprehensive exploration of frameworks

designed for privacy-preserving CTI sharing. The emphasis is placed on developing mechanisms that effectively anonymize sensitive information, thereby preserving the utility of the shared data. This study highlights the critical importance of implementing systems capable of achieving a balanced integration of automation and rigorous privacy controls. Such systems are essential to ensure that shared intelligence adheres to both legal mandates and ethical guidelines, while also remaining practical and actionable for deployment by security teams.

Another significant research gap in CTI is the need for a more comprehensive understanding of the methodologies used in intelligence analysis. As highlighted in the article by Oosthoek and Doerr (2021), the field of CTI often lacks a structured analytical framework, leading to inconsistencies in the quality of the intelligence produced. Many analysts rely on ad-hoc approaches and borrowed terminologies from intelligence studies, which lack the necessary rigour for facilitating an effective analysis that can be readily shared and comprehended by others. This gap is particularly concerning given the rapid evolution of cyber threats, which require timely and accurate intelligence to inform defensive measures. The integration of established analysis methodologies, both manual and automated, could strengthen the analytical consistency of CTI and improve the overall effectiveness of threat detection and response strategies.

Finally, a key unexplored area of research is how Cyber Threat Intelligence may help in the fast mitigation of 0-Day exploits. The Section 2.5 highlights one of the biggest challenges in today's cybersecurity landscape, the surge of Zero-Day Vulnerabilities. While current efforts on detecting and mitigating this exploits exists, the focus relies on the employment of AI or ML models. In an ecosystem that facilitates the seamless sharing and dissemination of CTI, it raises an important question regarding the manner in which threat intelligence can be used to effectively address these security flaws while minimising the window of exposure.

Addressing these research gaps is essential to advance CTI as a key pillar of cybersecurity. By focussing on standardisation, automation, privacy, quality assurance, and trust management, the field can overcome its current limitations and benefit from the capabilities of CTI to improve timely detection and mitigation of threats.

2.7 Summary

This chapter examined the research and developments in Cyber Threat Intelligence, focussing on its foundations, the standardisation efforts behind frameworks such as STIX and TAXII, and the information-sharing platforms that enable collaborative cybersecurity (e.g., MISP, OTX). The review also addressed how CTI

is currently operationalised within current incident response practices. Through this exploration, it becomes clear that despite the current progress in structured data representation and exchange protocols, the benefits of utilising CTI to transform raw intelligence into actionable detection mechanisms remain under-explored.

Critically, the literature highlights a recurring gap between the availability of structured intelligence and its practical use in automated systems. Current tools often lack the mechanisms to dynamically identify and contextualise relevant threats to the specific environments they protect, with many solutions focussing on collection and sharing rather than dynamic correlation and contextualisation. This observation substantiates the need for solutions that can bridge data-sharing mechanisms and operational alerting systems effectively.

These insights directly motivate Chapter 3, which introduces a conceptual approach to address these identified gaps through the design of an autonomous CTI-driven alert system. The next chapter outlines the design principles and architecture proposed to make CTI a proactive, integrated component of detection workflows, leveraging both external intelligence and local context to generate timely and relevant alerts.

Chapter 3

Proposed Approach

The growth of cyber threats, coupled with the increasing complexity of attacks, has placed ever increasing demands on the cybersecurity community. As established in Chapter 2, current efforts aim to improve the integration of Cyber Threat Intelligence into existing cybersecurity frameworks, enhancing incident detection and response capabilities. While these proposals have made strides in it's application, they often face issues related to the trustworthiness of the intelligence, the relevance to specific organisational contexts, and the timeliness of its delivery. This not only limits its effectiveness but also introduces a overhead that adversaries can exploit. Consequently, there is a pressing need for approaches that can leverage CTI in a more autonomous, proactive, and resilient manner.

This chapter introduces the proposed approach, aimed at addressing this gap by presenting a robust system that can independently process CTI, correlate it with local intelligence, and automatically generate actionable security alerts. Its design emphasises three key principles: the integration of external and internal data sources, the automation of risk assessment and warning mechanisms, and the ability to respond to evolving threats in real-time in the cybersecurity landscape.

The chapter begins with a presentation of the research hypothesis that supports this work, followed by a detailed explanation of the conceptual architecture of the system. The core components of the system are described in depth, underlining the role of CTI in the overall threat detection process.

3.1 Research Hypotheses

Cyber Threat Intelligence standards (such as STIX,OpenIOC) and platforms (such as MISP,OpenCTI) have matured over the years, but are primarily used for information exchange and enrichment. Current research efforts often focus on empowering existing incident response strategies or up and coming AI models, rarely addressing the possibility of autonomous detection and alerting based on threat intelligence.

Existing solutions often focus on providing additional details and insight on existing security threats

and IOCs in order to aid human analysts contextualise security incidents. In addition, some systems use CTI in the detection process to improve data, but still rely on rules and human-defined thresholds to generate alerts. And recent advances have used AI and Machine Learning to create complex models that can identify patterns and anomalies, often requiring significant training data and with limited applicability in real-time detection of new and evolving threats.

All of these approaches, while improving the current state of cybersecurity, may underestimate the full potential of CTI. Due to its community-driven nature and real-time updates, CTI has the potential and purpose of adapting to new threats, not only to respond to known risks, but also to detect threats as they arise. This under-use forms the gap for innovation to be addressed in this dissertation.

The main goal of this research is that a system can be designed to process and correlate CTI autonomously with local data, generating actionable security alerts without human intervention. This system leverages existing standards and technologies, integrating them into a coherent architecture that automates the detection and alerting process. In doing so, this system focusses on the timeliness and relevance of threat detection, reduces the need for manual analysis and filtering of external intelligence, and enhances overall cybersecurity posture. This approach not only addresses the current challenges of CTI usage, but also sets the stage for more proactive and adaptive cybersecurity strategies.

To achieve this, the following three specific hypotheses are proposed.

- H_1: CTI can be autonomously ingested and correlated with local data sources without requiring manual intervention.
- H_2: The system can autonomously generate contextualised alerts solely based on external CTI and local data.
- H_3: The system can autonomously adapt to new threats by continuously updating its internal and external information.

These points provide a structure for the design and assessment of the system, ensuring that each main component contributes to the validation of the central hypothesis.

If these hypotheses are confirmed, the work presents the capability for CTI to move beyond its current role as a passive, advisory, and enriching tool and serving as an active driver of autonomous detection that works alongside existing detection mechanisms. By designing and implementing a proof of concept system, this research aims to operationalise CTI to generate alerts, rather than limiting its use for support of existing workflows. If validated, the significance of this contribution lies in its potential to reframe CTI

as a primary input for detection. Rather than replacing existing techniques, its versatility is highlighted as a complementary capability that strengthens proactive cybersecurity strategies.

To validate this hypothesis, this research involves designing and implementing a capable system, followed by a series of controlled tests to evaluate its ability to automatically detect threats and generate alerts only based on external CTI and collected local data. It is equally important to outline the scope and boundaries of this research. The system developed in this dissertation is a proof of the various concepts explored, not a production-ready deployment. The evaluation focusses on the feasibility of the proposed approach, assessing whether the system can integrate external threat information and correlate it with local data, adapt to previously unseen threats, and ultimately produce contextualised warnings. The scope of this demonstration, while ambitious, sets realistic expectations for what can be achieved within the constraints of this research. Its results are expected to provide a solid basis for future research and innovation in Cyber Threat Intelligence. They will help clarify the technical and operational challenges to be addressed to further explore CTI's potential.

3.2 Conceptual System Architecture

Having established the research hypothesis and defined the scope of this work, the next step is to outline the conceptual architecture of the proposed system. This architecture provides a high-level view of the system components and their interactions, serving as a bridge between the theoretical foundations and the practical implementation presented in the proof of concept. The design is guided directly by the research hypothesis, ensuring that each module contributes to demonstrating the feasibility of an autonomous CTI-driven alerting process. The system is structured as a modular and extensible framework that can accommodate future integration with existing security platforms and operational practices.

At its core, the architecture seeks to transform raw intelligence into actionable insights by combining structured CTI feeds, expressed in standards such as STIX, with locally collected network and system data. This integration enables the system not only to identify relevant IOCs, but also to enrich them with contextual information that increases their operational value. By providing analysts with meaningful and timely alerts, the architecture supports faster and more informed responses to emerging threats. Designed for adaptability, it emphasises dynamic adjustment to evolving adversarial tactics, positioning CTI as a proactive driver of detection rather than a passive source of enrichment.

The architecture achieves these goals through key components, including CTI feed parsers, local data collectors, a correlation and enrichment engine, and an alerting subsystem connected to dashboards and

reporting interfaces. These modules work together to create an end-to-end pipeline capable of autonomously ingesting, processing, and generating contextualised alerts.

To illustrate the overall flow of data and interactions among these components, Figure 7 presents a high-level overview of the system.

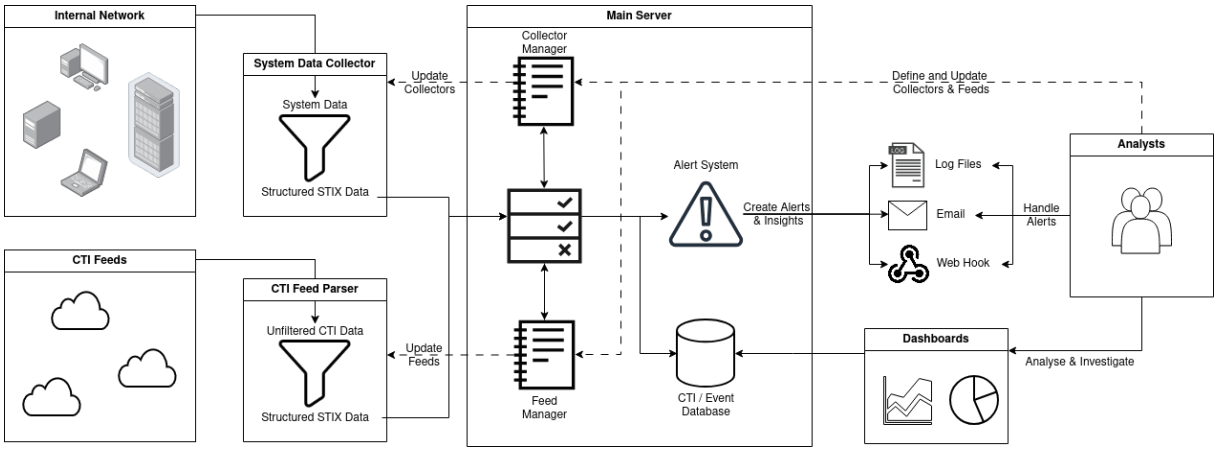


Figure 7: Conceptual system architecture and data flow.

Figure 7 shows how data from both external and internal sources is collected, normalised, and processed within the main server before being transformed into alerts and visualised on dashboards accessible to analysts. Each module is represented at a conceptual level, and its role in the alerting workflow is highlighted. The modular structure ensures that components can be extended or replaced as needed, supporting adaptability to different environments and integration with existing security infrastructures.

The remainder of this chapter examines these main components in greater detail. It begins with the main server, which represents the central logic of the system, before moving on to the data-ingestion processes. These include external feeds that supply structured threat intelligence and internal agents that provide local system and network data. The chapter then explores the alerting mechanisms that transform correlated data into actionable warnings. Finally, the chapter discusses the system's interfaces, namely the dashboards and reporting mechanisms, through which analysts interact with the generated alerts and explore the underlying data.

3.2.1 Core Processing and Coordination

At the centre of the proposed architecture lies the main server, which acts as the orchestration layer for all system operations. Its primary role is to ingest and correlate inputs from both internal and external sources, standardising them into a common representation (the STIX format) and generating alerts when the assessed risk exceeds the system's threshold. This limit is a predefined value that represents the

overall risk tolerance of the system, which is a balance between the sensitivity and volume of alerts. A low threshold may lead to an overwhelming number of alerts, many of which could be false positives, while a high threshold might result in missed detections of genuine threats.

Alternatively, a dynamic threshold approach can be employed, where the alerting sensitivity is adjusted based on contextual factors such as the current threat landscape or historical alert patterns. In this way, the system's responsiveness can automatically adapt to changing conditions, filtering out possible threats that it considers less relevant at a given time. However, this approach, if not carefully managed, can lead to inconsistencies in alert generation, increase in the rate of false negatives, or even exploitation by adversaries aware of the system's adaptive behaviour. Attackers could potentially flood the system with high volumes of low-risk indicators to manipulate the threshold upwards, thereby reducing the likelihood of detection for more severe threats.

Therefore, by default, the system employs a static threshold that balances sensitivity and specificity, ensuring consistent alerting behaviour. Nonetheless, as analysts become more familiar with the system's performance and the specific threat landscape they face, this threshold can be adjusted on the basis of the operational context and the organisation's risk tolerance.

Additionally, in order to build a system capable of handling both the volume and velocity of incoming data, while maintaining the integrity and relevance of the correlations produced, a robust processing pipeline is required. While a fully distributed architecture could be considered for scalability, it can introduce both significant complexity and latency, which may not be suitable for the Proof of Concept stage of this work. Instead, the proposed architecture opts for a centralized server that focuses on building a solid foundation for autonomous detection pipeline.

The pipeline model itself is chosen for its ability to break down complex processing tasks into manageable stages, each responsible for a specific function. Not only it allows for clarity and modularity in the system's design, but it also facilitates future enhancements, such as the integration of additional analytic modules or the replacement of existing components with more advanced alternatives. This is especially important considering the evolution of Machine Learning (ML) and behavioural analysis techniques, which could be incorporated into the pipeline to enhance detection capabilities.

For the input layers, the main server needs to processes the incoming CTI from various internal sources (also known as agents) and external feeds. This involves parsing and normalising the data into a consistent format, without the loss of critical information. For this purpose, the server employs two different approaches: for external feeds, it leverages existing standards such as STIX and ingests its data with little transformation due to the adoption of common standards (Section 3.2.2); for local data, it relies

on lightweight agents that collect and forward relevant information in a structured format that can be aligned with existing STIX Domain Objects (Section 3.2.3).

The main server also manages the generation of alerts, enforcing the logic that determines whether a given correlation between existing threats and local data warrants the creation of a contextualised warning (Section 3.2.4). This responsibility positions the server as both a processing hub and a decision-making engine. By decoupling ingestion and alerting into discrete modules, the architecture supports extensibility, enabling the inclusion of additional analytic services, such as reputation scoring or anomaly detection.

Finally, the server provides the means through which stakeholders can interact with the system and its alerts (Section 3.2.5). These alerts can be presented through dashboards, summarised in reports, or connected to automated response mechanisms such as SOAR platforms. In this way, the system extends beyond data processing and correlation to become a part of the broader security operations workflow.

3.2.2 External Threat Intelligence Ingestion

The external data ingestion module embodies the system's capacity to autonomously acquire and process CTI from heterogeneous sources. These sources, hereafter referred to as feeds, may include private community-driven repositories, industry information-sharing platforms, commercial threat feeds, or open-source initiatives. Each feed may vary significantly in structure, freshness, and reliability, making ingestion a critical step in ensuring that the intelligence is usable. To address this diversity, the module implements parsers and normalisation routines capable of converting disparate inputs into a unified representation, primarily based on widely adopted standards such as STIX. This ensures inter-operability between sources while minimising the loss of contextual information during transformation.

A major challenge in external CTI integration lies in the quality and reliability of the incoming data. Intelligence feeds may include redundant or outdated entries, and in some cases even maliciously injected IOCs intended to disrupt detection pipelines. To mitigate these risks, the ingestion module incorporates basic trust assessment mechanisms. Examples may include lightweight reputation scoring of sources, temporal validation of indicator freshness, and cross-checking across different feeds. Although the proof-of-concept system does not aim to replicate the sophistication of production-grade trust frameworks, the application of such mechanisms illustrates how even modest validation steps can significantly improve data integrity and downstream reliability.

By maintaining a continuous flow of validated intelligence, the system ensures that its knowledge base adapts dynamically to evolving threats. This capability reduces the time between the publication of new intelligence and its operational use, thus narrowing the window of exposure to emerging adversaries. More

importantly, it addresses the first of the research hypotheses (H_1): that CTI can be autonomously ingested and normalised without human intervention. In this way, the external ingestion module is not merely a data pipeline but a foundational enabler of autonomous, intelligence-driven detection.

In addition, the intelligence collected through this module also needs to have an associated level of risk or confidence. This is crucial for the alerting engine to prioritise and contextualise alerts effectively, as without such information, all indicators would be treated equally, potentially leading to alert fatigue or increases in false positives. Cyber Threat Intelligence, while often detailing threats, can contain additional observables that are not inherently malicious, rather merely suspicious or supporting evidence. Even among malicious indicators, the severity and risk of the threat can vary widely, from, for example, low-risk suspicious files to high-risk ransomware command-and-control servers. This level of nuance is essential for the system to generate timely alerts, assessing their severity and relevance.

The module therefore tags each ingested indicator with metadata reflecting its source and risk score, which can be leveraged in subsequent correlation and alerting processes. This tagging not only enhances the operational value of intelligence, but also greatly supports the third research hypothesis (H_3): that the system can autonomously adapt to new threats by continuously updating its internal and external information. This allows it to prioritise emerging intelligence based on its assessed risk and freshness, ensuring that the most critical threats are addressed promptly.

3.2.3 Local Data Collectors

While external CTI provides global awareness, internal agents provide the local context required to make that intelligence actionable and relevant to the monitored environment. These lightweight collectors are deployed within the internal network and collect telemetry such as existing files, network traffic, and system events. Their purpose is not to replicate the functionality of existing SIEMs, but to provide relevant information from local data to identify correlations with external CTI without human intervention.

Designing these agents involves balancing autonomy, performance, and security. Firstly, they must operate with minimal manual configuration to avoid introducing operational overhead, thus supporting the overarching goal of autonomous detection. Collectors must also be efficient, minimising their resource footprint to avoid impacting the performance of the host systems they monitor and ensuring scalability across potentially large and diverse environments. This is especially important since the field of cybersecurity often involves resource-constrained devices that cannot afford to employ heavy monitoring agents. Finally, security considerations are paramount. On one hand, the agents must be resilient against tampering or evasion by adversaries who may seek to exploit their presence. On the other, they must ensure

data integrity and preserve confidentiality, especially when dealing with sensitive organisational data.

In addition to these design principles, local collectors must also be able to normalise their output into a format compatible with the chosen SDO standards. This ensures that the data they provide can be seamlessly integrated with external CTI within the main server, facilitating effective correlation and analysis. This normalisation is crucial for achieving the first and second research hypothesis (H_1 and H_2), as it enables the system to autonomously correlate local events with external intelligence and generate alerts with contextualised data relevant to the monitored environment.

The collection of both local and external intelligence is central to the hypothesis. Without relevant local telemetry, external CTI remains abstract and disconnected from the operational environment. In contrast, without CTI, local data may lack enough global threat context to allow the system to autonomously detect threats. By bringing these sources together, the system enables the generation of alerts that are situationally relevant, reducing noise while increasing the likelihood of actionable outcomes.

3.2.4 Autonomous Alerting Engine

Having established the ingestion of both external CTI and local telemetry, the next critical component is the autonomous alerting engine. This module is responsible for correlating the normalised data streams and determining when an alert should be generated. It represents the core logic that transforms raw intelligence into operational insights, addressing the core of the second research hypothesis (H_2).

Once external CTI and local telemetry are normalised, they converge within this module, which applies correlation rules (Section 4.3.1) and risk assessments (Section 4.4.4) to identify and alert for potential threats. Unlike traditional detection mechanisms that are heavily dependent on human-defined thresholds or preconfigured rules, the alerting engine aims to demonstrate that CTI itself can drive the alerting process autonomously. For this to be possible, the engine accesses the vector connecting local agents to known threats, calculating the risk based on the severity of the threat and its connection to local events. If the calculated risk exceeds a predefined threshold, an alert is generated containing all the information related to the event.

Rather than simply flagging a match between local data collected and an indicator from a feed, the system enriches the alert with supporting information, such as the full vector that led to the detection, along with all the details of each SDO that composed it. This includes metadata such as the origin of CTI, the time of publication, and the assets affected within the local environment. This contextual layer increases operational value by reducing false positives and enabling analysts to quickly assess the severity and relevance of each alert.

The constant updates from both external feeds and local collectors ensure that the alerting engine operates with the most current information, allowing it to adapt dynamically. However, for it to be capable of distinguishing new threats, the system must incorporate mechanisms capable of identifying both previously reported alerts and the decay of the risk levels associated with older intelligence. This ensures that the system remains responsive to emerging threats while avoiding alert fatigue from repeated notifications about known issues. It also supports the third hypothesis (H_3), as the system continually refines its internal state based on the latest intelligence, thus adapting to the evolving threat landscape.

This approach also has implications when addressing complex and evolving adversaries such as an Advanced Persistent Threat (APT). These threat actors are typically characterised by their persistence, multi-stage tactics, and usage of custom tools that often evade traditional signature-based defences. Within the proposed architecture, the continuous ingestion and correlation of external and internal intelligence play a crucial role in improving visibility over such long-lived campaigns. As new indicators associated with an APT, such as new command and control servers or file artefacts, are published by intelligence providers, the system is capable of dynamically integrating and correlating them with local observables. All of this contributes to the ongoing detection and identification of potential APT activity within the monitored environment that may otherwise remain undetected by conventional rule sets.

Although the proof-of-concept system does not implement advanced Machine Learning or behavioural analysis techniques, the rise of such methods presents a promising avenue for future enhancements, supporting the previous point. By incorporating adaptive learning algorithms, the alerting engine could evolve to identify patterns and anomalies that extend beyond static correlations, further improving its ability to detect emerging threats autonomously. This potential evolution aligns with the larger goal of positioning CTI as a proactive detection driver, capable of adapting to the ever-changing threat landscape and helping security teams in their defensive strategies.

This engine demonstrates the feasibility of an autonomous alerting mechanism while maintaining a focus on practical applicability. Its integration into the broader architecture ensures that alerts are not generated in isolation but are part of a continuous feedback loop that incorporates the latest intelligence and local context. This dynamic adaptability is crucial for the possibility of the system's integration into real-world security operations as a complementary detection mechanism. Above all, this component underscores the system's commitment to leveraging CTI not just as a passive data source, but as an active driver of autonomous detection and alerting, thus supporting the core of the research hypothesis.

3.2.5 System Management and Interface

The final component of the system is the interface through which analysts interact with the system, its outputs, and existing intelligence. Dashboards and reporting mechanisms translate technical alerts and complex relationships between data into structured and accessible information. In the proposed design, visualisations highlight not only the raw indicators that triggered an alert, but also the context that led to the alert, thereby supporting decision-making. This ensures that the system's autonomous capabilities directly enhance, rather than overwhelm, human workflows and situational awareness.

This layer serves multiple purposes. Primarily, it provides a clear and organised view of the alerts generated by the system, allowing analysts to quickly assess and prioritise their responses. It also offers insights into the underlying data, allowing users to explore the relationships between local events and external CTI. This transparency is crucial to building trust in the autonomous operations of the system, as it allows analysts to understand the rationale behind each alert. It also provides a historical record of alerts, supporting post-incident analysis and continuous improvement of detection strategies.

Interacting with the system through dashboards allows an intuitive interface for analysts to validate alerts, ensuring their relevance. However, an underlying API layer allows for further integration with existing security tools and workflows. For example, the system could be connected to the Security Orchestration, Automation, and Response platforms, enabling automated responses to certain types of alerts. This could allow the system to not only generate alerts, but also trigger predefined actions within existing cybersecurity workflows, such as blocking malicious IP addresses or creating tickets in incident management systems, thereby extending its autonomous capabilities into the response phase of the security lifecycle.

Beyond immediate alerting, the management interface also supports the continuous evolution of the system. By providing insights on the current threat landscape and system data, analysts can identify gaps in coverage or areas for improvement. Analysts can improve, create, or delete the collectors and feeds to ensure emerging threats do not go unnoticed due to lack of data. This is a pivotal point because, while the system aims to autonomously detect threats, it recognises the value of human expertise in refining and guiding its operations over time. The contrast between automation and human oversight of the system is clearly drawn here. While the system autonomously collects data, processes it, and generates alerts, the human analyst remains in control of the overall intelligence that is collected and the final decision-making process regarding the alerts generated.

In general, this interface layer is not merely a passive display of information but an active component of the operational workflow of the system. It ensures that the autonomous capabilities of the system are effectively harnessed to enhance human decision making, thus closing the loop between data collection,

analysis, and response in a manner that is efficient and user-centric. This last component also highlights the practical applicability of the proposed architecture, demonstrating how autonomous CTI-driven detection can be integrated into existing security operations without disrupting established workflows.

3.3 Summary

This chapter presented the conceptual framework and system architecture designed to transform raw CTI from isolated data points into a valuable resource for dynamic detection, alerting and contextualisation within cybersecurity operations. It defined the core research hypothesis around the autonomous processing of CTI to generate actionable alerts, adapting to an organisation's specific threat landscape, a notion supported by the gaps identified in Chapter 2.

Deepening the understanding of the proposed approach, the chapter outlined the architectural components responsible for the various stages of data ingestion, processing, correlation, and alert generation. The proposed design emphasised modularity, interoperability through established standards, and a hybrid approach balancing automation with analyst oversight.

From a critical standpoint, the proposed architecture prioritises the feasibility of concept, supporting experimentation and validation of the research hypothesis, rather than aiming for immediate deployment and comparison to established systems at scale. It deliberately abstracts complex deployment challenges to focus on validating core concepts of the newly proposed system. The layered architecture also ensures flexibility, allowing future integration with existing security systems, such as SIEM or Security Orchestration, Automation, and Response (SOAR).

Building upon this conceptual foundation, Chapter 4 transitions from theory to implementation. It defines the main requirements and objectives for the practical implementation of the proposed system, guiding its development in alignment with the research goals. It details the Proof of Concept, where the conceptual architecture is implemented as a functioning system, describing technologies, design choices, and strategies used to construct the various components.

Chapter 4

Proof of Concept

This chapter presents the proof of concept developed to validate the feasibility of autonomous, CTI-driven alerting. Building on the conceptual architecture introduced in Chapter 3, the developed system implements its foundational components and demonstrates how they interact to transform raw intelligence and local information into actionable security insights.

The implementation prioritises modularity and future extensibility, allowing for easy integration of additional data sources, different collection methods, and correlation logic as needed. In particular, the design emphasises interoperability with existing CTI standards, efficient handling of local data streams, and the ability to adapt to new threat intelligence as it becomes available. The POC illustrates how the proposed architecture can operate and detect possible threats autonomously.

The chapter begins by presenting the design choices that refine the scope of the Proof of Concept, clarifying the core functional requirements to achieve the research objectives. Next, the tools and technologies used in the development phase are described, highlighting the selection of software libraries, frameworks, and data formats that enable interoperability with existing CTI standards and efficient processing of local data.

It continues with a detailed overview of the CTI Provider and Endpoint Agent components, explaining their roles in the overall architecture as the input layers for both external and internal data, respectively. These are explained in detail, outlining their implementation specifics, data formats, and interaction with the central server.

Finally, the core of the chapter is then dedicated to detailing the implementation of the Main Server, describing in depth the components developed and their interactions, and highlighting their role in the overall threat detection process. This section demonstrates how the conceptual architecture was translated into a working system, showcasing the practical challenges encountered and the solutions devised.

4.1 Design Choices

The creation and deployment of an alert system based on Cyber Threat Intelligence poses several challenges that must be resolved to guarantee the success of the project. These challenges range from technical choices to integration complexities and operational considerations. This section outlines the key design decisions made to focus the Proof of Concept on demonstrating the feasibility of autonomous, CTI-driven alerting while managing the inherent complexities of such a system. These choices focus on essential functionalities that validate each research hypothesis, deliberately defining the boundaries of the POC within the scope of this dissertation.

Given the goals of the project, it is possible to define clear objectives and functionalities for the POC, guiding both its development and evaluation. The proposed research hypotheses (Section 3.1) can be translated into specific and testable requirements that the system must meet to validate its overarching goals. These requirements are as follows:

- H_1 - Ingestion and Correlation:
  - R_1: The system must be capable of ingesting and normalising CTI from external sources;
  - R_2: The system must be able to collect and process local data from the monitored environment;
  - R_3: The system must be able to identify and correlate relevant SDOs from external and local intelligence.
- H_2 - Risk Assessment and Alerting:
  - R_4: The system must be able to assess the risk associated with correlated events;
  - R_5: The system must be able to prioritise alerts based on the assessed risk level;
  - R_6: The system must implement a decay mechanism to reduce the risk score of stale SDOs over time;
  - R_7: The system must provide details on the specific threat object that led to the generation of the alert, as well as all the context relating it to the affected agent, to aid in incident response.
- H_3 - Adaptability and Autonomy:
  - R_8: The system must be able to alert for new threats as they emerge in the CTI feeds without requiring manual updates;

- R_9: The system must be able to automatically enable or disable data collection based on the current threat landscape;

Following the establishment of these requirements, several key design choices were made to ensure that the Proof of Concept remains focused, manageable, and effective. Each decision aims to address the critical aspects of the research questions while creating a foundation for a robust technical implementation that is practical, testable, and replicable.

4.1.1 Data Standards

The first design choice concerns the data format used within the system and with the external intelligence sources. The system employs STIX as the primary format to represent threat intelligence, both for external feed ingestion and internal correlation processes. This decision is driven by the need for a structured, standardised format that facilitates interoperability and efficient data handling. As discussed in Section 2.2, the Structured Threat Information eXpression (STIX) framework stands out due to its flexibility, extensibility, and incremental adoption in the cybersecurity community.

By adopting it as the canonical representation for external intelligence, the system benefits from an established format that can be directly incorporated into today's security operations. While formats such as OpenIOC provide valuable capabilities for specific use cases, they lack the comprehensive structure and relational modelling offered by STIX. It also presents a good compromise between complexity and usability, providing sufficient detail to represent various aspects of the threat through its own STIX Domain Object (SDO)s and relationships, without overwhelming the system with the intricacies associated with the management of multiple complex formats.

This decision impacts the ingestion of both external and local data. For external CTI feeds, the system benefits from the availability of numerous public, community-driven and commercial sources that provide intelligence in the STIX format. However, it is acknowledged that not all sources will natively support the format, especially considering the need for local data collection. To address this, the system must be extensible enough to, in the future, incorporate parsers for the different providers, transforming them into STIX Domain Objects as needed. This decision will ensure that the system can adapt to a variety of data sources, improving its versatility and applicability in real-world scenarios.

By standardising its intelligence to the STIX format, the system can streamline its data processing pipeline, focussing on the core functionalities required to validate the research hypothesis. This process ensures that the system can efficiently ingest and normalise external intelligence into its internal representation without the need for complex transformations, thus directly addressing R_1.

As for the local data collection, systems provide information in various formats, often specialised to the context of the source. This can imply that, for example, in organisations with a wide range of operating systems and device types, the collected data may vary significantly in structure and detail. This diversity poses a challenge for both collection, normalisation, and correlation among heterogeneous data sources. To address this challenge, the local data collection is designed to collect information in a structured format that aligns with known SDOs where possible.

Addressing this requirement, the selected tool to collect local data was *osquery* (The Linux Foundation Projects, 2025), a lightweight open source endpoint that provides a SQL-like interface to query the system information. The tool abstracts the details of the underlying system, allowing it to retrieve data from various operating systems and their inner workings in a structured manner. Its flexibility and adoption of the SQL query language make it a suitable choice to collect relevant local data reliably, while simultaneously converting them into a structured format that can be more easily mapped to the expected SDOs.

By leveraging *osquery*'s capabilities, the Proof of Concept can collect relevant local data, such as process information, network connections, and file integrity, commonly used in threat detection. The collected data can then be transformed into the STIX format, aligning with the system's overall data representation strategy. This approach addresses R_2 and, when combined with external intelligence feeds, supports the general correlation mentioned in R_3 since both data sources will share a common format.

4.1.2 Programming Language

Another key design decision for the Proof of Concept was the selection of the programming language used to implement the system. The chosen language needs to balance several fundamental aspects of the POC's development:

- Robust support for cybersecurity and CTI-related libraries, particularly for handling the STIX format natively, as these standards underpin the system's data exchange;
- Seamless integration with external services and APIs, enabling the system to interact with heterogeneous CTI feeds and, in the future, other security tools;
- Suitability for prototyping while maintaining readability and extensibility, ensuring that the codebase can evolve to incorporate more complex functionalities and optimisations, laying the groundwork for future research and development;
- A mature ecosystem and active community, providing access to long-term support, reusable components, and a widespread knowledge base to facilitate adoption and future innovation.

Given these criteria, Python was selected as the implementation language for this project. As an established programming language, it benefits from widespread adoption in the broader programming community and offers mature, well-maintained libraries for parsing both common formats, such as JSON and XML, but also domain-specific libraries such as *stix2* (OASIS Open, 2025), which provide native support for the selected threat intelligence format. Python's extensive libraries and frameworks also facilitate rapid development of networked applications, making it easier to implement functionalities such as the communication between the server and clients, external data ingestion, and interacting with RESTful APIs.

Its emphasis on prototyping and modularity aligns directly with the proof-of-concept goals, where transparency and extensibility are as important as the base functionality. This is particularly relevant in a research context, where the ability to iterate and improve the system based on experimental findings is crucial. The goal of also making the system accessible to other researchers and practitioners further supports the choice of Python, as its readability and widespread use lower the barrier to entry for collaboration and future contributions.

Although languages such as C/C++ or Rust provide superior performance and concurrency handling, Python was prioritised because of its prototyping efficiency, ecosystem maturity, and accessibility to future stakeholders. Performance and scalability, though highly relevant for production-grade deployments, especially in potential resource-constrained environments, were considered future enhancements. These aspects fall outside the scope of this Proof of Concept and are explicitly recognised as opportunities for future optimisation.

4.1.3 Web Framework and API Layer

In addition to the choice of programming language, the design also required a framework to expose system functionalities in a structured and accessible way. This was essential to provide analysts with an interface to manage the platform, as well as to explore generated alerts and existing threat intelligence. Given the scope of the proof of concept and following what has been established in the previous section, the selected framework needed to prioritise simplicity, ease of use, and extensibility.

Flask, a Python-based web framework, was the selected tool to perform this role. It enables the system to expose a RESTful API layer through which intelligence, correlation results, and alerts can be queried, managed, or integrated with other tools. Its simplicity makes it particularly suitable for a POC context, while its flexibility ensures that additional routes, authentication layers, or middleware can be incorporated in future iterations to improve security and functionality.

Beyond the base API, Flask also provides the foundation for building a reliable web interface, allowing

for alerts and contextual information to be visualised directly without requiring external dashboards or third-party integrations. The web interface of the POC is focused on the ability to view and explore the system's CTI data and the generated alerts. This includes functionalities such as filtering, viewing the details of alerts, and accessing the underlying SDOs that contributed to each event.

This decision reflects the larger goal of ensuring that the system is not only capable of autonomous detection, but is also capable of providing its alerts in a usable and understandable format to aid analysts and incident responders in investigating potential threats. By leveraging Flask's capabilities, the system can maintain its modularity, separating its core logic and User Interface. Both the Web API and the User Interface (UI) are designed to be easily extendable, allowing more advanced features such as alert management, user roles, or integration with other security tools to be added in future work.

4.1.4 Main Components

The final step to translate the design requirements into a concrete Proof of Concept, is to define its structure and the roles of its components. This builds upon the conceptual framework (Section 3.2), bridging the gap between theory and the practical implementation. The developed system is structured around three fundamental components:

- **CTI Providers**: These sources emulate external CTI feeds, supplying structured STIX data through TAXII-like interfaces. Their inclusion enables controlled and repeatable testing of the CTI ingestion mechanisms of the system (H_1). By simulating dynamic external feeds, this component validates the system's ability to continuously update its intelligence base, adapting to new threats as they emerge (H_3).

- **Endpoint Agents**: Deployed on monitored hosts, the agents collect local observables such as running processes and active network connections. Their role is to collect local intelligence (H_1) and provide environmental context that can be correlated with external intelligence (H_2). The deployment of these agents, rather than collecting data from diverse logs or monitoring systems, unifies the data collection process, ensuring consistency and reliability in the local data ingested by the main server.

- **Main Server**: Acting as the system's core, the main server acts as the orchestration layer for ingestion, correlation, risk assessment, alert generation and contextualisation, exposing a UI for analyst interaction. It takes the role of the system's autonomous capabilities, coordinating the full detection pipeline while remaining modular and reproducible. The server represents the central

proof of feasibility for CTI-driven detection, directly addressing all three research hypotheses (H_1, H_2, and H_3) through its comprehensive functionalities.

Together, these elements form a complete ecosystem that demonstrates the feasibility of autonomous CTI-driven alerting. Each component is designed with the conceptual architecture (Section 3.2) as its foundation, ensuring that the system's overall behaviour supports the various hypotheses defined in Section 3.1. This includes a clear understanding of the roles of each component, their interactions, and how they contribute to the system's objectives, addressing the requirements R_1-R_9.

In Figure 8, an overview of the POC architecture is presented, illustrating the various components, their inner workings, and the interactions between them. Each of these aspects is further detailed in this chapter, being identified in the figure by their respective section, or subsection, numbers for clarity.

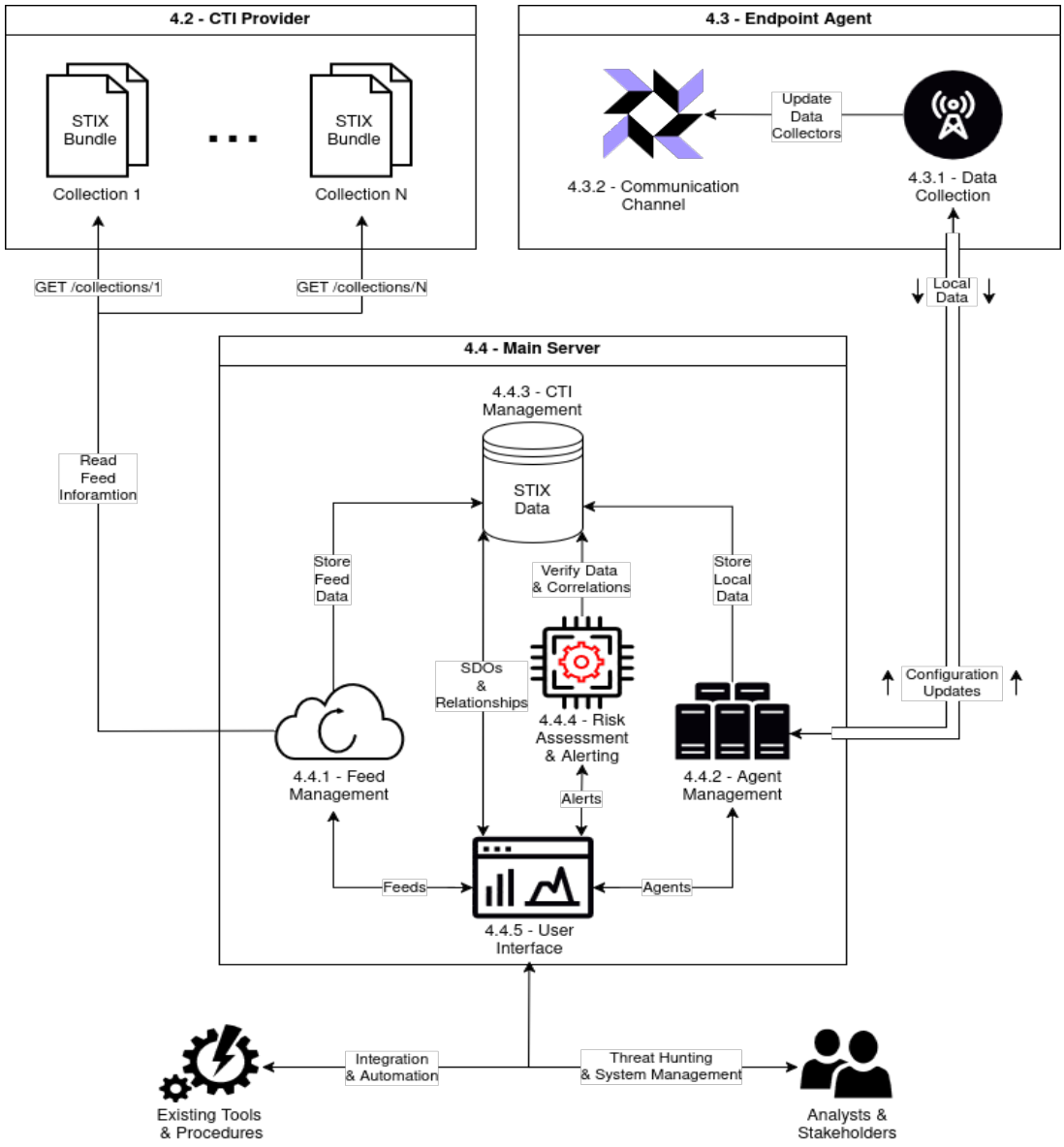


Figure 8: Overview of the POC architecture, illustrating the main components and their interactions.

This structure, while closely aligned with the conceptual architecture presented in Figure 7, represents the practical implementation of the POC, providing a clear mapping between the theoretical framework and its implementation.

CTI Providers (Section 4.2) simulate external intelligence feeds, representing the diverse sources from which the system ingests information. By providing structured indicators in STIX format, providers demonstrate the system's ability to integrate and continuously update its knowledge base (R_1, R_8). In the Proof of Concept, providers are deliberately limited to its core functionality, focussing solely on the publication of threat intelligence.

Endpoint Agents (Section 4.3) act as information collectors within the monitored endpoints. They gather live information about the host, such as running processes, existing files, or active network connections, and forward it in a structured format to the main server. This data can then be mapped to known SDOs. These agents directly address R_2 and are essential for contextualising global intelligence within the operational environment, thus enabling the correlations required for autonomous alerting (R_3). Their design also supports adaptability to everchanging environments and threat landscapes, aligning with R_9 by having set in place the mechanisms to change collection parameters based on the server's directives.

Finally, the Main Server (Section 4.4) represents the core processing and coordination layer of the POC system. It is responsible for the centralised management of the entire pipeline from data ingestion, both from providers and agents, to the generation of alerts based on correlated intelligence, presenting the information through the implemented interfaces. With this goal, the server ingests CTI, normalises the data into STIX Domain Objects, correlates them based on reported STIX relationships, and applies the risk assessment logic to determine when alerts should be created and presented to stakeholders. By centralising orchestration, the server directly supports CTI ingestion and correlation (R_1-R_3), and risk-based alerting and contextualisation (R_4-R_8). To address both R_8 and R_9, the server is designed to operate autonomously, continuously updating its data, assessing the overall threat landscape, and adjusting the behaviour of data collection from agents as needed without human intervention, with the objective of demonstrating the feasibility of an adaptive, CTI-driven alerting system.

These three components constitute the foundation upon which the Proof of Concept is built. The flow of data within the system aims to demonstrate how autonomous CTI ingestion and local data collection can be orchestrated, in practice, to detect and contextualise cyber threats.

The following chapters will further detail the implementation of the CTI providers, the endpoint agents, and the main server, respectively. This includes component-specific design decisions, their inner workings and relevant technical details.

4.2 CTI Provider

The first component is the CTI provider. Following the research conducted in Section 2.3, implementation focusses on replicating common threat intelligence sharing practices, especially when it comes to the STIX format. For its technology stack, the server is built using the Python Flask framework, as described in Section 4.1.3.

The Proof of Concept (POC) follows the principles of the TAXII collections, as they provide a straightforward request-response model that aligns with the scope of the project. This decision ensures control over the shared data, as the server can be prepopulated with relevant STIX documents that align with the testing scenarios.

With these considerations in mind, the provider component is implemented as a simple web server serving STIX collections on defined endpoints. This mimics the TAXII server-client architecture, allowing the main server to periodically poll for new intelligence. As for the data itself, the collections consist of a array of STIX bundles. A bundle is a data structure that encapsulates multiple STIX objects, allowing them to be grouped together for easier management and sharing. In this project, each bundle has the following structure:

- **"type"** - This indicates that the object is a STIX bundle;
- **"id"** - A unique identifier for the bundle;
- **"objects"** - An array of STIX objects contained within the bundle. These objects can include various types of threat intelligence data, such as IP addresses, file hashes, and more;
- **"relationships"** - An array of relationship objects that define how the various STIX objects within the bundle are related to each other. This is crucial for understanding the context and connections between different pieces of threat intelligence, contextualising them within a broader threat landscape.

This structure allows for a comprehensive representation of threat intelligence, enabling the system to ingest and process complex data sets that reflect existing cyber threats (R_1). The inclusion of relationships between objects is particularly important, as it provides the necessary context for effective correlation and alert generation, which are key functionalities of the overall system (R_3).

The provider is then configured to receive a list of collections at startup, previously stored JavaScript Object Notation (JSON) files (each representing bundles of STIX objects), allowing for easy expansion and

modification of the intelligence data available. Each collection is loaded into memory, and, for the purpose of replicating real-time scenarios, bundle by bundle, the data contained within these files are periodically updated to reflect changes in the threat landscape. By accessing these endpoints, the main server can periodically retrieve up-to-date CTI.

This setup not only facilitates the testing of the system's core functionalities, but also ensures that the POC reflects current practices. The CTI Provider offers a controlled and dynamic delivery of data that mimics the nature of threat intelligence sharing in the real-world.

4.3 Endpoint Agent

The endpoint agent is responsible for gathering local intelligence within the network's endpoints. Its primary function is to monitor and collect relevant information from the host system, which can then be sent to the main server for analysis and correlation with threat intelligence data. These data points must be comprehensive enough to allow the main server to effectively parse them into existing SDO structures, as outlined in requirement R_2.

This agent is capable of reliably communicating with the main server, both for data transmission and for management purposes. This includes the ability to receive configuration updates to the collection practices, allowing a dynamic and adaptable monitoring setup that can respond to ever-changing security needs, as specified in the requirement R_8.

For this to be achieved, this component implements a data collection and a communication module. The data collection module is responsible for gathering relevant information from the host system, according to current server requirements. This includes monitoring ongoing processes, network connections, file system changes, and other pertinent activities that can indicate potential security threats. The other component is the communication channel that handles the transmission of data to the main server and its management on local configurations. This module is designed to ensure reliable and secure communication, using established protocols (Section 4.3.2) to support the integrity and reliability of the data.

In this chapter, the implementation details of both modules are discussed in more detail, highlighting their roles in fulfilling the requirements of the overall system. The agent's data collection takes advantage of *osquery*'s capabilities, as described in Section 4.1.1. This provides a flexible and extensible foundation by incorporating a reliable way of collecting extensive data from its host. This enables a practical yet comprehensive implementation that balances complexity and performance, while still providing a clear communication protocol that future data collection methods can adhere to to integrate with the main

server without significant changes to the overall architecture.

4.3.1 Data Collection

The data collection module is implemented by managing *osquery* interactively in order to gather relevant information, while directly preparing it to be parsed into SDO structures. The Listing 4.1 presents an example of how *osquery* the agent collects data from the host system.

```
1  SELECT
2      f.path, f.size, f.atime, f.ctime, f.mtime,
3      h.md5, h.sha1, h.sha256
4  FROM
5      file f JOIN hash h
6      ON f.path = h.path
7  WHERE
8      (f.mode & 73) != 0
9      AND
10     f.directory IN ('/tmp');
```

Listing 4.1: Example of an *osquery* query to collect information about executable files under "/tmp".

This example leverages *osquery*'s ability to collect valuable system information as relational data, combining its file and hash tables to provide both metadata and identification attributes of executable files located in the directory "/tmp" (Listing 4.1, line 10). The file table records key properties such as its *path, size, access, creation, and modification timestamps*, while the hash table calculates cryptographic digests (MD5, SHA1, SHA256) for each file, allowing unambiguous identification across systems and feeds. The JOIN clause (Listing 4.1, line 5) is thus used to aggregate these complementary views, creating a single enriched record that captures both structural and integrity information for every file. This information is crucial not only for identifying potentially malicious files, but also for contextualising their presence and activity on the monitored host.

The query presented then is restricted in scope by the WHERE condition (Listing 4.1, line 7), filtering results to files under "/tmp", a common location for temporary files like malware staging, persistence scripts, or other artefacts, and that meet specific permission modes, in this case *bitmask* 73. By structuring the query in this way, the system efficiently focusses the collection on high-value telemetry while avoiding noise from benign or irrelevant files elsewhere in the filesystem.

After data is collected, due to the query structure, the results can be directly mapped to a STIX Domain Object. In this case, each result can be represented as a file object, with attributes such as *path, size, access time, creation time, modification time,* and *hash* values (*MD5*, *SHA1*, *SHA256*). This structured representation contributes to the system's ability to standardise local telemetry into threat intelligence objects, fulfilling the requirement R_2.

As for the management of *osquery* itself, the agent is designed to run the tool interactively, allowing for real-time data collection and monitoring, rather than relying on scheduled queries. Queries are stored locally and can be updated by the main server at any time via the established communication channel. Each query is then executed at a fixed interval, currently set to 30 seconds, and the results are sent to the main server immediately after each execution. This interval can be adjusted on the basis of the system's needs, balancing between data freshness and resource consumption. Such balance ensures that the data collection process does not affect the host system's performance significantly.

Additionally, adhering to both R_8 and R_9, the agent is designed to prioritise responsiveness and adaptability. Such fundamentals are supported by the system's ability to access needed information and select only the relevant collectors at any given time to gather the necessary data. This approach ensures that the agent can respond promptly to changes in the system's needs, without the need for reconfiguring or restarting the tool, providing up-to-date information.

4.3.2 Communication Channel

This section details the establishment of a reliable communication channel. This component of the endpoint agent enables reliable and secure interaction with the main server, having two main functions:

- **Data Transmission** - The channel is responsible for transmitting the collected system telemetry to the main server in a structured and timely manner.
- **Remote Management** - The channel also facilitates the reception of new collector queries and configuration updates, allowing for dynamic management of the agent's data collection processes.

This bidirectional capability directly supports the requirements R_2 and R_8, by ensuring the continuous delivery of host data while maintaining the flexibility to receive instruction updates in response to emerging threats. Furthermore, it lays the foundation for the dynamic data gathering discussed above, which supports the requirement R_9, allowing the server to autonomously enable, disable, or adjust specific collectors based on the evolving threat landscape.

The server-client channel is implemented through a custom protocol layered on top of the base TLS, providing both confidentiality and integrity of transmitted data. This decision balances performance with robustness, defining an essential set of message types, while remaining extensible for future enhancements. The protocol defines the following four primary message types:

- **"data"**(data) - This message type is used by the client to send collected data from the host to the server. It encapsulates the results of executed queries, allowing for easy parsing and direct conversion into SDO structures.
- **"upd"**(update) - This message type is sent by the server to update the client's configuration. It contains a set of new or modified queries that the agent should execute, allowing dynamic adjustment of data collection based on current needs.
- **"err"**(error) - This message type is used to report errors in the overall pipeline. The client or server can send it to indicate issues such as malformed messages, errors in data collection, or other anomalies that may require attention. This ensures that both ends of the communication can maintain robustness and reliability by providing feedback on the operational status of each other.
- **"ack"**(acknowledgment) - This message type serves as an acknowledgement for received messages. Ensures reliable delivery by confirming that a message has been successfully received and processed.

Each message is encoded as a JSON object that, while not the most efficient format in terms of size, is widely supported and easily integrated with various programming languages and frameworks. Thus, by adopting it, the system ensures that the results are both human-readable and easily processed by the server, facilitating debugging and future development.

From the client side, two dedicated loops manage communications by balancing incoming and outgoing messages. The sender loop periodically executes all the active queries, encodes their results as a "data" message, and transmits them to the server at the preconfigured interval. This ensures that the system continuously receives the relevant data without the need for a manual trigger, directly supporting the autonomy required by the hypothesis. On the other hand, the listener loop is reactive, waiting for messages from the server. For example, when an "upd" message is received, the agent dynamically updates its query set without interruption to its operation, demonstrating adaptability and aligning with R_8 and R_9.

The secure connection between these two components is established using TLS certificates, with the client verifying the authenticity of the server through a trusted Certificate Authority (CA) file. This prevents

man-in-the-middle attacks and guarantees that data flows only between the authenticated endpoints. Error messages (err) further enhance robustness by providing explicit feedback when problems are detected during data transmission. Together, these measures improve the security and reliability of the communication channel, due to its overall importance in the transport of sensitive information.

In summary, the Endpoint Agent constitutes the local intelligence gathering component of the system. It effectively collects relevant data from the host system, having the mechanisms to adapt its collection practices dynamically. Its communication channel ensures that this data is reliably transmitted to the main server in a secure manner, while also allowing for remote management and updates.

4.4 Main Server

The main server represents the central component of the proposed architecture, acting as the processing, correlation, and alerting hub of the system. Although the previous sections introduced the input mechanisms through the CTI provider and endpoint agents, this section focusses on the server's role in the overall workflow. Having outlined the scope of the POC, this section delves into the server's architecture, detailing its core functionalities and how they align with the various requirements.

This implementation is structured around a set of interconnected modules, each of which plays a specific role in the overall detection pipeline depicted in Figure 9, while collectively enabling the system to ingest, process and act on Cyber Threat Intelligence.

Figure 9: Overview of the system's detection pipeline.

The process begins with the Feed Management module (Section 4.4.1), which ensures the reliable ingestion of external CTI feeds and provides the contextual material that helps in the subsequent identification of threats. Complementing this, the Agent Management module (Section 4.4.2) oversees the local agents, ensuring that monitored data is continuously collected and aligned with the system's data requirements. Both of these data sources converge in the CTI Management module (Section 4.4.3), where threat intelligence is parsed, normalised, and stored for efficient retrieval, establishing a structured foundation for correlation. Building on the input layers, this component integrates local and external intelligence, identifying relationships between monitored hosts and known threats.

This Risk Assessment & Alerting module (Section 4.4.4) is also responsible for identifying possible threats, evaluating the risk associated with each detected relationship. When the assessed risk exceeds a predefined threshold, the module creates a new alert for investigation. Finally, the User Interface (Section 4.4.5) is responsible for presenting these alerts to analysts and stakeholders. The interface is also responsible for providing visibility into the system's operations, and above all it's data, allowing users to further investigate and validate the generated alerts with the contextualised intelligence, enriching the overall threat detection and aiding in incident response. This architecture ensures that each module focusses on a specific aspect while supporting seamless interaction across the overall pipeline.

4.4.1 Feed Management

The Feed Management module is responsible for managing and ingesting external Cyber Threat Intelligence in a reliable and consistent manner. By simulating real-world CTI feeds, through the implemented Providers (Section 4.2), it ensures that the system remains aware of emerging threats and evolving attack patterns. The ingestion process is developed following the Collections principles described in Section 2.3, using a request-response model that, not being the most optimised for large-scale deployments, is well suited for the proof of concept. The server is thus intentionally designed to mimic TAXII-like collection access, where intelligence is periodically retrieved and stored internally. This approach creates a controlled, yet dynamic environment in which new intelligence can be introduced to test adaptability.

To manage these feeds, the server configuration includes a list of endpoints representing the different CTI providers. Each entry consists of the provider's URL and specific identifier. This is then used by the module to periodically poll for new intelligence data, tagging each retrieved object with its source for traceability and enhanced context.

The feed configuration can be extended in future iterations to include additional parameters, such as authentication credentials, polling intervals, and filtering criteria, for example, based on the Traffic Light Protocol (TLP) classification or specific IOC types. This would allow the system to adapt to a wider range of real-world scenarios, where different providers may have varying access requirements and adhere to diverse data sharing standards.

By enabling the structured ingestion of intelligence (R_1) and ensuring continuous, autonomous, updates of external sources (R_8), the Feed Management module directly supports the central research hypothesis. It establishes the foundation upon which external CTI is correlated with local telemetry and demonstrates adaptability by reflecting changes in the evolving threat landscape.

4.4.2 Agent Management

The Agent Management module coordinates server communication with the deployed endpoint agents (Section 4.3) across the monitored environment. Its main role is to handle incoming telemetry, manage the set of active agents, and relay updated collection policies as dictated by the server. This ensures that data retrieval can be continuously updated and remains in accordance with the needs of the server.

Similarly to threat intelligence feeds, the agents are defined in the server configuration, where each entry includes the agent's network address(es) and a unique identifier. This allows the server to maintain a registry of agent activity, facilitating configuration updates, as well as data management and filtering. Data received from agents is parsed and validated, ensuring its adherence to the structure of STIX Domain Objects, before being stored for further correlation with external intelligence. Each data point is also tagged with its source agent, providing additional context for subsequent analysis.

Complementing the aspects discussed in Section 4.3, this module encompasses the communication channel with the agents, following the previously defined protocol. This means that upon receiving a "data" message from an agent, the server must parse the incoming information, previously collected by *osquery*, and store it in a structured format, as to accomplish requirement R_2. The parsing process of both internal and external data is done by leveraging information from the collectors, which define the expected SDO type, as well as the relevant fields for the incoming data. This ensures that the data is consistently structured, facilitating efficient correlation and analysis in later stages.

As an example of this workflow, the collector presented in Section 4.3.1 must first be registered in the server collectors configuration. This defines the main aspects of the data that is being collected from the endpoints, including the following fields:

- **"name"** - This is a human-readable name for the collector which helps in identifying the query being executed (as well as its results);
- **"query"** - This is the SQL-like query that *osquery* will be executed on the endpoint to retrieve the relevant data;
- **"type"** - This specifies the type of SDO that the collected data will be mapped to;
- **"threshold"** - This defines the minimum risk score required to activate this collector (further discussed in Section 4.4.4);
- **"enabled"** - This boolean field indicates whether the collector is currently active and, if so, if it is sent to the agents for data collection.

This structure allows the server to maintain a clear understanding of the data being collected, ensuring that it aligns with the system's overall objectives and requirements (R_2). When the connection between an agent and the server first starts, the server emits an "upd" message that includes the list of all active collectors. This enables the agent to configure its data collection process according to the server's current requirements and start the data collection process.

Through this communication protocol, when the server identifies the need to update an agent's collection policy (further discussed in Section 4.4.3), it re-sends a message of type "upd" containing the new set of queries to be executed by the agents. Once agents receive this message, they update their local configuration and start reporting the newly requested data back to the server. It should be noted that, due to the periodic nature of data collection and bidirectional communication, there may be cases where the information being received by the server is not immediately reflective of the latest configuration changes.

However, despite this potential delay, the system is designed so that data collected is not lost, even if the collector has been disabled in the meantime. The main server identifies, based on the message, what query results are being reported and continues to identify and store the received objects. Another important aspect is that the delay between configuration updates and data collection is tied to the agent's polling interval, which is set to a relatively short period, currently defined at 30 seconds. Future iterations may extend this by implementing an interactive query system, allowing the server to request specific data on demand, rather than relying solely on periodic updates, improving the system's responsiveness to specific threats or investigative needs.

These data collection and update mechanisms ensure that the system remains robust and capable of adapting to changing data requirements without losing valuable information. Without this, agents would be unable to adapt their data retrieval methods as the threat landscape evolves, enhancing their role in fulfilling the requirement R_9.

4.4.3 CTI Management

Following the main input layers of the system, the CTI Management module includes the main database of the system, where both external and local intelligence are stored in a structured manner (STIX). It also provides a data broker that allows efficient retrieval and management of the stored SDOs. This module also covers the parsing and normalisation processes, providing the necessary utilities for the feed and agent management modules to store incoming data consistent with the expected SDO format.

This database is designed with the goal of having a centralised repository of threat intelligence in its natural structure, allowing efficient querying and correlation. For this purpose, and given the scope of the

POC, the technology chosen originates from Python's stix2 library (OASIS Open, 2025). This provides two main components, a memory store and a composite data source, which together form the backbone of the CTI management module.

The MemoryStore is an in-memory database that serves as the primary persistence layer for all ingested STIX objects. It provides basic Create, Read, Update, Delete (CRUD) operations on structured threat data. This type of storage is optimised for fast insertion and retrieval of threat data, being particularly well suited for this POC as it allows prototyping and testing without the overhead of managing a full-fledged large-scale database system. Within the POC, this store acts as the canonical repository of intelligence, storing both externally ingested indicators and locally generated observables. Due to it being an in-memory storage, it guarantees low-latency access, which is essential for real-time correlation and alerting, having its main limitation being the lack of persistence across server restarts and scalability to very large datasets.

This design decision closely aligns with the POC's focus on demonstrating core functionalities and concepts, rather than handling production-scale data volumes. Large-scale implementations will require further considerations regarding data retention policies, backup strategies, and, most importantly, security measures to protect sensitive threat intelligence against unauthorised access, tampering, or breaches.

The second component is the CompositeDataSource, which functions as an abstraction layer allowing for future extensibility of the data storage backend. Provides a unified interface for querying and retrieving SDOs, regardless of their underlying storage mechanism. This means that while the current implementation of the Proof of Concept uses an in-memory storage approach, the architecture is designed to accommodate additional data sources in the future without the need for significant rework.

Detailing the processes involved in parsing and normalising the incoming data, both the Feed Management and Agent Management modules rely on predefined methods to convert raw input into structured SDOs that can be stored in the database. This is achieved by leveraging existing objects within the STIX framework, which provides the necessary structure to represent various types of threat intelligence and their relationships, and an extensible schema that can accommodate system-specific requirements.

One of the key concerns when dealing with diverse data sources is ensuring that the ingested information is deduplicated and consistently formatted, avoiding redundancy and preserving data integrity. This is particularly important in the context of CTI, where multiple sources can report on the same threat or indicator, including both external feeds and local telemetry. To achieve these goals, the storage layer is complemented by a dedicated CTI Broker. While memory and composite stores provide the raw database and querying capabilities for STIX objects, the broker introduces a management and enrichment layer that augments those SDOs with contextual metadata.

The broker is responsible for managing the lifecycle of SDOs, enriching them with additional attributes, handling deduplication, and maintaining a full modification history. Specifically, to further contextualise each object, it is assigned the following metadata fields:

- **"id"** – The unique STIX identifier of the object, used for direct retrieval and linking between observables and relationships. Within the broker's context, this also serves as a stable reference across updates and enrichments;
- **"type"** – The STIX Domain Object type (e.g., ipv4-addr, file, process). This classification is essential for correlation and data filtering;
- **"tlp"** – The Traffic Light Protocol marking associated with the object. This marking is used to define the sharing policies and sensitivity of the data, ensuring that it does not get exposed beyond its intended audience;
- **"risk"** – A numerical score representing the assessed risk of the object. This value is reflective of the object's threat level, which can be adjusted based on contextual updates or decay over time (R_6);
- **"origin"** – The first source or feed from which the object was ingested. This field does not change over time, providing a traceable lineage for each object;
- **"history"** – A chronological log of modifications and updates applied to the object. Each entry includes a timestamp and the action performed, creating an auditable lifecycle record fundamental for contextualisation of alerts and what led to them (R_7).

A central feature of the broker is its deduplication mechanism as it is built around the concept of a canonical fingerprinting. When a new STIX object is ingested, non-essential metadata fields (such as timestamps or internal management values) are removed for fingerprinting, leaving only canonical attributes that define the object's semantics and are relevant between data sources. The resulting structure is then serialised into a normalised JSON string. This dictionary, at this point, is sorted to ensure consistent ordering of keys and contains only the fields that define the object's identity. This string is subsequently encoded and hashed using the SHA-256 algorithm, producing a fixed-length fingerprint that is stored alongside the object. This fingerprint represents the uniqueness of the object, ensuring that semantically identical indicators produce the same fingerprint regardless of differences such as its source or creation time, since, for example, local data sources may report the same observable with different timestamps due to it being directly related to the time of collection.

Fingerprinting serves two purposes. First, it allows consistency by preventing the storage of duplicate objects, which is particularly important when aggregating intelligence from overlapping feeds. Instead of creating redundant entries, the broker simply updates the metadata of the existing object, ensuring that the correlation process can successfully link related indicators without confusion or ambiguity (R_3). Second, it helps the data lookup process, particularly during the ingestion phase. By maintaining a set of known fingerprints, the broker can quickly determine whether an incoming object already exists in the database. Despite having unique identifiers, two objects may still be semantically identical, an operation that would otherwise require scanning through potentially large collections of existing objects. This is especially beneficial when dealing with high-throughput data streams, where the ability to efficiently manage and deduplicate incoming intelligence is crucial. Given the high volumes typically associated with threat intelligence databases, this approach drastically reduces the computational overhead of semantic deduplication and ensures that ingestion can remain performant even as the database grows.

Beyond single object management, the CTI Management module supports the system's correlation capabilities through a set of utility functions that facilitate graph-based correlation. Due to the interconnected nature of cyber threats, where SDOs often relate to one another in complex ways, these utilities enable both the system to traverse and analyse relationships between them (R_3), but also aid analysts in understanding the broader context of an object, relationship, or alert (R_7).

By utilising the inherent linking mechanisms provided by the STIX standard, the system can directly identify connections between local telemetry and known threats, forming the basis for risk assessment and alert generation. When ingesting new data from the monitored endpoints, the system creates a new relationship between the collected observable and the corresponding host. As for external intelligence, relationships reported by the feed providers are also stored, further enriching the contextual information available for correlation, allowing the system to traverse these links during the correlation process.

This graph-based approach, where different SDOs are interconnected through defined relationships, is crucial for effective threat detection, allowing the correlation between diverse data points and the identification of complex attack patterns that may not be immediately apparent from isolated objects. By mapping out these relationships the system can then assess the risk (Section 4.4.4) that associated entities may pose to the overall monitored environment, as well as gather all the necessary data for alert contextualisation (Section 4.4.5).

For this purpose, it can traverse from a single observable to all its related entities, limiting its search to a predefined depth to avoid overwhelming complexity. This is particularly important for local intelligence, where an endpoint may be connected to multiple other hosts, processes, or files, each of which may have

their own set of relationships. By limiting the depth of traversal, the system can focus on the most relevant connections without being encumbered by an excessive number of nodes and edges. This limit isn't fixed and can be adjusted based on the specific use case or analyst preference, allowing for flexibility in how deep the correlation should go, facilitating deeper investigations/threat hunting and allowing for a balance between detail and manageability.

The correlation mechanism enables the identification of wider intrusion patterns, such as mapping a malicious Internet Protocol (IP) address to related malware samples, processes or vulnerabilities. The inclusion of these utilities provides analysts not only with atomic indicators, but also with a structured view of the overall threat landscape, further fulfilling the system's requirement to contextualise alerts (R_7).

Furthermore, and perhaps most importantly, the main usage of the correlation utilities described in the previous paragraphs is in the correlation between monitored hosts and known threats, which is the core of the system's detection capabilities. By linking local SDOs with external IOCs, the system can autonomously identify potential compromises or suspicious vectors that affect the monitored environment (R_3). This correlation is not a simple relationship between two objects, but rather a more complex vector that may involve multiple hops and intermediary relationships. For example, a monitored host may report a connection to an IP address that is benign on the surface. However, through correlation, the system may identify that this IP address is associated with a known command-and-control server, which in turn is linked to a specific attacker entity, information previously shared by a trusted CTI feed. This chain of relationships provides a richer context than the simple connection previously detected, allowing the system to generate more informed alerts.

While this correlation alone isn't sufficient to definitively classify an endpoint as compromised, or even to assess the risk associated with the detected relationship, it does provide a crucial step towards that goal. By establishing these connections, the system lays the groundwork for further analysis and risk assessment, which is discussed in Section 4.4.4. In addition to this, since the data is constantly being updated, the system can also identify changes in the threat landscape, autonomously adapting its detection capabilities as new intelligence emerges. This then causes new relationship vectors to be formed and submitted for risk assessment (R_8).

4.4.4 Risk Assessment & Alerting

Having established the mechanisms for ingesting and correlating threat intelligence, the next step in the pipeline is the risk assessment and the generation of alerts. This stage is the cornerstone of the proposed architecture, as it transforms the collected and correlated information by utilising contextual intelligence

into actionable insights. It specifically addresses the requirements defined under hypothesis H_2, namely the system's ability to assess the risk associated with the correlated events (R_4), allowing their prioritisation (R_5), account for the temporal relevance of known threats through a decay mechanism (R_6), and provide detailed context supporting incident response and decision-making processes (R_7).

For these features to be feasible, the system implements a dedicated AlertManager component, which is responsible for evaluating the previously established relationships between local observables and known threats stored in the CTI database (Section 4.4.3). The identified relationship vectors are assessed for their risk and, if the predefined threshold is met, a new alert is generated. This process implies a continuous and dynamic evaluation that ensures that threat hunting remains responsive to new intelligence and changes in the monitored environment.

The initial risk scoring is performed at the SDO's level, with each data point assigned a numerical value that reflects its assessed threat score. However, this score is not static, rather evolving dynamically based on three primary factors:

1. **External intelligence:** An object that is repeatedly reported across multiple trusted feeds is likely to have its risk score increased, as the broader community consensus suggests a higher threat level, being updated frequently to match the latest intelligence (R_8);

2. **Correlation Depth:** When evaluating relationships between IOCs and local agents, the system needs to account for the vector connecting the two entities. A direct correlation (e.g., a connection from a malicious IP) is inherently riskier than an indirect one (e.g., a local file is reported to be related to a domain belonging to a suspicious actor). This calls for, at the moment when the risk is being computed, the adjustment of the base risk score of the object based on the path depth, not impacting its atomic score (R_4);

3. **Temporal decay:** To prevent outdated or stale indicators from retaining disproportionate threat levels, the system periodically reduces the risk scores of all objects. This decay factor is configurable and serves to simulate the natural degradation of intelligence relevance, ensuring that newer threats take precedence over older data (R_6, R_8).

The combination of these factors ensures that the risk score remains adaptive, reflecting both the evolving threat landscape and the relevance of the data. This approach further guarantees that the system remains resilient to feed noise and over-reporting of indicators, which are common challenges in real-world CTI ingestion. It's capabilities can, in further iterations, be extended to include additional factors, such as the asset criticality of the local host being monitored, how its relation to other high-risk objects may further

increase its threat level, or how the decay function may be adjusted based on the TLP classification of the object, ensuring that more sensitive data retain its relevance for longer periods.

With the risk scores of the different STIX Domain Objects established, the next step is the overall assessment of the risk associated with the relationship vectors and therefore the generation of alerts. For this purpose, the alert mechanism evaluates the monitored environment, scanning for suspicious relationships between local agents and existing threat objects. The workflow is as follows:

1. **Graph construction:** For each monitored agent, the system resorts to constructing an object graph up to a predefined search depth. This graph has as its core the agent itself, spreading outwards to include the observables reported by it, as well as any relationships that may connect to these objects. This graph serves as the basis for identifying potential intrusion paths linking local hosts to known threats (R_3);

2. **Path discovery:** Using the constructed graph, the manager starts by identifying all SDOs that present a non-zero risk score. It then performs a search for all possible paths between the agent node and the identified threat objects present in the graph. These paths represent potential intrusion vectors. Due to the same threat being potentially reachable through multiple paths, the system is designed to identify and evaluate all such routes, rather than stopping at the first discovered path. This comprehensive approach ensures that the risk assessment considers all possible connections, providing a more accurate representation of the threat landscape affecting the monitored agent;

3. **Risk computation:** For each path, *p*, the risk is calculated by taking into account several factors. The first is the base risk score of the threat object being evaluated, *R*. This score is then adjusted based on the depth of the path, *d*, not accounting for the agent node itself or the edges. The adjustment is done using a multiplicative factor, *f*, which is a configurable parameter that defines how much the risk should increase or decrease based on the path length, having it's base risk value be the same when the number of nodes (without the agent) equals *f*. Fitting the risk on a 0 to 100 scale, nodes that are closer to the agent have a higher impact on the overall risk (limited to 100), while those further away contribute less. Bellow, a comprehensive deduction of the risk computation formula is presented:

   (a) The path *p* alternates between nodes and relationships, starting with the agent node. Therefore its total length is:

$$L = |p| = 1 + 2d$$

Where *d* is the number of nodes excluding the agent, i.e., the depth of the path. From this relation it follows that:

$$d = \frac{L-1}{2}$$

(b) The threat object has then a base risk score *R*. To adjust this risk according to path depth, a multiplicative factor f (the depth multiplier) is applied. The adjusted value is initially expressed as:

$$R \cdot f$$

(c) Since the risk value must remain within the defined interval, the system needs to limit the maximum score to the highest threat level (100):

$$\text{vector\_risk} = \min(R \cdot f,\ 100)$$

(d) To incorporate the path depth *d*, the adjusted risk is divided by *d*, which ensures that objects further away from the agent contribute proportionally less to the overall score. It also ensures that, due to the multiplicative factor, the risk remains the same when the path length (without the agent) equals this value i.e., when $d = f$:

$$\text{vector\_risk} = \min\left(\frac{R \cdot f}{d},\ 100\right)$$

(e) Finally, substituting $d = \frac{L-1}{2}$ back into the expression results in the final version for the risk assessment formula used in the POC:

$$\text{vector\_risk} = \min\left(\frac{R \cdot f \cdot 2}{L-1},\ 100\right)$$

This approach, while having room for improvement, balances effectively the influence of direct and indirect relationships, ensuring that the risk assessment reflects both the severity of the threat, implicitly accounting for its freshness due to the decay function, and its proximity to the monitored agent (R_4, R_5, R_6);

4. **Threshold evaluation:** Following the assessment, the computed risk is compared to the globally defined alerting threshold, generating a new alert if the value is equal to or exceeds this limit. To prevent alert flooding, the system checks if a similar alert (i.e., one involving the same agent,

threat object and vector connecting them) has been generated. If no such alert exists and the current attack vector has yet to be reported, a new alert is generated for analysis. This ensures that only new, high-confidence threats are escalated to contextual alerts, thereby aiming to reduce false positives and analyst fatigue (R_5);

5. **Alert contextualisation:** Each alert then includes not only the triggering object and its risk score, but also the full path leading from the locally monitored agent to the target object. Each object and relationship in this path is included in the alert, providing its full details and history, empowering analysts with comprehensive context for the detected threat. This graph-based approach not only enriches the alert with detailed insights into the origin and relevance of the alert, but also aids visual representation and natural understanding of the threat scenario (R_7).

Upon detection, each generated new entry is added to a local collection of "active" alerts. This may then be analysed by the response team, integrated into a SIEM or SOAR platform, or visualised through the User Interface (Section 4.4.5). At that point, each alert can then be marked as "confirmed" (if confirmed as true positives and addressed by the response team) or "dismissed" (if identified as false positives). Maintaining these separate states allows for traceability and historical analysis of the system's detection performance, as well as providing feedback for future tuning of the risk assessment parameters.

The system is also able to autonomously adapt to new intelligence without manual intervention due to its ability to abstract the ingestion, correlation, and risk assessment processes from the specific data sources. This means that once the initial configuration is set, the system can continuously evolve as new CTI is ingested and new relationships are formed between local agents and known threats, generating alerts as necessary without requiring human oversight in the alerting process. This is particularly important in dynamic threat environments, where the rapid emergence of new threats emerges and large volumes of data can overwhelm traditional, manually configured detection systems. By automating the entire pipeline from ingestion to alerting, the system ensures that it remains responsive and relevant, even as the threat landscape changes (R_8).

Despite this autonomy, the system is not designed to operate in isolation or replace human analysts. Instead, it aims at improving detection by utilising data that could otherwise be overlooked due to its volume, complexity, or lack of integration into current workflows. By providing a contextualised scenario for each threat and what led to its detection (R_7), the system empowers analysts to make informed decisions (R_4), prioritise their response efforts (R_5), and focus on the most critical threats affecting their environment, accounting for their temporal relevance (R_6).

Moreover, due to its risk assessment capabilities, the AlertManager module periodically computes the

mean risk values per SDO type. While at the surface this may seem like a purely statistical exercise, it serves a more strategic purpose within the system's adaptive framework. By monitoring these mean risk scores, the system can identify trends and shifts in the overall threat landscape that it may need to adapt to. For instance, a sudden increase in the average risk score of a particular object type (e.g., a critical vulnerability being discovered in a core process) may indicate the rapid rise of threats targeting such objects. This, in turn, may warrant adjustments in its detection strategies as, by default, the system may choose to run a more modest set of collectors on its agents to prioritise critical information, reducing both noise and overhead.

With this in mind, the calculated means, together with the analyst defined collectors, serve as feedback mechanism allowing the system to autonomously adjust its data collection policies. As defined in Section 4.4.2, each collector has an associated threshold and SDO type. If the mean risk score of the corresponding object type exceeds this threshold, the collector server automatically includes it in the active set of collectors to be sent to the agents in the next update cycle. Conversely, if the mean risk falls below the defined value, this collection query is then removed from being active. This dynamic adjustment ensures that the data collection process remains aligned with the current threat environment, focussing on the most relevant indicators while minimising unnecessary data collection (R_9).

4.4.5 User Interface

The User Interface (UI) is the final component of the proposed architecture, serving as a bridge between the autonomous processes of the system and human analysts. While the previous modules focus on data ingestion, correlation, risk assessment, and alert generation, the UI ensures that these outputs are accessible, interpretable, and actionable for stakeholders and incident responders. This interface is crucial for validating the system's effectiveness and usability, as it transforms raw alerts and telemetry into meaningful insights that aid incident response and decision-making. In this sense, the system interface directly supports the requirement R_7, contextualising alerts, and related data, with sufficient context to enable effective analysis and response.

This module acts as both an output layer for the system's data and as an input mechanism for its users to manage and interact with the system. The interface exposes stored SDOs, its relationships, agents, and alerts through a series of clearly defined routes, each responsible for delivering specific types of information. This modular organisation ensures that the interface remains easy to navigate and scalable for future expansions, such as interactive dashboards or role-based access control.

Within the interface, the design prioritises clarity and usability by following a tree-based navigation

structure. It aims to create clear and intuitive pathways for users to explore the system's data, manage agents, and review alerts. The structure is organised as follows:

- **"/"**: The entry point of the UI, guiding users to the main functions. It provides a clear overview of the server's functionalities and ensures intuitive navigation. This can be tied to usability and transparency in presenting CTI-driven detection outputs.
- **"/data"**: This route exposes the system's data collection, providing structured access to both raw and contextualised CTI collected from both external and internal intelligence. It's sub-routes include:
  - **"/data/observables"**: A comprehensive list of all observables stored in the system, allowing users to filter and search based on various attributes such as type or id.
  - **"/data/relationships"**: This route presents all relationships between observables, allowing users to explore how different entities are connected within the threat landscape.
  - **"/data/traffic"**: This section specifically highlights network events that, similar to relationships, provide insight into communications among monitored hosts and external entities.
- **"/agents"**: This section shows the registered endpoint agents and, for each, the associated data and relevant information. Analysts can use this view to explore different depths of relationships, visualising how a monitored host might connect to external indicators.
- **"/collectors"**: Following the agents section, this route lists all available collectors that can be sent to the agents, working as the management interface for data collection policies.
- **"/alerts"**: This is the central component of the overall system. Alerts are presented with details such as the affected agent, the assessed risk score, the alerted threat, and the contextual graph path that connects them. This section is designed to facilitate rapid understanding and prioritisation of threats, directly supporting the incident response process.
- **"/system"**: This route, while intending for debugging, is essential for researching and development, as it provides logging information of the different components inner workings, allowing both analysts and developers to monitor the system's health and performance, ensuring transparency and trust in its operations, while supporting the overall feedback loop for future improvements.

Beyond the technical structure of its routes, the interface also plays a crucial role in addressing the requirements of the system. Most directly, the alerts page aims at providing the necessary context to

support effective incident response, as well as clearly demonstrating the system's ability to correlate local hosts with known threats, thus addressing R_7 and R_3, respectively. This ensures that the alerts are not isolated threats, but are instead a result of contextualised intelligence within the current threat landscape. In addition to the specific system requirements, the interface is designed to support a practical view for analysts. In Figure 10, an example alert is shown, demonstrating how the system presents a detected threat along with its full context.

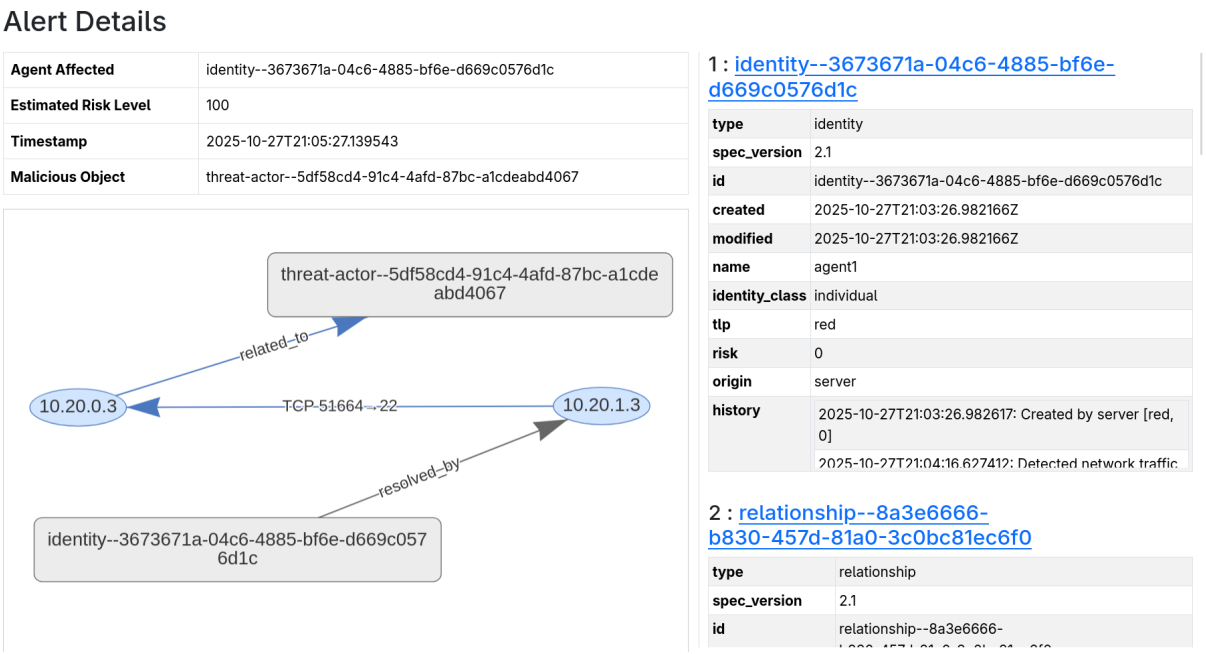


Figure 10: Example alert presented in the user interface, showing the full context of the detection.

The interface in Figure 10 presents the alert with its key attributes, including the affected agent, the threat object, the timestamp, and the assessed risk score. More importantly, it then provides two different views of the contextual graph that led to the alert. The first is a graph-based visualisation, allowing analysts to quickly grasp the relationships and connections between the monitored host and the detected threat. While currently limited to simple representation of nodes and its connections, it serves as a foundational step towards more advanced visual analytics that could be integrated in future iterations. The second view is a detailed list of all objects and relationships that form the path between the monitored host and the alerted threat. This list includes not only the basic attributes of each SDO, but also its full history, providing analysts with a comprehensive understanding of how all the pieces fit together and evolved over time. Moreover, this list also contains details about each relationship, including its type and any additional attributes that may provide further context. This dual representation ensures that analysts have both a high-level overview and a detailed breakdown of the alert.

Deepening the base analysis, all SDOs presented in the alert are linked to their respective detailed

views within the data section of the interface, as shown in Figure 11.

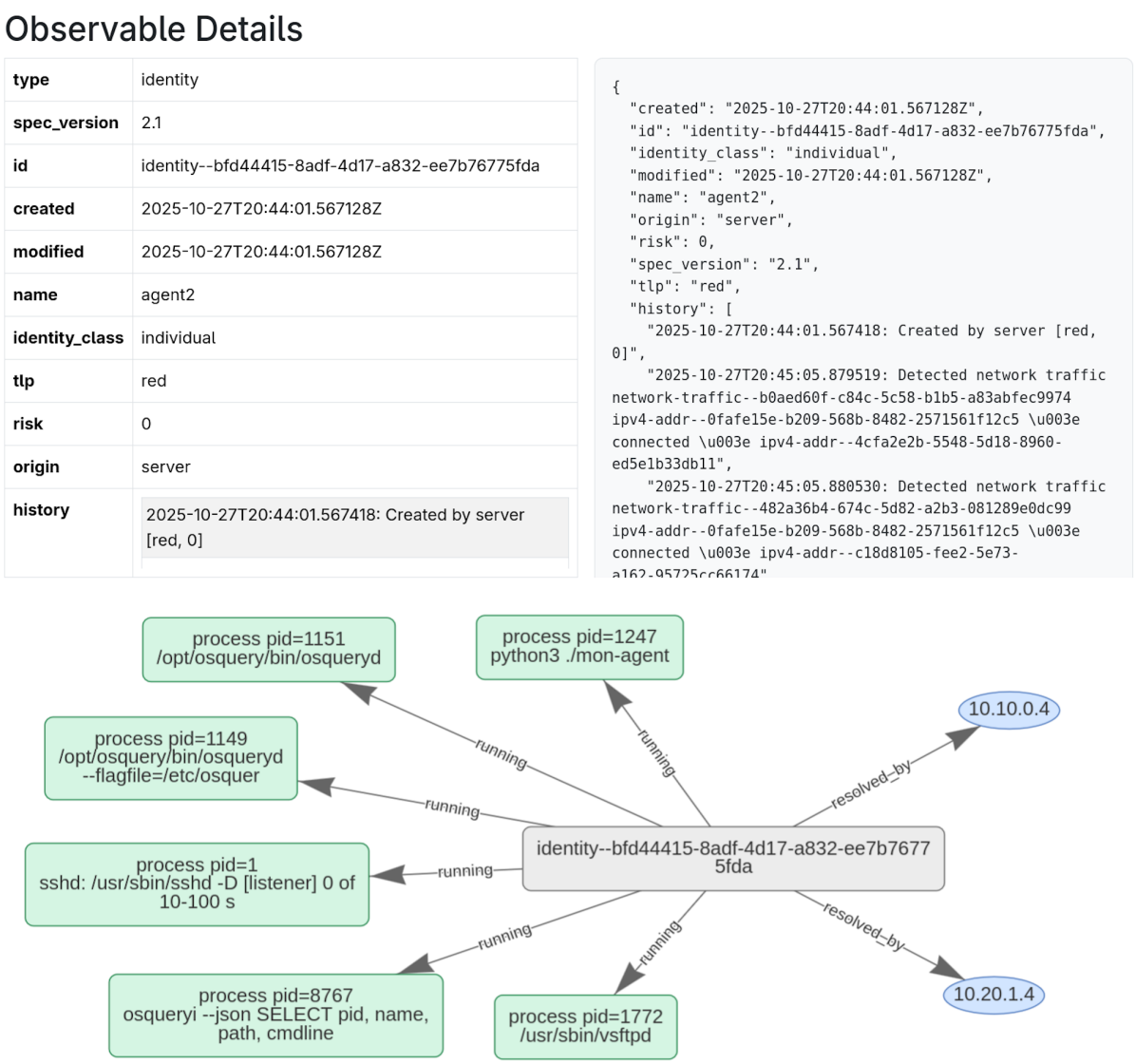


Figure 11: Detailed view of an observable, showing its attributes and history.

This view allows analysts to further investigate each individual object, exploring its attributes, history, and how it may relate to other entities. It provides a textual representation of the SDO, including all its attributes and metadata, as well as a visual graph of its direct relationships. By providing a deeper level of detail, the interface aids analysts in tracing the origins and implications of each alert, as well as related entities that may further contextualise existing threats, supporting thorough investigations and informed decision-making. The same concept applies to the agents section, where each monitored host is presented with its own detailed view (Figure 12).

Agent Details

name	agent1
type	agent
obj_id	identity--3673671a-04c6-4885-bf6e-d669c0576d1c
risk	0
internal_ip	10.10.0.3
external_ip	10.20.1.3
last_seen	2025-10-27T21:12:25.251095

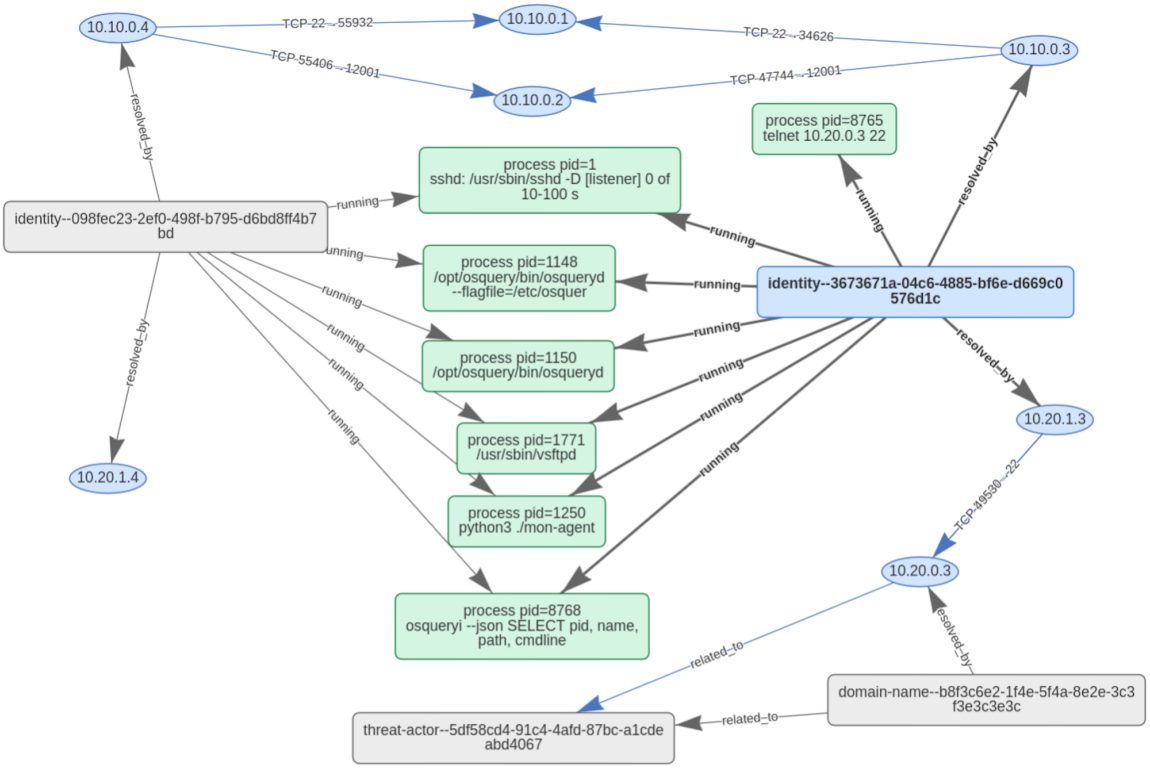


Figure 12: Detailed view of a monitored agent, showing its attributes and relationships.

Although more visually orientated, this dashboard allows analysts to further explore not only the data directly related to the host, but also its connections to known threats, as well as its integration within the broader monitored landscape.

The UI serves as the system's main interface that transforms the proposed solution from a purely autonomous detection pipeline into an accessible and user-friendly platform, bridging the gap between the automation and human decision-making. By providing clear, structured access to both internal and external data and alerts, it demonstrates how the system can deliver contextualised intelligence in a usable form. This therefore supports the overall goal of the system to enhance threat detection and response through the effective use of CTI which, while focussing on autonomous capabilities of the system, does not neglect or replace the importance of human analysis and decision making in the cybersecurity domain.

4.5 Summary

This chapter translated the conceptual architecture proposed in Chapter 3 into a tangible Proof of Concept implementation. It defined clear testable requirements in alignment with the various research hypotheses, ensuring the system's ability to demonstrate the feasibility of CTI-driven alerting.

The chapter described the rationale behind key design choices, including the selection of programming languages and standards to ensure interoperability and extensibility. This included understanding the trade-offs made to balance complexity, performance, and maintainability within the constraints of a POC. It also takes into account the insights gained from Chapter 2 on the existing standards and practices in CTI integration and usage within cybersecurity operations.

The final system architecture was described in detail, focussing on the three primary components: the CTI Provider, Endpoint Agent, and Main Server. The implementation of each component was discussed in depth, highlighting how they collectively function to compose the overall detection and alerting pipeline. Its inner workings, from data ingestion to correlation and alert generation, were explained not only to illustrate the practical realisation of the proposed approach but also to provide a foundation for further development and research in the field.

In Chapter 5, the experimental setup and testing methodology used to systematically assess the system is presented. Here, the criteria is defined for evaluating the feasibility of POC, as well as the general research objectives. This evaluation focusses on validating the core functionalities of the system, setting the stage for subsequent analysis and discussion of the system's capabilities.

Chapter 5

Experimental Setup

This chapter outlines the experimental setup used to evaluate the POC implementation described in Chapter 4. Its primary purpose is to demonstrate that the developed system satisfies the defined requirements in Section 4.1.

To achieve this, the chapter begins by describing the test environment used in this POC, including how it was configured and how it can be replicated. This includes details on the simulated CTI feeds, local data sources, and any other relevant components that form part of the experimental setup. This then leads into a discussion of the testing methodology, which outlines the specific tests and goals used to assess the system's capabilities to meet all the defined requirements.

The chapter then presents the results of the evaluation, detailing how the system performed against each requirement. The results are analysed in the context of the research hypothesis, discussing how well the system meets the intended objectives and any insights gained from the testing process. This concludes with a discussion of the findings, including how they reflect on the overall feasibility of the approach and their relevance to real-world applications, laying the groundwork for future work and potential improvements to the system.

5.1 Test Environment

For this project, the Proof of Concept was deployed in a controlled and reproducible test environment. To ensure the system's consistency across different tests and eliminate manual configuration, which could introduce variability and possible errors, the entire environment was provisioned automatically through a combination of Ansible playbooks, Dockerfiles, and Docker Compose configurations. Choosing an infrastructure-as-code approach allows the environment not only to be redeployed reliably and consistently but also to form a reproducible foundation for the evaluation and future work.

The logical architecture of the environment consists of several distinct roles, each encapsulated within

its own Docker container to ensure isolation and manageability. It also divides the network into internal and external segments, simulating a realistic scenario in which internal hosts are monitored, while external entities act both as providers of threat intelligence or adversarial actors. An overview of the test environment is shown in Figure 13.

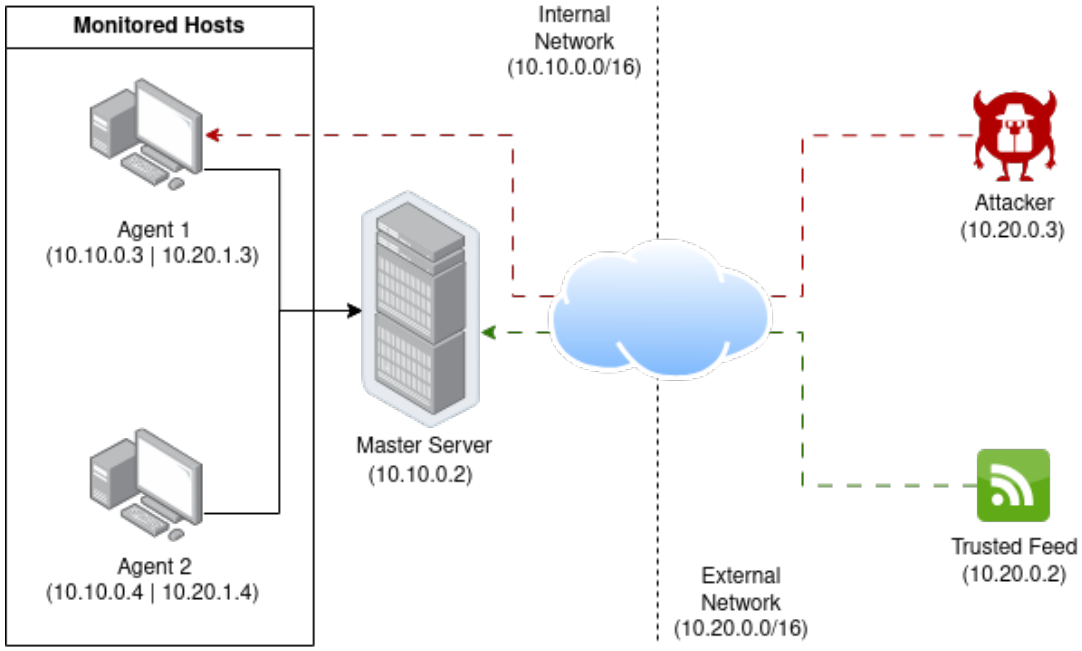


Figure 13: Overview of the experimental test environment.

This environment consists of two internal agents (10.10.0.3, 10.10.0.4), a central monitoring server (10.10.0.2) and two external entities, one emulating a trusted provider (10.20.0.2) and the other an adversarial actor (10.20.0.3).

In the internal network, each client is represented by a lightweight Linux container running a minimal operating system, along with *osquery* and the developed agent responsible for collecting the data from the host system and forward it to the main server. The server component is the core of the system, handling both internal and external CTI ingestion, correlation, and alert generation. The server, running in its own isolated container, hosts the monitoring server, managing the various feeds, as well as the web interface for user interaction.

On the external side, the trusted host represents a benign threat intelligence feed trusted by the internal monitoring server. It hosts the implemented CTI Provider (Section 4.2), which exposes structured threat intelligence data in the form of STIX bundles over a RESTful Application Programming Interface (API). This allows the monitoring server to periodically poll for new intelligence, simulating real-world TAXII collections. The malicious host, on the other hand, simulates an attacker within the network. It is configured to be

a minimal system that can generate observable events, such as network connections or file drops, that can be detected and correlated by the monitoring server. Both these hosts are crucial for validating the system's ability to ingest and process external intelligence data, as well as to validate its evolution based on the current threat landscape.

5.2 Testing Methodology

To evaluate the effectiveness of the developed system, a structured testing methodology was employed, focussing on validating each of the core functionalities outlined in the project requirements (Section 4.1). The testing process was then designed to assess the feasibility of the system, rather than its performance or comparative advantages to established solutions. This approach ensures that the system meets its intended purpose, displaying Cyber Threat Intelligence as the central element of its operation, rather than jumping directly to performance metrics.

By automating the deployment and configuration of the environment and providing tools for direct interaction with each component, the testing methodology supports a thorough evaluation of the system's functionality. This ensures that the overall process is both efficient and effective, allowing for iterative refinement and validation of the system's capabilities.

The testing of the system components themselves was divided into three key phases: data collection and correlation (Section 5.2.1), alerting and risk assessment (Section 5.2.2), and autonomy and adaptability (Section 5.2.3). Each phase was designed to validate specific aspects of the functionality of the system, addressing both specific requirements and each hypothesis outlined in Section 3.1. This structured approach ensures a comprehensive evaluation of the overall architecture and workflow of the POC.

5.2.1 Data Collection and Correlation

The first phase reflects data collection and basic correlation capabilities, ensuring that data from local agents and external CTI providers is accurately ingested and stored. This then directly addresses the first hypothesis (H_1), which focusses on the system's ability to collect and manage threat intelligence. To this end, the following aspects need to be able to validate these objectives:

- T_1 - **External** CTI **Ingestion**: The system's ability to ingest CTI from external sources was tested by simulating a trusted provider within the test environment, providing structured threat intelligence data in the form of STIX bundles. Such data, to verify its correct ingestion, have to be accessible for users through the system's web interface, ensuring its correct storage and retrieval.

- T_2 - **Local Data Collection**: The local agents' capability to collect and forward observable data from the host systems was validated by deploying the implemented agents within the test environment's local network. These agents then contact the central server, registering themselves, and sending the necessary configuration to start collecting data. The collected data, which, at the start of the tests, collects existing network traffic, shall then be visible within the web interface, confirming its successful collection and forwarding to the centralised server.

- T_3 - **Basic Correlation**: The system's ability to correlate local observable data with ingested CTI was tested by combining the data collected by the two previous steps, along with the generation of traffic connecting one of the local agents to the external malicious host. This connection, along with the data reported by the external feeds, should then trigger the generation of an alert, which would clearly correlate the local host with a malicious external entity, not previously reported by any local agent. This would confirm the system's ability to effectively correlate local and external data.

5.2.2 Alerting and Risk Assessment

Following the validation of the system's basic data collection and correlation capabilities, the next phase of testing focused on its alerting and risk assessment capabilities. This phase directly addresses the second hypothesis (H_2), which emphasises the system's ability to autonomously generate and contextualise alerts based on the data collected and correlated in the previous phase. To validate these objectives, the following aspects were tested:

- T_4 - **Risk Assessment**: For each generated alert, the ability to assess its risk level was verified, based on the controlled environment used for both the risk assigned to the SDO used in the test, and when the local agent would detect and report the connection. This means that, using the formula defined in Section 4.4.4, the risk level of the generated alert could be predicted and then compared to the one assigned by the system, ensuring its correct calculation.

- T_5 - **Alert Prioritisation**: The system's ability to prioritise alerts based on their risk levels was tested by generating multiple alerts with varying risk scores. The expected behaviour was that, by calculating the risk for each alert based on its context, the system could then provide users a mechanism to sort alerts based on their risk score, ensuring that the most critical alerts were highlighted for immediate attention. For this, the web interface was used to view and sort the previously generated alerts, ranking them from highest to lowest risk score.

- T_6 - **Decay Mechanism**: The implementation of a decay mechanism for alert relevance over time was evaluated by observing the different SDOs history. As defined in Section 4.4.3, each object has its own risk score and history. Thus, by defining the original risk score of the object and then waiting for a predefined period without any new updates, the risk score should decrease accordingly, changing not only the object's risk but also updating the object's history to reflect this change[7].

- T_7 - **Alert Contextualisation**: Following the generation of alerts, the ability of the system to provide contextual information was tested. This involves verifying that each alert included relevant details as expected in Section 4.4.5. This includes having the specific alert details, a visual representation of the alert's graph, the various objects and relationships that compose the alert, as well as their detailed information and history. Additionally, each SDO should provide links to its web page, allowing users to easily access and investigate additional information (such as other directly related observables) about each object involved in the alert.

5.2.3 Autonomy and Adaptability

Finally, the last phase of testing focused on the autonomy and adaptability capabilities of the system, directly addressing the third hypothesis (H_3). This phase aimed to ensure its ability to operate with minimal human intervention, adapt to changes in the threat landscape, and update its detection capabilities over time to detect emerging threats. For this purpose, the following aspects were evaluated:

- T_8 - **Continuous Evolution**: To assess the system's ability to evolve its detection capabilities over time, the test involved simulating changes in the threat landscape by updating the external CTI feeds with new threat intelligence data. The system starts by continuously collecting data from its local agents, and, at first, since no external CTI has been ingested, no alerts should be generated. As external feeds reports new threats, the system should then be able to correlate this new information with the data that have already been collected by local agents, generating alerts for any new threats that match the observed data. This process validates the system's ability to adapt to new threats that, being previously unknown and not initially detectable, can then be identified and alerted upon as they emerge.

- T_9 - **Autonomous Management**: The system's capability to autonomously manage its operations was tested by provisioning it with only one collector enabled at all times, while the rest required

[7] The history is updated each time the risk value decreases by a multiple of ten, down to a minimum of 0. This threshold-based update helps capture meaningful changes in risk without overfilling the history with minor fluctuations.

a higher threat level to be activated. As external CTI is ingested, the risk associated with different types of objects changes, leading to the activation of additional collectors as their thresholds are met. Similarly, as time passes and those data points affected by the decay mechanism, the collectors should autonomously be deactivated, reducing their resource consumption and optimising performance.

5.3 Functional Testing

Due to the correspondence between tests and requirements, each subsection below relates directly to a specific objective of the project, validating the system's capabilities and the underlying research hypotheses. As discussed previously, the goal of this evaluation is not to measure raw performance or scalability, but rather to demonstrate the feasibility of a system that is capable of autonomously ingesting, correlating and utilising Cyber Threat Intelligence as a central component of threat detection and alerting.

All described tests were conducted within a controlled and reproducible environment as described in Section 5.1. Before each run, the environment was redeployed using the automated deployment and configuration process to ensure a clean and consistent testing state. The following sections outline the execution and outcomes of each test, highlighting their relevance and contribution to the system's overall goals and requirements.

T_1 - External CTI Ingestion

The first test focused on validating the system's ability to autonomously ingest structured threat intelligence from an external source, as defined in requirement R_1. This is a critical foundational capability, as the entire system relies on the availability of relevant CTI to perform subsequent correlation and alert functions.

To verify this, the trusted provider within the test environment was configured to expose a API endpoint serving STIX bundles. Upon initialisation, the monitoring server periodically polled this endpoint, retrieving the latest threat intelligence data. The ingested data then gets inserted into the system's database, where it can be accessed for correlation and analysis.

The test was executed by populating the trusted provider with the STIX bundle shown in Listing 5.1, which contains a single threat actor object with a high-risk score.

```
[
    {
        "type": "bundle",
        "id": "bundle--5df58cd4-91c4-4afd-87bc-a1cdeabd4067",
        "objects": [
            {
                "type": "threat-actor",
                "tlp": "amber",
                "risk": 100,
                "spec_version": "2.1",
                "id": "threat-actor--5df58cd4-91c4-4afd-87bc-
                   a1cdeabd4067",
                "name": "attacker1"
            }
        ],
        "relationships": []
    }
]
```

Listing 5.1: Test STIX Bundle published by Trusted Feed

The monitoring server was started, and both its logs and object database interface were monitored to confirm successful ingestion. In Figures 14 and 15, examples of the ingestion log and the web interface displaying the ingested objects are shown.

```
2025-10-27 21:04:27,063 - INFO - Reading feed 'trusted1' from http:
//10.20.0.2:5000/collections/0...
2025-10-27 21:04:27,067 - INFO - Processing alerts for agent identi
ty--098fec23-2ef0-498f-b795-d6bd8ff4b7bd
2025-10-27 21:04:27,067 - INFO - Current mean risks by type: {}
2025-10-27 21:04:27,069 - INFO - Successfully read feed 'trusted1'.
2025-10-27 21:04:27,069 - INFO - Added new object with ID threat-ac
tor--5df58cd4-91c4-4afd-87bc-a1cdeabd4067 from feed 'trusted1'.
```

Figure 14: Ingestion of External Feed Log

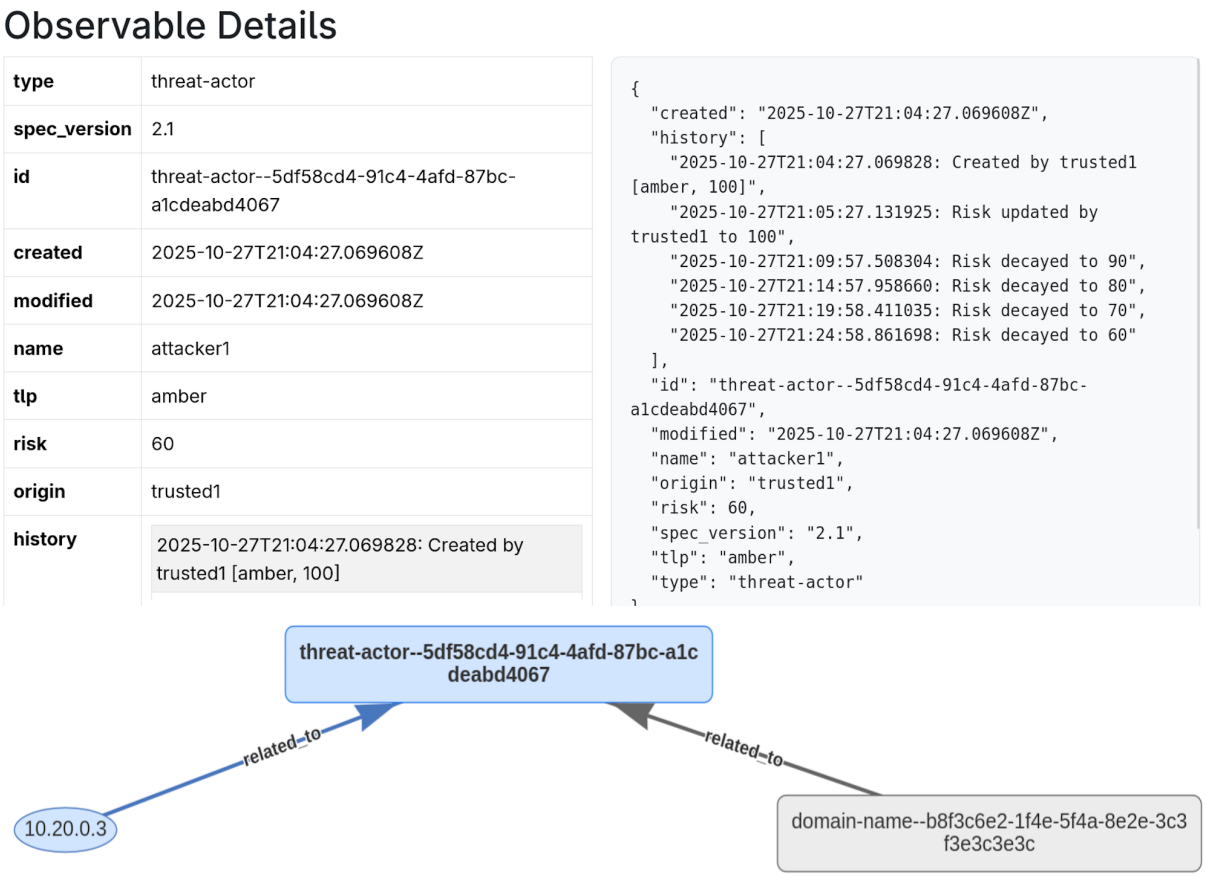


Figure 15: Ingested Object in Web Interface

This test confirmed that the monitoring server successfully retrieved and stored threat intelligence data without errors or data loss. The ingested objects were available for further processing, demonstrating that the external ingestion mechanism operates as intended, thus satisfying requirement R_1, establishing a solid foundation for subsequent tests. This is a core capability for the entire system, as all additional functionality depends on the availability of relevant CTI.

T_2 - Local Data Collection

The second test aimed to validate the ability of the system to collect and transmit data from internal hosts to the monitoring server, as defined in requirement R_2. Upon startup, both agents connected to the monitoring server, establishing a secure communication channel. For this test, the collector represented in Listing 5.2, which is responsible for gathering network connection data, was enabled, being transmitted by the server to the monitored hosts, updating their configurations.

```
{
    "list_network_connections": {
        "query": "SELECT (CASE family WHEN 2 THEN 'IP4' WHEN 10
           THEN 'IP6' ELSE family END) AS family, (CASE protocol
           WHEN 6 THEN 'TCP' WHEN 17 THEN 'UDP' ELSE protocol END
           ) AS protocol, local_address, local_port,
           remote_address, remote_port FROM process_open_sockets
           WHERE family IN (2,10) AND protocol IN (6,17) AND
           local_address LIKE '10.%' AND remote_address LIKE
           '10.%';",
        "type": "network-traffic",
        "threshold":0,
        "relationship": "connected",
        "enabled": false
    }
}
```

Listing 5.2: Test Collector Configuration

Upon successful update, agents began collecting the needed SDOs within each host, transmitting the data back to the monitoring server at regular intervals. The test was executed by simulating normal host activity, which includes the processes associated with the agent component and network connections to the main server, to generate relevant telemetry data representing expected behaviour within the monitored environment. This observable information was then stored locally and thus made available for visualisation and correlation. In Figures 16 and 17, examples of data collected from the agents and their corresponding logs are shown.

```
2025-10-27 21:32:40,320 - INFO - Server started on 10.10.0.2:12001
with SSL.
2025-10-27 21:32:40,321 - INFO - Successfully read feed 'trusted1'.
2025-10-27 21:33:00,780 - INFO - SSL connection established from ('
10.10.0.3', 49488)
2025-10-27 21:33:00,781 - INFO - Received message from ('10.10.0.3'
, 49488)
2025-10-27 21:33:00,788 - INFO - SSL connection established from ('
10.10.0.4', 45494)
2025-10-27 21:33:00,788 - INFO - Received message from ('10.10.0.4'
, 45494)
2025-10-27 21:33:10,349 - INFO - Processing alerts for agent identi
ty--2c9620c8-17fa-4e6a-9f99-d701205e048b
2025-10-27 21:33:10,350 - INFO - Processing alerts for agent identi
ty--6a11a16f-0aef-4074-a528-859c3bf6c7f1
2025-10-27 21:33:10,350 - INFO - Current mean risks by type: {}
2025-10-27 21:33:31,039 - INFO - Received message from ('10.10.0.3'
, 49488)
2025-10-27 21:33:31,039 - INFO - Processing input from ('10.10.0.3'
, 49488): list_network_connections
2025-10-27 21:33:31,040 - INFO - Added new destination 10.10.0.2 wi
th ID ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174 to CTI databa
se.
2025-10-27 21:33:31,041 - INFO - Added object network-traffic--dade
4585-dc4d-522d-af16-8708c0599070 of type network-traffic to CTI dat
abase.
2025-10-27 21:33:31,041 - INFO - Created network traffic object net
work-traffic--dade4585-dc4d-522d-af16-8708c0599070 between ipv4-add
r--6e37061d-bdbf-5e83-813e-d20601db4b1f and ipv4-addr--c18d8105-fee
2-5e73-a162-95725cc66174.
2025-10-27 21:33:31,041 - INFO - Added new destination 10.20.0.3 wi
th ID ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9 to CTI databa
se.
2025-10-27 21:33:31,042 - INFO - Added object network-traffic--73d6
1f51-0f0c-58fc-ab26-cf47e049fd19 of type network-traffic to CTI dat
abase.
```

Figure 16: Agent Registration and Object Collection Logs

Network Traffic

Show 10 entries Search:

ID	Type	From (source_ref)	To (target_ref)	Details
network-traffic--4c30176b-0541-560e-82f5-679654d57f79	network-traffic	ipv4-addr--2a75de35-a4da-5999-a6d3-ed1710a7172b	ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9	View
network-traffic--57ebd254-82b6-5d94-a162-2147f42411db	network-traffic	ipv4-addr--0fafe15e-b209-568b-8482-2571561f12c5	ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174	View
network-traffic--684aa4b5-3090-5ac2-ae1d-312a49113b77	network-traffic	ipv4-addr--0fafe15e-b209-568b-8482-2571561f12c5	ipv4-addr--4cfa2e2b-5548-5d18-8960-ed5e1b33db11	View
network-traffic--73d61f51-0f0c-58fc-ab26-cf47e049fd19	network-traffic	ipv4-addr--2a75de35-a4da-5999-a6d3-ed1710a7172b	ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9	View
network-traffic--d6dff5f8-cd70-5a61-8f0e-fab26a5fddc3	network-traffic	ipv4-addr--6e37061d-bdbf-5e83-813e-d20601db4b1f	ipv4-addr--4cfa2e2b-5548-5d18-8960-ed5e1b33db11	View
network-traffic--dade4585-dc4d-522d-af16-8708c0599070	network-traffic	ipv4-addr--6e37061d-bdbf-5e83-813e-d20601db4b1f	ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174	View
network-traffic--eee87214-215e-5290-a991-824d2c0dd0c1	network-traffic	ipv4-addr--2a75de35-a4da-5999-a6d3-ed1710a7172b	ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9	View

Showing 1 to 7 of 7 entries Previous 1 Next

Figure 17: Network Traffic Objects Collected from Agents

From the traffic presented in Figure 17, an example of a network connection object collected from Agent 1 is then shown in Figure 18.

Traffic: network-traffic--dade4585-dc4d-522d-af16-8708c0599070

type	network-traffic
spec_version	2.1
id	network-traffic--dade4585-dc4d-522d-af16-8708c0599070
src_ref	ipv4-addr--6e37061d-bdbf-5e83-813e-d20601db4b1f
dst_ref	ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174
src_port	49488
dst_port	12001
protocols	['TCP']
tlp	red
risk	0
origin	agent1
history	['2025-10-27T21:33:31.041331: Created by agent1 [red, 0]']

Source: ipv4-addr--6e37061d-bdbf-5e83-813e-d20601db4b1f

type	ipv4-addr
spec_version	2.1
id	ipv4-addr--6e37061d-bdbf-5e83-813e-d20601db4b1f
value	10.10.0.3
tlp	red
risk	0
origin	server
history	['2025-10-27T21:32:40.302286: Created by server [red, 0]', '2025-10-27T21:33:31.041394: Detected network traffic network-traffic--dade4585-dc4d-522d-af16-8708c0599070 ipv4-addr--6e37061d-bdbf-5e83-813e-

Communicates →

Target: ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174

type	ipv4-addr
spec_version	2.1
id	ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174
value	10.10.0.2
tlp	red
risk	0
origin	agent1
history	['2025-10-27T21:33:31.040465: Created by agent1 [red, 0]', '2025-10-27T21:33:31.041397: Detected network traffic network-traffic--dade4585-dc4d-522d-af16-8708c0599070 ipv4-addr--c18d8105-fee2-5e73-

Figure 18: Example Network Traffic Object Collected from Agent 1

With the data presented in Figure 18 it can be verified that the network traffic object was successfully collected and transmitted to the monitoring server, specifically due to the observation of its origin. The system demonstrates its ability not only to ingest external CTI but also to gather local intelligence within the monitored environment. This dual ingestion process is crucial for the system's overall functionality, as it builds the basis for all correlation and alerting mechanisms. The successful execution of this test directly satisfies requirement R_2, finishing the ingestion phase and preparing the system for the correlation and alert tests that follow.

T_3 - Basic Correlation

To test the basic correlation capability of the system, Agent 1 generated an outbound connection to the malicious host, simulating a behaviour such as beaconing to a known command-and-control server. The malicious IP address was previously reported by the Trusted Feed, alerting the server to the potential threat. With this information already ingested, the monitoring server was able to correlate the local agent with the externally reported threat, as presented in Figure 19.

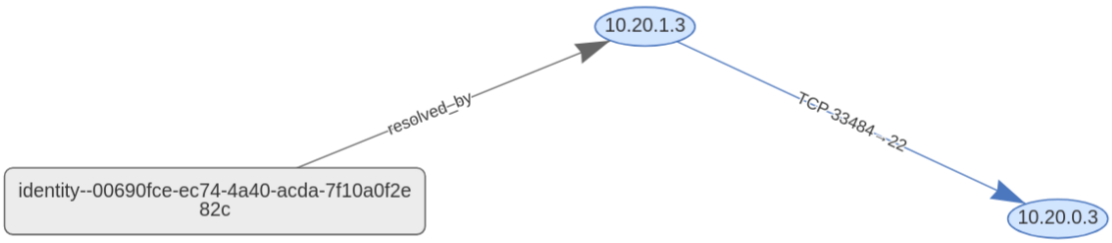


Figure 19: Correlation Graph of Agent 1 with Malicious IP

Where "*identity–00690fce-ec74-4a40-acda-7f10a0f2e82c*" represents the monitored host (Agent 1), as evidenced in Figure 20, and the IP "10.20.0.3" corresponds to the malicious host reported by the external feed, as shown in Figure 13.

Agent Details

name	agent1
type	agent
obj_id	identity--00690fce-ec74-4a40-acda-7f10a0f2e82c
risk	0
internal_ip	10.10.0.3
external_ip	10.20.1.3
last_seen	2025-10-15T22:38:24.194464

Figure 20: Agent 1 Details

This direct correlation between the host and a known malicious SDO confirms that the system can effectively link local and external intelligence, thus validating the requirement R_3. The successful correlation of data from both sources serves to demonstrate the feasibility of the first research hypothesis H_1. This also establishes the necessary context for identifying potential threats within the monitored environment, allowing the graph of knowledge to grow with each new piece of intelligence added to the system.

T_4 - Risk Assessment

Based on the correlations amongst objects, the system generated alerts indicating potential threats within the monitored environment. This includes testing cases where the alerts risk could go past the maximum value of 100, assuring that the implemented capping mechanism works as intended. It also needs to verify the alerting treshold, defined as 40 during testing. This treshold was defined based on Kent (1964) scale of estimative probabilities, where a value of 40 represents the lowest bound where the chances are considered "about even". This signifies that alerts with a risk score below this value are considered unlikely to represent significant threats, Although this value can be adjusted based on organisational risk tolerance and operational context. This threshold aims to ensure that only relevant alerts are generated, representing expected and actual score as "*NA*" when no alert is generated.

As for the multiplicative factor f, it was set to 3, meaning that connections of up to 3 hops away from the agent would be considered with heigher weight, while those further away would have a reduced impact on the overall risk score. This setting aims to balance the influence of direct and indirect relationships, ensuring that the risk assessment reflects both the severity of the threat and its relation to the agent.

For each threat reported by the external feed, the expected risk score was calculated manually, considering the depth of correlation and the score associated with the external SDO, as defined in the risk

assessment equation in Section 4.4.4. In the tests, risk decay factor was set to decrease 1 point each 30 seconds, meaning that when calculating *vector_risk* using Equation 3d (simplified depth value, d), R is calculated by subtracting twice the decay factor times the number of minutes elapsed (Δ_T) to the original risk value (R_0). Then, the expected score was compared with the actual risk assigned by the system to validate the correctness of the implemented scoring algorithm. The results[8] of this comparison are summarised in Table 1.

Table 1: Comparison of Expected and Actual Risk Scores

Alert ID	Initial Risk (R_0)	Time Elapsed (Δ_T)	Depth (d)	Expected Score	Actual Score
1	80	1	1	100	100
2	50	3	2	66	66
3	100	0	6	50	50
4	75	10	3	55	55
5	35	0	3	NA	NA
6	100	5	9	NA	NA

The results in Table 1 show that the expected and actual risk scores were perfectly aligned across all alerts tested. This confirms the accuracy of the implementation of the risk assessment, reflecting the theoretical model accurately in practice. The successful validation of requirement R_4 demonstrates that the system can effectively quantify the severity of potential threats based on their correlations and inherent risk factors, providing a reliable basis for prioritising alerts and guiding analyst attention.

T_5 - Alert Prioritization

For this test, multiple alerts were generated with varying contextual parameters and risk factors. The alerts were then accessed through the web interface to verify their correct prioritisation. In Figure 21, an example of the alert list sorted by risk score is shown:

[8] Each test was repeated 5 times.

Active Alerts

Show 10 entries Search:

ID	Timestamp	Agent	Associated Object	Estimated Risk	Details
alert--9b94b308-eda1-46e0-b41e-315c7ad0951b	2025-10-16T22:32:51.113899	identity--6ccb7905-85fc-41de-a24a-c57d8a896741	process--1e42945a-6715-4802-aa85-6b0039995b7c	100	View
alert--f61f823d-097f-45d6-af1e-82ec545094cb	2025-10-16T22:28:20.726906	identity--6ccb7905-85fc-41de-a24a-c57d8a896741	threat-actor--5df58cd4-91c4-4afd-87bc-a1cdeabd4067	100	View
alert--ab9a7c7e-6ce9-45b7-af6b-712c5155052a	2025-10-16T22:38:21.637262	identity--d1e22b19-3d85-484c-a7a7-2c2a9ce09d87	process--1e42945a-6715-4802-aa85-6b0039995b7c	96	View
alert--312c6e51-fc14-4449-82a5-d0c58a9738ae	2025-10-16T22:32:51.104377	identity--d1e22b19-3d85-484c-a7a7-2c2a9ce09d87	threat-actor--5df58cd4-91c4-4afd-87bc-a1cdeabd4067	91	View
alert--f9d03100-320b-4bda-b763-c24f8d82a616	2025-10-16T22:28:20.726780	identity--6ccb7905-85fc-41de-a24a-c57d8a896741	ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9	45	View

Figure 21: Alert List Sorted by Risk Score

This prioritisation mechanism supports efficient triage by dynamically sorted alerts according to their risk scores, validating requirement R_5. This test also supports the general usability of the system, ensuring that analysts can focus on the most pressing threats first. Since the alert risk accounts for both the severity of the threat and its correlation to the affected hosts, this prioritisation directly contributes to the system's effectiveness in real-world scenarios where timely response to threats directly affecting assets is crucial.

T_6 - Decay Mechanism

The temporal decay mechanism was evaluated by monitoring a series of SDOs with predefined risk scores over a fixed period. During this time, no new intelligence updates were introduced. As expected, the risk scores decreased over time, logging the decay process in the objects' history at set levels. An example of a object's risk log is shown in Figure 22, illustrating the gradual reduction in risk score over time:

```
{
  "history": [
    "2025-10-16T19:49:22.714860: Created by agent2 [red, 0]",
    "2025-10-16T19:49:22.715405: Detected network traffic network-traffic--
e1b1a975-6aa0-5eab-a63c-a22e50b46222 ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9
\u003c connected \u003c ipv4-addr--469479d7-836e-58a0-83e2-68f805530b34",
    "2025-10-16T19:50:32.904173: Risk updated by trusted1 to 100",
    "2025-10-16T19:51:23.746410: Detected network traffic network-traffic--
ad4c342d-10c0-5bd4-86e9-a7c9aa37a8b0 ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9
\u003c connected \u003c ipv4-addr--469479d7-836e-58a0-83e2-68f805530b34",
    "2025-10-16T19:55:03.278615: Risk decayed to 90",
    "2025-10-16T20:00:03.709072: Risk decayed to 80",
    "2025-10-16T20:05:04.176363: Risk decayed to 70",
    "2025-10-16T20:10:04.658976: Risk decayed to 60",
    "2025-10-16T20:11:20.044536: Detected network traffic network-
traffic--44f7eea5-9e5c-5e17-84cd-66ff8b661b24 ipv4-addr--
f3b80c9e-3573-5474-9cbf-0b25aef227c9 \u003c connected \u003c ipv4-
addr--469479d7-836e-58a0-83e2-68f805530b34"
  ],
  "id": "ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9",
  "origin": "agent2",
  "risk": 51,
  "spec_version": "2.1",
  "tlp": "red",
  "type": "ipv4-addr",
  "value": "10.20.0.3"
}
```


Figure 22: Risk Score Decay Log

This confirmed the accuracy of the decay implementation and demonstrated that the system can dynamically adapt the relevance of its intelligence data over time. Additionally, when the object's risk was 100%, Agent 2 made a connection to the malicious IP, which led to the generation of an alert (Figure 23).

Alert Details

Agent Affected	identity--dacddb5a-75a5-469c-a236-b0c4b612e4ec
Estimated Risk Level	100
Timestamp	2025-10-16T19:50:32.917462
Malicious Object	ipv4-addr--f3b80c9e-3573-5474-9cbf-0b25aef227c9

Figure 23: Alert Generated When Risk was 100%

This was repeated a second time, with Agent 1 making the connection, once the risk had decayed to

60%, also resulting in a alert. This behaviour led to the following alert being generated (Figure 24).

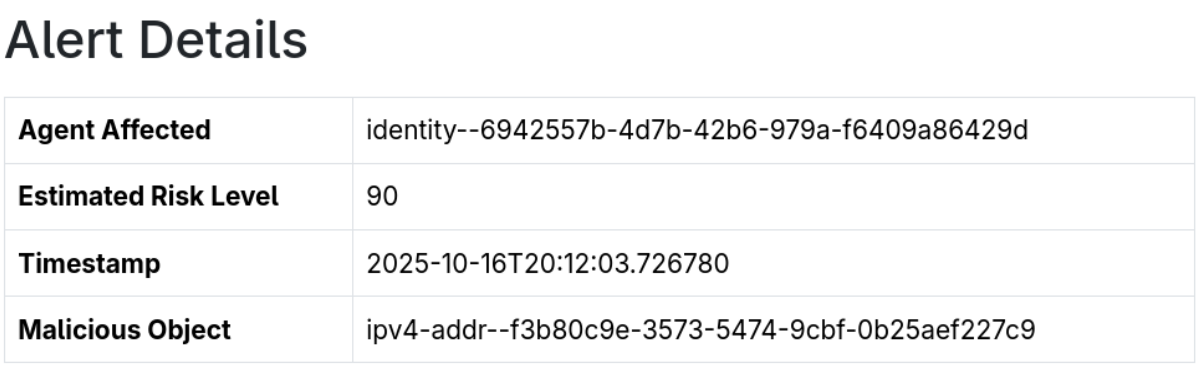


Figure 24: Alert Generated When Risk was 60%

The details of both alerts are summarised in Table 2, showing the differences in assessed risk based on the decayed risk levels of the malicious object at the time of connection.

Table 2: Alerts Generated at Different Risk Levels

Alert ID	Object Risk (R_0 - 2 Δ_T)	Depth	Affected Host	Alert Risk
A1	100	2	Agent 2	100
A2	60	2	Agent 1	90

As seen in alert A2, even though the decay has reduced the risk level of the object to 60%, the system still generated an alert with an increased risk score of 90%. This increase is attributed to the depth of the correlation which adds significant weight to the overall risk assessment formula (Equation 3d).

This demonstrates that the system correctly adjusts its severity based on the decayed risk levels, validating requirement R_6 and supporting the previous test T_4. The decay directly influences the alert generation process, ensuring that the system remains responsive to current and historical CTI.

T_7 - Alert Contextualization

The contextualisation of generated alerts was evaluated by examining the details of each alert within the web interface. The alert page provides three main sections to aid both analysts and other stakeholders understand the importance of the alert. The first is the specific details of the alert, depicted in Figure 25, which includes the id of the affected agent and the malicious object, the assessed risk score and the time stamp of its generation.

Alert Details

Agent Affected	identity--6ccb7905-85fc-41de-a24a-c57d8a896741
Estimated Risk Level	100
Timestamp	2025-10-16T22:32:51.113899
Malicious Object	process--1e42945a-6715-4802-aa85-6b0039995b7c

Figure 25: Alert Details Section

The correlation graph is displayed in Figure 26. This graph visually represents the relationships between the objects involved. This graph helps analysts quickly grasp what the alert pertains to and what entities are involved.

Figure 26: Alert Correlation Graph Section

Finally, the third section lists all related objects, Figure 27, in full detail. These are also presented in an oredered manner, starting with the affected agent, detailing its properties and history, followed by all relationships and objects, in the same sequence as shown in the correlation graph (Figure 26), ending with the malicious object that triggered the alert.

1 : identity--6ccb7905-85fc-41de-a24a-c57d8a896741

type	identity
spec_version	2.1
id	identity--6ccb7905-85fc-41de-a24a-c57d8a896741
created	2025-10-16T22:26:20.566116Z
modified	2025-10-16T22:26:20.566116Z
name	agent2
identity_class	individual
tlp	red
risk	0
origin	server
history	2025-10-16T22:26:20.566406: Created by server [red, 0] 2025-10-16T22:27:10.589501: Detected network traffic network-traffic--96aac09b-977c-539b-9c70-34a2aabeca1c ipv4-addr--0fafe15e-b209-568b-8482-2571561f12c5 > connected > ipv4-addr--c18d8105-fee2-5e73-a162-95725cc66174 2025-10-16T22:27:10.590509: Detected network traffic network-traffic--14d09b17-82c0-5f93-af83-e5620ead8db6 ipv4-addr--0fafe15e-b209-568b-8482-2571561f12c5 > connected > ipv4-addr--4cfa2e2b-5548-5d18-8960-ed5e1b33db11

2 : relationship--bfc0b7f8-2a04-45b5-9805-bb82a8d5f96d

type	relationship
spec_version	2.1
id	relationship--bfc0b7f8-2a04-45b5-9805-bb82a8d5f96d
created	2025-10-16T22:32:45.658164Z
modified	2025-10-16T22:32:45.658164Z
relationship_type	running
source_ref	identity--6ccb7905-85fc-41de-a24a-c57d8a896741
target_ref	process--1e42945a-6715-4802-aa85-6b0039995b7c
tlp	red
risk	0
origin	agent2
history	2025-10-16T22:32:45.658497: Created by agent2 [red, 0]

3 : process--1e42945a-6715-4802-aa85-6b0039995b7c

type	process
spec_version	2.1
id	process--1e42945a-6715-4802-aa85-6b0039995b7c
cwd	/usr/bin/curl
command_line	curl -L -o /dev/null https://ash-speed.hetzner.com/1GB.bin
tlp	red
risk	43
origin	trusted1
history	2025-10-16T22:29:20.779883: Created by trusted1 [white, 50]

Figure 27: Alert Relateed Objects Section

From this point on, any object can be further explored by clicking on it, allowing analysts to delve deeper into the context and history of each entity. Not only does this provide a comprehensive view of the alert itself, but it also allows for further investigation into the relationships and history of all the objects that make up the attack vector.

This contextualisation provides analysts with a comprehensive understanding of why an alert was raised, what entities are involved, and how they are interconnected. This understanding enables informed decision making based on a complete Figure of the threat scenario, rather than, for example, simply alerting to an isolated network connection. The successful implementation of this feature satisfies the requirement R_7, demonstrating that alerts are not only generated autonomously but also enriched with meaningful context to aid in their interpretation and the incident response process.

T_8 - Continuous Evolution

To evaluate the system's ability to adapt to new threat intelligence, the trusted provider updates its feeds over time with new SDOs representing emerging threats, in this case, a connection to a "clean" host that then is reported as being connected to a threat actor. For Agent 1, the connection to the new malicious host was made before the feed update, simulating a scenario where the host may have been compromised prior to the intelligence being available. In this case, once the feed is updated, the system detects that the previously collected data now correlates with the newly ingested threat intelligence, generating an alert retrospectively, as presented in Figure 28.

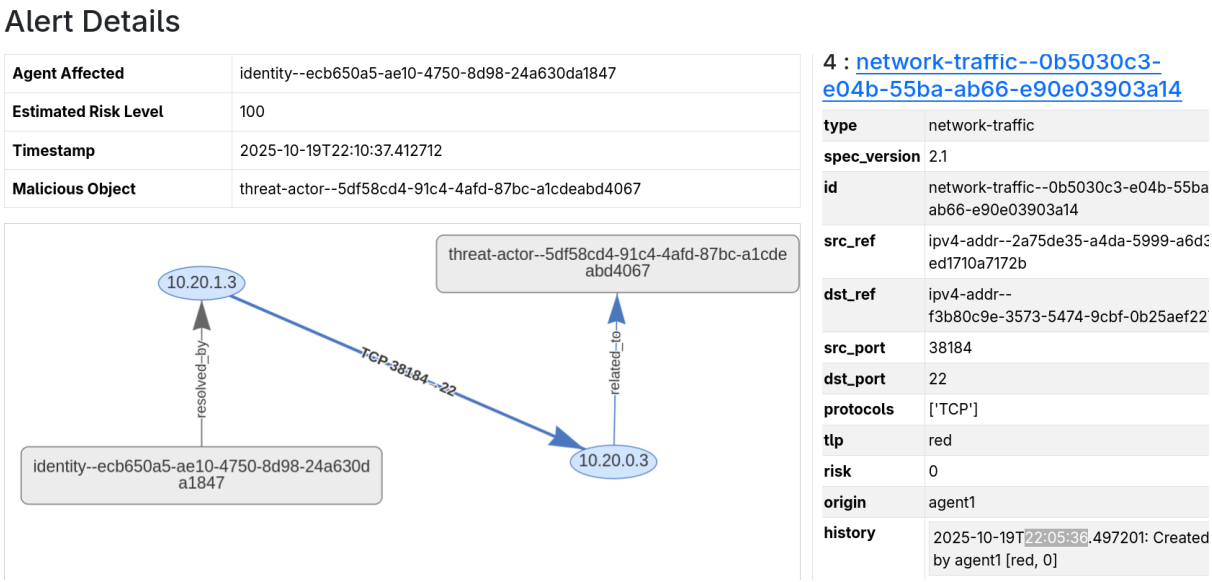


Figure 28: New Alert for Agent 1 After Feed Update

As presented in Figure 28, the alert was generated at ”22:10:37”, however, as highlighted in the *network traffic* details, the actual connection to the malicious host occurred at ”22:05:36”, prior to the feed update. This can be confirmed by Figure 29, which shows the *threat actor* details.


```
{
  "created": "2025-10-19T22:09:37.342627Z",
  "history": [
    "2025-10-19T22:09:37.342875: Created by trusted1 [amber, 100]",
    "2025-10-19T22:10:37.405005: Risk updated by trusted1 to 100"
  ],
  "id": "threat-actor--5df58cd4-91c4-4afd-87bc-a1cdeabd4067",
  "modified": "2025-10-19T22:09:37.342627Z",
  "name": "attacker1",
  "origin": "trusted1",
  "risk": 93,
  "spec_version": "2.1",
  "tlp": "amber",
  "type": "threat-actor"
}
```


Figure 29: Threat Actor Object Details

This confirms that the threat actor was only ingested into the system at ”22:09:37”, minutes after Agent 1 had already made the connection to the related host.

Similarly, for Agent 2, the connection to the new malicious host was made after the feed update. In this case, the system immediately recognised the correlation between the collected data and the newly ingested CTI, generating an alert in real-time, as shown in Figure 30.

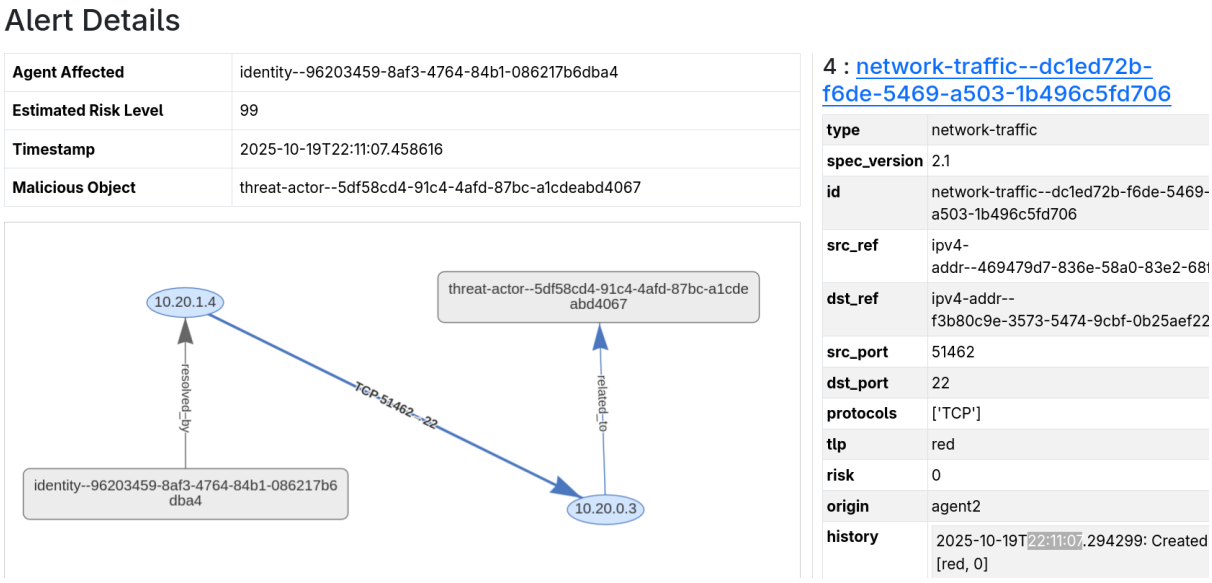


Figure 30: New Alert for Agent 2 After Feed Update

The generation of both alerts confirms not only that the system can effectively adapt to new threat intelligence, detecting emerging threats as they appear, but also retrospectively identifying possible malicious activity that occurred prior to the intelligence update. This dual capability ensures that the system remains responsive to the evolving threat landscape, continuously refining its detection mechanisms based on the latest available data. It also highlights the system's ability to bridge temporal gaps in intelligence and indirect threat detection, successfully identifying potential compromises despite the systems not having direct connections to any known malicious entities.

This dynamic adaptation is crucial to maintain an up-to-date security posture. Its relevance is especially pronounced when combating 0-Day threats, which by definition are previously unknown and thus not present in any threat intelligence feed at the time of compromise, which makes retroactive detection essential to identify potential compromises. The successful execution of this test validates requirement R_8, demonstrating the system's ability to continuously evolve its threat detection capabilities in response to the ever-changing threat landscape.

T_9 - Autonomous Management

Finally, the autonomous management capabilities were evaluated by configuring a subset of collectors with different activation thresholds based on the mean risk levels of certain object types (Listing 5.3). This allows the system to dynamically adjust its data collection strategy based on the perceived risk.

```
{
  "list_running_processes":{
    "query":"SELECT pid,name,path,cmdline FROM processes;",
    "type":"process",
    "threshold":45,
    "relationship":"running",
    "enabled": false
  },
  "list_network_connections": {
    "query": "SELECT (CASE family WHEN 2 THEN 'IP4' WHEN 10 THEN
       'IP6' ELSE family END) AS family, (CASE protocol WHEN 6
       THEN 'TCP' WHEN 17 THEN 'UDP' ELSE protocol END) AS
       protocol, local_address, local_port, remote_address,
       remote_port FROM process_open_sockets WHERE family IN
       (2,10) AND protocol IN (6,17) AND local_address LIKE
       '10.%' AND remote_address LIKE '10.%';",
    "type": "network-traffic",
    "threshold":0,
    "relationship": "connected",
    "enabled": false
  },
  "list_runnable_temporary_files": {
    "query": "SELECT f.path, f.size, f.atime, f.ctime, f.mtime, h
       .md5, h.sha1, h.sha256 FROM file f JOIN hash h ON f.path =
        h.path WHERE (f.mode & 73) != 0 AND f.directory IN ('/tmp
       ');",
    "type": "file",
    "threshold":70,
    "relationship": "owns",
    "enabled": false
  }
}
```

Listing 5.3: Autonomous Collector Management Configuration

As new information becomes available through the external feed and the mean risk levels associated with certain types of objects increased, additional collectors were automatically activated (Figure 31).

```
2025-10-27 22:44:40,856 - INFO - Successfully read feed 'trusted1'.
2025-10-27 22:44:40,857 - INFO - Added new object with ID process--
1e42945a-6715-4802-aa85-6b0039995b7c from feed 'trusted1'.
2025-10-27 22:45:01,657 - INFO - Received message from ('10.10.0.3'
, 44308)
2025-10-27 22:45:01,657 - INFO - Processing input from ('10.10.0.3'
, 44308): list_network_connections
2025-10-27 22:45:01,683 - INFO - Received message from ('10.10.0.4'
, 45899)
2025-10-27 22:45:01,683 - INFO - Processing input from ('10.10.0.4'
, 45899): list_network_connections
2025-10-27 22:45:10,887 - INFO - Processing alerts for agent identi
ty--11ab5fbb-232a-4242-b516-5e4c59ad96c2
2025-10-27 22:45:10,888 - INFO - Processing alerts for agent identi
ty--e38189cd-70d8-4755-a7c5-139cfd7ef8e4
2025-10-27 22:45:10,888 - INFO - Current mean risks by type: {"thre
at-actor": 96.0, "ipv4-addr": 36.0, "domain-name": 26.0, "relations
hip": 26.0, "process": 49.0}
2025-10-27 22:45:10,888 - INFO - Query 'list_running_processes' ena
bled state changed from False to True
```

Figure 31: Collector Activation Log

This leads to the collection of more detailed information from agents about specific object types. By correlating this new data to the reported threats, new alerts were generated, as shown in Figure 32.

Alert Details

Agent Affected	identity--11ab5fbb-232a-4242-b516-5e4c59ad96c2
Estimated Risk Level	100
Timestamp	2025-10-27T22:46:40.960061
Malicious Object	process--1e42945a-6715-4802-aa85-6b0039995b7c

Figure 32: Alert Generated After Autonomous Collector Activation

This represents the main goal of the entire system, adapting its detection mechanisms autonomously based on the perceived threat landscape and alerting to emerging risks. This capability is crucial for maintaining an effective security posture in dynamic environments, where threats can evolve rapidly and require varying levels of detail to be effectively detected and mitigated.

On the other hand, when the mean risk associated with certain types of objects decreased with time (Figure 33), some collectors were automatically deactivated to reduce unnecessary data collection and processing overhead.

```
2025-10-27 22:47:41,005 - INFO - Current mean risks by type: {"thre
at-actor": 91.0, "ipv4-addr": 31.0, "domain-name": 21.0, "relations
hip": 21.0, "process": 44.0, "file": 96.0}
2025-10-27 22:47:41,005 - INFO - Query 'list_running_processes' ena
bled state changed from True to False
2025-10-27 22:47:41,046 - INFO - Reading feed 'trusted1' from http:
//10.20.0.2:5000/collections/0...
2025-10-27 22:47:41,048 - INFO - Successfully read feed 'trusted1'.
2025-10-27 22:48:05,003 - INFO - Received message from ('10.10.0.3'
, 44308)
2025-10-27 22:48:05,004 - INFO - Processing input from ('10.10.0.3'
, 44308): list_network_connections
2025-10-27 22:48:05,008 - INFO - Processing input from ('10.10.0.3'
, 44308): list_runnable_temporary_files
2025-10-27 22:48:05,012 - INFO - Received message from ('10.10.0.4'
, 45899)
2025-10-27 22:48:05,013 - INFO - Processing input from ('10.10.0.4'
, 45899): list_network_connections
2025-10-27 22:48:05,016 - INFO - Processing input from ('10.10.0.4'
, 45899): list_runnable_temporary_files
```

Figure 33: Collector Deactivation Log

This dynamic adjustment of the data collection strategy ensures that the system remains efficient and focused on the most relevant threats, avoiding information overload and potential performance degradation while still maintaining a high level of situational awareness.

With the autonomous management of collectors and its consequent effect on the overall detection capabilities, the test demonstrates the successful implementation of an adaptive mechanism that responds to environmental conditions without manual intervention. This fullfills the last requirement R_9, confirming its relevance and effectiveness in enhancing the overall functionality of the system over time.

5.4 Summary

This chapter detailed the experimental validation established to evaluate the POC implementation. It defined the test environment, ensuring that it was replicable and comprehensively detailed so that future researchers could reproduce the conditions and results.

It also outlined the evaluation methodology, detailing the specific tests and how they aligned with the requirements defined in Section 4.1. Each test was designed to systematically assess the functionalities that contribute to the system's ability to autonomously process CTI, correlate it with local data, and generate contextualised alerts. In doing so, it demonstrated the technical feasibility of CTI-driven alerting while exposing important aspects such as the importance of balancing the risk assessment process and the need for continuous adaptability to ever evolving threat landscapes.

Critically, these results underline the promise of the proposed system, while acknowledging its limitations. Functional testing validates the core hypotheses of this research, showing that the system can achieve its intended objectives within the defined scope. Nonetheless, further work is necessary to extend the evaluation to quantitative performance metrics and scalability studies. Reflecting on these aspects establishes a strong foundation for Chapter 2, where the broader implications and findings of this work, particularly regarding the usage of Cyber Threat Intelligence, are discussed.

Chapter 6

Discussion

This chapter provides a critical discussion of the dissertation in its entirety, integrating both insights and lessons derived from the research, system design, implementation, and evaluation phases. Instead of reiterating implementation specificities, the focus is on the broader implications of the work concerning the feasibility and relevance CTI-driven threat detection, its practical impact on cybersecurity workflows, and lessons learned from translating the proposed theoretical model into an operational system.

By critically reflecting on the results, challenges and insights obtained throughout this study, this chapter aims to provide a holistic evaluation of both the contributions made and the lessons learnt regarding the practical and conceptual value of CTI.

6.1 Reflection on the Proposed Solution

The design, development and evaluation of the proposed solution provided a comprehensive platform to assess the feasibility of autonomous CTI-driven detection and alerting. It enabled both theoretical and practical exploration of the core research objectives outlined in Chapter 1, specifically regarding the capabilities of CTI to function as an active component in threat detection workflows.

Choosing a modular and pipeline-oriented design was key to achieving this outcome. By separating the input layers, processing logic, and output mechanisms, the architecture enables clear visibility into each stage of the workflow, which contributes both to further development and evaluation of the system. This modularity also allows for future integration of additional data sources, the refinement of correlation logic, or the adoption of more sophisticated risk assessment techniques without disrupting the overall workflow.

The results presented in Chapter 5 demonstrate that each requirement set forth during the design phase (Section 4.1) was successfully met. This, while conducted in a controlled environment focused on functional validation, poses a strong indication that CTI driven automation is not only theoretically feasible but also practically achievable. Although its performance in real-world scenarios remains to be tested,

the tests conducted here confirm the system's ability to autonomously ingest, correlate, and contextualise information with minimal human intervention. This addresses a significant gap identified in the literature review (Chapter 2), where CTI is often treated as a support tool rather than as an active detection driver.

Furthermore, the ability to, without preconfigured rules or Machine Learning models, dynamically identify new threats that are related to the local context as they emerge highlights the adaptability of the proposed approach. This is particularly relevant as one of the main challenges in intelligence sharing is identifying which threats are pertinent to a specific environment. Using both external and internal data sources, the system effectively narrows the scope of relevant SDOs, enhancing the practical utility of CTI in operational settings without overwhelming analysts with irrelevant information.

However, this reliance on external intelligence also highlights the importance of quality and timeliness of the CTI feeds used, which, while not directly evaluated in this work, can lead to the propagation of false positives or missed detections if the intelligence is outdated or inaccurate. In this regard, the current system addresses this challenge in two ways: first, by allowing the integration of multiple feeds, assuring a broader coverage of threats and reducing dependence on any single source, and second, by providing transparency into the decision-making process through deterministic logic and the full contextualisation of alerts and their associated data, enabling analysts to trace back the origin of alerts.

Ultimately, the developed POC validates that threat detection driven by CTI is achievable when supported by a structured processes that balances automation and human oversight. It establishes both a theoretical and technical foundation upon which more complex adaptive mechanisms can be later developed to further explore the full potential of CTI in cybersecurity operations.

6.2 The Impact of Cyber Threat Intelligence

The research carried out has revealed several important insights into the practical and conceptual value of CTI in cybersecurity operations. Firstly, the study of current standards and practices, as discussed in Chapter 2, highlighted the intrinsic challenges of handling intelligence data. CTI feeds can often be inconsistent, outdated, or restricted, demanding flexibility and adaptability from any system that seeks to operationalise them. The developed POC addressed these challenges by implementing a modular architecture capable of extending and adapting its input sources as needed. These findings also drove the design of critical components such as risk assessment and decay mechanisms, which proved crucial in maintaining the adaptability and relevance of the detection process over time.

Secondly, with the development and testing of the POC, it was confirmed that CTI can be directly

operationalised to drive automated detection processes. This shifts the role of CTI from being a mostly passive resource used during incident investigations to also playing an active role in real-time threat detection workflows. The system demonstrated that CTI can be transformed into actionable intelligence that both aids detection and contextualisation.

Finally, the results demonstrated the relevance of contextual mechanisms in threat detection. By correlating external threats with local intelligence, the system improves both detection relevance and incident understanding, providing richer insights for security analysts. The alerts enriched with contextual information, such as related SDOs and visualised relationships, provided a clear notion of what led to their generation and why they can be considered significant for the specific environment. This finding underscores that the real strength of CTI lies not only in the information about a threat, after its detection, but also in how it enables a deeper understanding of the threat landscape through contextual awareness.

Together, these insights demonstrate that the true impact of CTI extends well beyond its traditional uses. It can improve situational awareness through structured reasoning and contextual understanding, without losing its effectiveness in helping the incident response process. Above all, one of the most significant facets of CTI is its native ability to adapt to the evolving threat landscape. As its existence is rooted in collective observations and reports from a wide array of entities, its efficacy in reflecting the dynamic nature of cyber threats depends on the general capabilities of the broader community. Timely and contextualised threats reports build the foundation for relevant and valuable Cyber Threat Intelligence, empowering systems that leverage it to quickly adapt and evolve as the cybersecurity landscape evolves. By translating it into machine-actionable data, current cybersecurity operations can evolve to become even more proactive and resilient in the face of emerging threats.

6.3 Technical and Research Implications

From a technical point of view, this dissertation proposes a novel architecture for CTI-driven automated detection and alerting. The implemented POC successfully validates that the proposed architecture can meet the core challenges of integrating external CTI with local data to create a complete detection and alerting pipeline. This establishes a practical foundation for further automation, while maintaining transparency through deterministic logic and interpretable decision-making. The modularity of the system ensures that individual components can be extended or replaced, promoting long-term adaptability.

The work also conceptually contributes to the field of Cyber Threat Intelligence, introducing a structured approach to transform raw intelligence into actionable alerts. It defines concrete hypotheses that are

addressed throughout the design, development, and evaluation of the system. By demonstrating that CTI can be automatically ingested, correlated, and acted upon, the project strengthens the argument for intelligence-driven detection as a viable component of future cybersecurity architectures. This opens avenues for further exploration into more sophisticated techniques based on Cyber Threat Intelligence.

At the research level, this dissertation also provides a reproducible framework that can be used to benchmark and evaluate other future approaches to CTI integration as a detection mechanism. Although at this stage the evaluation efforts focus on validating functionality and feasibility, the transparent structure of the system allows future quantitative assessments, such as measuring accuracy, precision, or scalability when compared to existing solutions. Moreover, the identified challenges and trade-offs offer valuable insights for practitioners and researchers alike, guiding further investigation of adaptive correlation algorithms, trust management, and feedback integration.

In summary, this work contributes to the advancement of both the technical and conceptual aspects of CTI-driven detection and alerting. It provides a replicable POC that operationalises CTI as an active component in threat detection and outlines clear avenues for future research and development in this evolving field.

6.4 Summary

This chapter critically examines the results of the dissertation, exploring the theoretical, technical, and experimental dimensions of the work carried out. The discussion confirmed that CTI-driven threat detection is not only feasible in a technical sense but can also prove beneficial in enhancing proactive measures within cybersecurity workflows.

Beyond this, the chapter also identified key lessons about the operational and conceptual value of Cyber Threat Intelligence. These insights extend its traditional role, highlighting its potential to enrich situational awareness and adaptability, supporting more informed decision-making through contextualised threat detection. This reinforces the central goal of this dissertation that CTI can be effectively operationalised as a central component of threat detection techniques.

The reflections hereby presented establish the transition to the concluding part of this dissertation. Chapter 7 synthesises the main contributions, outlines the main challenges and limitations of the dissertation, and discusses potential avenues for future research and development in the field of CTI.

Chapter 7

Conclusions and Future Work

This final chapter brings together the findings of the research and development process, reflecting both the achievements and the challenges encountered throughout the project. It begins with a summary of the key contributions made by this dissertation. The discussion then transitions to the challenges identified during the course of the work, considering how they may be addressed in future iterations. Finally, the chapter outlines potential directions for future research and practical development, highlighting how the presented concepts may inform or inspire further innovation in the field.

7.1 Summary of Contributions

The conceptual design of the architecture highlights a shift in how CTI can be applied within established cybersecurity workflows. Traditionally, threat intelligence has been used primarily as an enrichment tool, either by complementing existing detection mechanisms by providing additional context to suspicious activities or by supporting analysts in contextualising existing incidents and attacker tactics. This dissertation contributes to the broader field of cybersecurity by presenting an active facet of Cyber Threat Intelligence (CTI), demonstrating its potential as a proactive driver of detection in its own right. By showing that intelligence feeds can be autonomously ingested, normalised, and correlated with local data, the work advances the understanding of how threat intelligence can move beyond solely passive support functions to also become a more prominent component of active threat detection strategies.

The proposed design introduces a CTI-oriented platform that can be integrated with existing workflows to enhance their detection capabilities. This ensures that the value of threat intelligence can be tested and demonstrated without requiring a complete replacement of current security infrastructures, especially when it is considered that it does not aim to substitute existing tools such as SIEMs, SOARs, or IDSs. In this way, the design addresses an ongoing gap in the literature, where the utilisation of CTI is often discussed in theoretical terms but lacks practical and tangible implementations that can be iterated upon

and evaluated in real world scenarios, while acknowledging current limitations and needs in order to reach production readiness standards.

The system also contributes to ongoing discussions on the balance between automation and human expertise in cybersecurity. By focussing on an autonomous detection pipeline that still culminates in analyst-facing interfaces, the architecture highlights the complementary nature of machine-driven and human-driven processes. Analysts are not removed from the decision loop but are instead empowered by the system's ability to filter and prioritise alerts, reducing the need for manual intelligence triage and correlation and allowing them to focus on threat analysis, decision making, and enhancement of the security posture.

The Proof of Concept value lies in showing the feasibility of the overarching goal: that autonomous CTI-driven detection is possible and can generate contextualised alerts based solely on external intelligence and local data. By framing the development objectives to the scope of the project, rather than a leap to performance comparisons with well-established systems, the work sets realistic expectations for its impact and opens avenues for future research to build upon its foundations. These include refinement of trust evaluation in external intelligence, incorporation of adaptive correlation methods to reduce false positives, and deeper integration of analyst feedback into the alerting process.

7.2 Challenges and Limitations

This dissertation aimed to design, implement and evaluate a prototype system that demonstrates the feasibility of autonomously ingesting CTI, correlating it with local telemetry, and generating contextualised alerts without human intervention. The research deliberately focused on the feasibility of the research goals: establishing a modular architecture, validating the core detection pipeline, and demonstrating that CTI can be operationalised as the core of the system. The POC therefore achieves its primary objective by showing that the proposed design functions within a controlled, reproducible environment. At the same time, placing feasibility at the centre of the work imposes explicit boundaries on the claims that are made by this dissertation. The remainder of this section articulates those boundaries, acknowledging challenges faced and how they may be addressed in future work.

The first deliberate limitation is the scale and deployment realism of the POC. The project was validated in a controlled containerised environment with a handful of agents and simulated CTI providers. While this setup is appropriate to demonstrate correctness of the ingestion, correlation, and alerting logic, it does not capture the operational stresses of enterprise-scale deployments. Consequently, quantitative

characteristics such as throughput, latency, and memory footprint remain in prospects for future testing. Real-world environments often involve thousands of endpoints and high-speed telemetry streams which would require resources and optimisations beyond the scope of this project.

Data diversity is the second major challenge. The Proof of Concept was developed and tested using the STIX format for both external CTI and local intelligence. This decision was made to prioritise interoperability and standard compliance, as STIX is widely adopted in the threat intelligence community. However, real-world environments can involve a variety of data formats and sources, including proprietary or legacy systems that may not natively support STIX. Taking into account this, and despite the current implementation not including such extensions, the modular architecture allows for the addition of new collectors and parsers to accommodate different data formats in future iterations.

The user-facing elements and the operational integration were limited by design. The web interface implements core exploration and contextualisation features sufficient to validate the underlying hypotheses. However, in order to reach a production-ready standard, the UI requires enhancements in terms of user experience and native integration with existing security workflows. Features such as Role-Based Access Control (RBAC), data sharing, and bidirectional feedback loops can present valuable enhancements to complement current capabilities.

Finally, and perhaps the biggest challenge to be addressed in future work, is the evaluation of the system's accuracy and effectiveness in real-world scenarios. Not only does this require access to real telemetry and intelligence feeds, but the representativeness of the test scenarios themselves is a challenge. As threats evolve rapidly, the ability to simulate realistic attack patterns and validate them at scale is a significant undertaking. Additionally, comparing with existing detection systems can present challenges on its own due to the different nature of the data sources and detection methodologies. As the system presents a different approach to threat detection, its comparison with existing SIEM or IDS solutions may not be straightforward, and evaluation metrics may need to be adapted to account for the unique characteristics of CTI-driven detection.

7.3 Future Research Directions

Building on the foundations established in this dissertation, several directions emerge for future research and development with the most one being to extend the system beyond the controlled environment used in this work. Scaling the proposed architecture and testing it to handle larger volumes of both external and internal data. This includes benchmarking metrics such as throughput, latency, and reliability, as well as

exploring alternative storage methods and ingestion pipelines to ensure fault tolerance and scalability.

Another key area for enhancement lies in the refinement of the risk assessment model. The current temporal decay and graph-based assessement provides a solid foundation but does intentionally not account for complexities such as behavioural patterns or historical context. Future iterations could explore more complex scoring algorithms that incorporate these factors, potentially leveraging Machine Learning techniques to dynamically adjust risk scores based on the evolution of its threat landscape and operational feedback.

Another promising direction comes from the detection and contextualisation techniques. Behavioural analytics and anomaly detection can enable the identification of previously unseen or obfuscated threats. On the other hand, natural language processing could further aid analysts by augmenting the system's ability to contextualise the generated alerts, provide explanations within the alert context, and recommend common response actions based on historical data and threat intelligence. Although excluded from the present scope due to data and computational constraints, the inclusion of such techniques could transform the system into a more adaptive and intelligent platform.

Finally, further work should focus on improving operational usability and evaluation realism. Enhancing both user and system interface, making the platform more accessible to the end-users, and integrating it with existing security platforms and workflows. In parallel, assessing the system's accuracy and adaptability using real data and live threat intelligence feeds will provide empirical validation under realistic threat landscapes. Collectively, these future efforts would strengthen the system's maturity, further placing Cyber Threat Intelligence as a fundamental component of proactive cybersecurity strategies.

In summary, this dissertation contributes to the field of cybersecurity by reimagining the role of CTI as an autonomous active component of threat detection. It provides a practical framework for integrating external intelligence with local data sources, demonstrating the feasibility of generating actionable alerts without human intervention. By doing so, it not only addresses existing gaps in the literature, but also paves the way for more proactive and adaptive cybersecurity strategies that take advantage of the full potential of threat intelligence.

Bibliography

Lillian Ablon and Andy Bogart. *Zero days, thousands of nights: The life and times of zero-day vulnerabilities and their exploits*. Rand Corporation, 2017.

Mauro Allegretta, Giuseppe Siracusano, Roberto Gonzalez, and Marco Gramaglia. Are crowd-sourced cti datasets ready for supporting anti-cybercrime intelligence? *Computer Networks*, 234:109920, 2023.

HP ArcSight. Common event format. Technical report, tech. rep., July, 2009.

Sean Barnum. Standardizing cyber threat intelligence information with the structured threat information expression (stix). *Mitre Corporation*, 11:1–22, 2012.

Sean Barnum, Robert Martin, Bryan Worrell, and Ivan Kirillov. The cybox language specification. *The MITRE Corporation*, 2012.

Adrien Bécue, Isabel Praça, and João Gama. Artificial intelligence, cyber-threats and industry 4.0: Challenges and opportunities. *Artificial Intelligence Review*, 54(5):3849–3886, 2021.

Leyla Bilge and Tudor Dumitraş. Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 833–844, 2012.

Sheng-Shan Chen, Ren-Hung Hwang, Asad Ali, Ying-Dar Lin, Yu-Chih Wei, and Tun-Wen Pai. Improving quality of indicators of compromise using stix graphs. *Computers & Security*, 144:103972, 2024.

David Chismon and Martyn Ruks. Threat intelligence: Collecting, analysing, evaluating. *MWR InfoSecurity Ltd*, 3(2):36–42, 2015.

Julie Connolly, Mark Davidson, and Charles Schmidt. The trusted automated exchange of indicator information (taxii). *The MITRE Corporation*, pages 1–20, 2014.

World Wide Web Consortium. Logging in w3c httpd, 1995. URL https://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format.

CVE.ICU. Cve publications by year, 2025. URL https://cve.icu/index.html.

cybermindr. The race against exploitation: Average time-to-exploit in 2025, 2025. URL https://www.cybermindr.com/blog/average-time-to-exploit-in-2025/. Accessed: 2025-10-25.

Ricardo M Czekster, Roberto Metere, and Charles Morisset. Incorporating cyber threat intelligence into complex cyber-physical systems: A stix model for active buildings. *Applied Sciences*, 12(10):5005, 2022.

Filigran. Understanding the cyber threat intelligence lifecycle. https://filigran.io/understanding-cyber-threat-intelligence-lifecycle/, 2024. Accessed: 2025-01-27.

Filigran. Opencti. https://filigran.io/solutions/open-cti/, 2025. Accessed: 2025-01-27.

Anissa Frini and Anne-Claire Bourey-Brisset. *An intelligence process model based on a collaborative approach*. Defence R & D Canada, 2011.

Yang Guo. A review of machine learning-based zero-day attack detection: Challenges and future directions. *Computer communications*, 198:175–185, 2023.

Stefan Hagen. Csaf common vulnerability reporting framework (cvrf) version 1.2. OASIS Committee Specification 01, September 2017. URL http://docs.oasis-open.org/csaf/csaf-cvrf/v1.2/cs01/csaf-cvrf-v1.2-cs01.html. Edited by Stefan Hagen.

Ying He, Ellis Inglut, and Cunjin Luo. Malware incident response (ir) informed by cyber threat intelligence (cti). *Science China. Information Sciences*, 65(7):179105, 2022.

IBM. What is a zero-day exploit?, 2025. URL https://www.ibm.com/think/topics/zero-day.

Zafar Iqbal, Zahid Anwar, and Rafia Mumtaz. Stixgen-a novel framework for automatic generation of structured cyber threat information. In *2018 International Conference on Frontiers of Information Technology (FIT)*, pages 241–246. IEEE, 2018.

Bret Jordan, Rich Piazza, and Trey Darley. Stix version 2.1. OASIS Standard, June 2021. URL https://docs.oasis-open.org/cti/stix/v2.1/os/stix-v2.1-os.html. Standards Track Work Product. Copyright © OASIS Open 2022. All Rights Reserved.

Panos Kampanakis. Security automation and threat information-sharing options. *IEEE Security & Privacy*, 12(5):42–51, 2014.

Sherman Kent. Words of estimative probability. 1964.

Inc. LevelBlue. Alienvault otx, 2025. URL https://otx.alienvault.com/. Accessed: 2025-01-24.

Francesco Marchiori, Mauro Conti, and Nino Vincenzo Verde. Stixnet: A novel and modular solution for extracting all stix objects in cti reports. In *Proceedings of the 18th International Conference on Availability, Reliability and Security*, pages 1–11, 2023.

Robert Martin, Steven Christey, David Baker, and MITRE Corporation. The common vulnerabilities and exposures (cve) initiative. *MITRE Corporation*, 2002.

OASIS Open. Oasis tc open repository: Python apis for stix 2, 2025. URL https://github.com/oasis-open/cti-python-stix2. Accessed: 2025-09-24.

Kris Oosthoek and Christian Doerr. Cyber threat intelligence: A product without a process? *International Journal of Intelligence and CounterIntelligence*, 34(2):300–315, April 2021. ISSN 0885-0607. doi: 10.1080/08850607.2020.1780062.

OASIS Open. Introduction to taxii, 2025. URL https://oasis-open.github.io/cti-documentation/taxii/intro.html. Accessed: 2025-01-24.

Daan Planque. Cyber threat intelligence-from confusion to clarity; an investigation into cyber threat intelligence. *Master's thesis*, 2017.

Andrew Ramsdale, Stavros Shiaeles, and Nicholas Kolokotronis. A comparative analysis of cyber-threat intelligence sources, formats and languages. *Electronics*, 9(5):824, 2020.

Langley Rock, Stefan Hagen, and Thomas Schmidt. Common Security Advisory Framework Version 2.0 Errata 01. OASIS Approved Errata, January 2024. URL https://docs.oasis-open.org/csaf/csaf/v2.0/errata01/os/csaf-v2.0-errata01-os.html. Edited by Langley Rock, Stefan Hagen, and Thomas Schmidt.

Farhan Sadique, Sui Cheung, Iman Vakilinia, Shahriar Badsha, and Shamik Sengupta. Automated structured threat information expression (stix) document generation with privacy preservation. In *2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 847–853, 2018a. doi: 10.1109/UEMCON.2018.8796822.

Farhan Sadique, Sui Cheung, Iman Vakilinia, Shahriar Badsha, and Shamik Sengupta. Automated structured threat information expression (stix) document generation with privacy preservation. In *2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, page 847–853, November 2018b. doi: 10.1109/UEMCON.2018.8796822. URL https://ieeexplore.ieee.org/abstract/document/8796822?casa_token=8GIBcSdQ-KkAAAAA:Bfm0wuQcPbFykaJFJa2BRsKoIKIhUCHFIjkFE-mlcrXAd_sK8kYQ28w3eErnn043yS978vkn_TA.

Md Sahrom Abu, Siti Rahayu Selamat, Aswami Ariffin, and Robiah Yusof. Cyber threat intelligence – issue and challenges. *Indonesian Journal of Electrical Engineering and Computer Science*, 10(1):371, April 2018. ISSN 2502-4760, 2502-4752. doi: 10.11591/ijeecs.v10.i1.pp371-379.

Kumar Saurabh, Vaidik Sharma, Uphar Singh, Rahamatullah Khondoker, Ranjana Vyas, and OP Vyas. Hms-ids: Threat intelligence integration for zero-day exploits and advanced persistent threats in iiot. *Arabian Journal for Science and Engineering*, 50(2):1307–1327, 2025.

Daniel Schlette, Fabian Böhm, Marco Caselli, and Günther Pernul. Measuring and visualizing cyber threat intelligence quality. *International Journal of Information Security*, 20(1):21–38, February 2021a. ISSN 1615-5270. doi: 10.1007/s10207-020-00490-y.

Daniel Schlette, Marco Caselli, and Günther Pernul. A comparative study on cyber threat intelligence: The security incident response perspective. *IEEE Communications Surveys & Tutorials*, 23(4):2525–2556, 2021b. ISSN 1553-877X. doi: 10.1109/COMST.2021.3117338.

Daniel Schlette, Manfred Vielberth, and Günther Pernul. Cti-soc2m2–the quest for mature, intelligence-driven security operations and incident response capabilities. *Computers & Security*, 111:102482, 2021c.

SigmaHQ. Introducing sigma specification v2.0, 2025. URL https://blog.sigmahq.io/introducing-sigma-specification-v2-0-25f81a926ff0. Accessed: 2025-11-28.

StrangeBee. Thehive. https://strangebee.com/, 2025. Accessed: 2025-01-27.

The Linux Foundation Projects. osquery/osquery: Performant endpoint visibility, 2025. URL https://osquery.io/. Accessed: 2025-09-24.

The MISP Project. Misp: Malware information sharing platform, 2025. URL https://www.misp-project.org/. Accessed: 2025-01-24.

The MITRE Corporation. Cve-2024-3094, 2024. URL https://www.cve.org/CVERecord?id=CVE-2024-3094.

Antonio Villalón-Huerta, Ismael Ripoll-Ripoll, and Hector Marco-Gisbert. Key requirements for the detection and sharing of behavioral indicators of compromise. *Electronics*, 11(33):416, January 2022. ISSN 2079-9292. doi: 10.3390/electronics11030416.

Cynthia Wagner, Alexandre Dulaunoy, Gérard Wagener, and Andras Iklody. Misp: The design and implementation of a collaborative threat intelligence sharing platform. In *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*, pages 49–56, 2016.

Thomas D. Wagner, Khaled Mahbub, Esther Palomar, and Ali E. Abdallah. Cyber threat intelligence sharing: Survey and research directions. *Computers & Security*, 87:101589, November 2019. ISSN 0167-4048. doi: 10.1016/j.cose.2019.101589.

Roman Wirtz and Maritta Heisel. Cvss-based estimation and prioritization for security risks. In *ENASE*, pages 297–306, 2019.

YesWeHack. Cve surge: Why the record rise in new vulnerabilities?, january 2025. URL https://www.yeswehack.com/news/cve-surge-record-jump-vulnerabilities.

Tommaso Zoppi, Andrea Ceccarelli, and Andrea Bondavalli. Unsupervised algorithms to detect zero-day attacks: Strategy and application. *Ieee Access*, 9:90603–90615, 2021a.

Tommaso Zoppi, Mohamad Gharib, Muhammad Atif, and Andrea Bondavalli. Meta-learning to improve unsupervised intrusion detection in cyber-physical systems. *ACM Transactions on Cyber-Physical Systems (TCPS)*, 5(4):1–27, 2021b.