LUISS

Department of Business and Management

Course of: Cybercrime and Fraud Detection

The Impact of Artificial Intelligence on OSINT Technologies

SUPERVISOR

Prof. Gianluigi Me

CANDIDATE

Eliya Allam

ID: 289791

Academic Year: 2024/2025

ACKNOWLEDGEMENTS

I would like to express my sincere and profound gratitude to the individuals who have supported and guided me throughout the journey of completing this Bachelor's thesis.

First and foremost, I extend my deepest appreciation to my thesis supervisor, Professor Gianluigi Me. His unwavering support, insightful guidance, and continuous encouragement at every stage of this research have been invaluable. Professor Me not only challenged me to strive for excellence but also played a pivotal role in igniting my passion for cybersecurity through his engaging subject, "Cybercrime and Fraud Detection". His mentorship has been instrumental in the development of this work.

I am also grateful to LUISS University for providing an environment conducive to learning and research, offering the resources and academic framework that made this thesis possible.

On a more personal note, I owe an immeasurable debt of gratitude to my mother, who has been my number one supporter since the very beginning, offering unconditional love and encouragement throughout all of my endeavors. Her belief in me has been a constant source of strength. I also carry with me the enduring support and inspiration of my late father, whose memory continues to motivate me.

To my closest friends, thank you for your steadfast presence and understanding, especially during the more demanding periods of this undertaking. Your camaraderie has been a much-appreciated anchor.

Finally, my heartfelt thanks go to my partner, whose unwavering support, patience, and encouragement have been a constant source of motivation and comfort throughout this entire process.

This thesis is a culmination of not only my efforts but also the collective support and belief of these wonderful individuals.

ABSTRACT

The proliferation of publicly available information presents both significant opportunities and complex challenges for Open-Source Intelligence (OSINT) in the cybersecurity domain. Existing OSINT tools, however, often exhibit critical limitations in analytical depth, automation, cross-source correlation, and adaptability. This thesis addresses these deficiencies by proposing, designing, and implementing OSINT CyberVision, an advanced proof-of-concept platform that leverages cutting-edge Artificial Intelligence (AI) to revolutionize security intelligence gathering and analysis.

The core objective was to demonstrate the practical efficacy of integrating Large Language Models (LLMs), Retrieval-Augmented Generation (RAG), and autonomous agent-based architectures within a cohesive OSINT system. OSINT CyberVision was developed with a modular, layered architecture, incorporating custom data collection for diverse cybersecurity sources (NVD, MITRE ATT&CK, ArXiv), a specialized knowledge base with semantic embedding, a custom RAG pipeline utilizing Claude 3.7 Sonnet for grounded contextual reasoning, and a ReAct-style agent framework for autonomous multi-step analysis and tool utilization.

Evaluation of the OSINT CyberVision PoC demonstrated significant enhancements over traditional OSINT approaches. The system achieved high factual accuracy (92%) and low hallucination rates (4%) in its AI-generated responses, effective retrieval quality (P@5 0.83), and a notable task completion rate (89%) for its autonomous agent. Case studies further illustrated its practical application in vulnerability analysis, threat actor profiling, and indicator of compromise investigation, showcasing improved analytical depth and automation.

This research contributes a novel AI-enhanced OSINT system architecture, a functional proof-of-concept demonstrating the synergistic benefits of LLMs, RAG, and agents, and an empirical validation of their potential to address long-standing OSINT limitations. OSINT CyberVision establishes a new paradigm for OSINT, offering a pathway towards more intelligent, efficient, and proactive cybersecurity intelligence capabilities.

TABLE OF CONTENTS

LIST OF TABLES

Chapter 1

INTRODUCTION

1.1 Background and Problem Statement

Open-Source Intelligence (OSINT) involves the methodical collection, processing, and analysis of publicly available information to produce actionable intelligence. The rapidly evolving threat landscape necessitates that security professionals efficiently gather, process, and analyze intelligence from a multitude of diverse sources, where OSINT plays a crucial, indispensable role (Yadav et al., 2023). Historically, OSINT has undergone a significant transformation, particularly with the advent of the digital age. While pre-digital OSINT focused on tangible events and public declarations, the proliferation of the Internet and the World Wide Web has reshaped its landscape, with information now predominantly accessible online and retrieval methods growing in sophistication (Yadav et al., 2023). This evolution is particularly evident in the context of cybersecurity, where OSINT provides vital insights for understanding and mitigating threats.

Despite its importance, the practical application of OSINT is often hampered by significant limitations inherent in currently available tools. A comprehensive analysis of leading OSINT solutions, conducted as part of the foundational research for this thesis (detailed further in Section 2.4), reveals widespread shortcomings, particularly in areas critical for effective intelligence operations. Key findings from this market analysis indicate that many existing tools exhibit deficiencies in automation capabilities, lack analytical depth, possess inadequate threat identification mechanisms, and struggle with cross-source correlation. Furthermore, a majority of these tools are constrained by technical limitations that restrict their extensibility and adaptability to the dynamic nature of modern threats (Pastor-Galindo et al., 2020). These challenges collectively impede the ability of security professionals to fully leverage the potential of open-source data.

The advent of advanced Artificial Intelligence (AI) techniques, particularly Large Language Models (LLMs), Retrieval-Augmented Generation (RAG), and agent-based architectures, presents a transformative opportunity to address these identified deficiencies in the OSINT domain. These AI paradigms offer capabilities in natural language understanding, complex reasoning, and autonomous task execution that can significantly enhance OSINT workflows (Hasanov et al., 2024). However, a critical problem persists: the majority of current OSINT tools have not yet effectively integrated these cutting-edge AI methodologies. This results in a distinct market gap for a comprehensive, AI-enhanced OSINT platform capable of overcoming the traditional limitations related to automation, analytical depth, and correlation. This thesis introduces OSINT CyberVision, a novel platform designed to bridge this gap. OSINT CyberVision aims to establish a new paradigm for security intelligence gathering and analysis by systematically leveraging LLMs, RAG, and agent-based systems to enhance automation, deepen analytical insights, improve cross-source correlation, and strengthen threat identification capabilities.

1.2 Research Objectives and Contributions

The primary objective of this thesis is to address the identified critical market need for a more advanced and integrated Open-Source Intelligence solution.

This research aims to design, implement, and evaluate OSINT CyberVision, a comprehensive cybersecurity intelligence platform that systematically leverages cutting-edge Artificial Intelligence (AI) techniques—specifically Large Language Models (LLMs), Retrieval-Augmented Generation (RAG), and agent-based architectures—to overcome the prevalent limitations in existing OSINT tools. The goal is to establish a new paradigm for security intelligence gathering and analysis, enhancing automation, analytical depth, cross-source correlation, and threat identification capabilities.

To achieve this overarching aim, the research pursues the following specific objectives:

1. To design a modular and extensible high-level architecture for an AI-enhanced OSINT system capable of ingesting, processing, and analyzing diverse intelligence sources.

2. To investigate and implement the effective integration of LLMs for contextual understanding and reasoning, RAG pipelines for grounding responses in factual knowledge,

and autonomous agent frameworks for complex, multi-step intelligence processing within an OSINT context.

3. To develop and demonstrate specific system capabilities that directly address the shortcomings of current OSINT tools, including enhanced automation of intelligence collection and analysis workflows, deeper analytical processing of unstructured and structured data, robust cross-source correlation of intelligence fragments, and more effective threat identification and assessment.

4. To evaluate the performance and efficacy of the developed OSINT CyberVision platform through quantitative metrics and qualitative case studies, comparing its capabilities against the identified limitations of existing OSINT solutions.

This thesis makes several key contributions to the field of Open-Source Intelligence and AI-driven cybersecurity:

1. **A Novel AI-Enhanced OSINT System Architecture:** It proposes and details the architecture of OSINT CyberVision, a platform that cohesively integrates LLMs, RAG, and agent-based processing. This design serves as a blueprint for future AI-driven OSINT systems, addressing how these distinct AI components can work synergistically.

2. **Demonstration of AI Efficacy in Addressing OSINT Gaps:** The research provides a practical demonstration of how specific AI techniques can overcome long-standing limitations in OSINT tools. This includes showcasing enhanced automation in intelligence gathering, improved analytical depth through LLM-powered reasoning, superior cross-source correlation facilitated by RAG and agent interactions, and more nuanced threat identification. This work directly contributes to a future research direction highlighted by Yadav et al. (2023) concerning the automation of information gathering and intelligence extraction using artificial intelligence.

3. **A Functional Proof-of-Concept Implementation:** OSINT CyberVision is implemented as a functional proof-of-concept, providing tangible evidence of the feasibility and benefits of the proposed AI-enhanced approach. This implementation

includes modules for multi-source data collection, advanced knowledge processing, AI-enhanced analysis, agent-based intelligence processing, and a natural language interface.

4. **Comprehensive Evaluation and Comparative Insights:** The thesis presents an evaluation of the OSINT CyberVision platform, offering insights into its performance across various OSINT tasks. This includes metrics on retrieval quality, response quality, and agent performance, alongside case studies that illustrate its practical applications in scenarios like vulnerability analysis and threat actor profiling. This evaluation provides a benchmark for future AI-OSINT systems and offers a comparative perspective against the capabilities of traditional tools.

1.3 Thesis Structure Overview

This thesis is organized into seven chapters, designed to systematically present the research, design, implementation, and evaluation of the OSINT CyberVision platform.

Chapter 1: Introduction lays the groundwork by discussing the background of Open-Source Intelligence, identifying the problem statement concerning the limitations of current OSINT tools, and outlining the research objectives and contributions of this work.

Chapter 2: Literature Review and State-of-the-Art Analysis provides a comprehensive review of existing literature. It covers the evolution and limitations of OSINT, explores the application of Large Language Models in security intelligence, examines advanced AI techniques such as Retrieval-Augmented Generation and agent-based architectures, and concludes with a detailed gap analysis of current OSINT solutions, thereby contextualizing the need for OSINT CyberVision.

Chapter 3: System Design and Architecture details the foundational design of the OSINT CyberVision platform. This chapter discusses the system requirements and operational constraints, presents the high-level architecture and data flow, elaborates on the technology stack selection with its rationale, and highlights key architectural decisions made during the design phase.

Chapter 4: Implementation describes the practical development of the OSINT CyberVision proof-of-concept. It covers the setup of the OSINT sources framework and data collection mechanisms, the implementation of the knowledge base and vector database, the integration of the selected LLM and the RAG pipeline, the development of the agent framework and associated tools, and the creation of the user interfaces.

Chapter 5: Evaluation and Results presents the empirical assessment of the OSINT CyberVision platform. This chapter outlines the evaluation methodology and metrics used, analyzes the system's performance in various OSINT tasks, showcases its practical applications through illustrative case studies, and provides a comparative analysis against existing solutions based on the identified gaps.

Chapter 6: Discussion and Future Work offers an in-depth analysis of the research findings and implementation experiences. It discusses how the results address the identified OSINT limitations, reflects on the challenges and constraints encountered during development, and proposes future research directions and potential enhancements for the OSINT CyberVision platform.

Chapter 7: Conclusion summarizes the principal contributions and achievements of this thesis. It reiterates the significance of the research in advancing AI-driven OSINT capabilities and offers closing remarks on the broader implications of this work for the field of cybersecurity intelligence.

The thesis also includes appendices detailing component references, configuration options, system usage guidelines, and example prompts and responses to further support the presented work.

Chapter 2

LITERATURE REVIEW AND STATE-OF-THE-ART ANALYSIS

This chapter provides a focused review of the existing literature and an analysis of the current state-of-the-art relevant to the development of an advanced, AI-enhanced Open-Source Intelligence (OSINT) platform. It begins by concisely examining the evolution and core challenges of OSINT. Subsequently, the chapter explores the pivotal role of Large Language Models (LLMs) in security intelligence, including their capabilities and inherent limitations. The discussion then shifts to advanced AI techniques, specifically Retrieval-Augmented Generation (RAG) and agent-based architectures, as foundational components for next-generation OSINT. The chapter culminates in a gap analysis of current OSINT solutions, underscoring the necessity for the OSINT CyberVision platform.

2.1 Open-Source Intelligence (OSINT): Evolution and Limitations

Open-Source Intelligence (OSINT) is the discipline of collecting, processing, and analyzing publicly available information to produce actionable intelligence (Pastor-Galindo et al., 2020; Yadav et al., 2023). Its scope has dramatically expanded with the digital revolution, moving from traditional media to the vast datasphere of the internet, including social media, public records, and technical databases. This evolution presents both immense opportunities and significant challenges for intelligence practitioners across domains such as cybersecurity, law enforcement, and national security (Yadav et al., 2023).

The primary benefits of OSINT include the sheer volume of accessible data, the increasing power of computational tools for analysis, and its versatility across diverse applications (Pastor-Galindo et al., 2020). However, the effective utilization of OSINT is often hindered by several persistent challenges. These include the overwhelming *volume and velocity* of data, the predominantly *unstructured nature* of much of this information, the pervasive issues of *misinformation and source reliability*, and complex *ethical and legal considerations* regarding privacy and data handling (Pastor-Galindo et al., 2020).

Furthermore, a critical review of contemporary OSINT tools, conducted as part of the foundational research for this thesis (detailed in Section 2.4), reveals systemic limitations. Most current tools lack sophisticated *analytical depth*, offer only modest *automation capabilities*, struggle with effective *cross-source correlation*, possess basic *threat identification* features, and are often constrained by *technical inflexibility* that limits their adaptability and extensibility. These shortcomings highlight a significant gap between the potential of OSINT and the capabilities of existing practitioner tools, a gap this thesis aims to address through the application of advanced AI.

In the view of cybersecurity and intelligence analysts, OSINT today is useful but also has many practical difficulties. Most analysts spend much time and effort in collecting data from several different tools, each with its own structure and rules. Following, having to go through lots of largely unorganized, buzzing and occasionally questionable data to sort out important details and small linkages is not easy for intelligence analysts. Also, because automated tools are limited in complex investigative processes, many important tasks such as multi-level analysis or constant monitoring still require employees doing the work which in turn affects the amount of intelligence that can be gathered and how quickly it is done. OSINT CyberVision is meant to tackle these specific difficulties by introducing an integrated platform. It helps users gather, examine and associate data from many sources using AI, make analysts more productive and overall, make it simpler for analysts to conduct OSINT.

2.2 Large Language Models in Security Intelligence

Large Language Models (LLMs) have emerged as a potentially transformative technology within the cybersecurity landscape, offering new avenues for enhancing security intelligence operations. The rapid advancements in LLMs, such as GPT-4 (OpenAI et al., 2024) and various open-source alternatives (Touvron et al., 2023), have spurred significant research into their application for both defensive and offensive security tasks (Hasanov et al., 2024).

Capabilities and Applications

LLMs bring powerful natural language understanding and generation capabilities to security intelligence. They can process and analyze vast amounts of unstructured textual data common in OSINT, such as threat reports, forum discussions, and technical documentation. Key applications in the broader security domain, which inform their potential utility in OSINT, include: enhanced threat detection through pattern recognition in diverse data streams; automated analysis and interpretation of security logs; assistance in vulnerability assessment by parsing advisories and technical details; and the generation of realistic phishing simulations or security reports (Bhusal et al., 2024; Hasanov et al., 2024). Recent trends, identified during the preparatory research for this thesis, also point towards the development of LLM-powered autonomous OSINT agents capable of complex intelligence gathering and reasoning workflows.

Challenges and Limitations

Despite their promise, the deployment of LLMs in security-sensitive contexts is not without significant challenges. A primary concern is their susceptibility to *hallucinations* and generating factually incorrect information, a critical issue where accuracy is paramount (Bhusal et al., 2024; Hasanov et al., 2024). The *dual-use nature* of LLMs means their capabilities can be exploited for malicious activities, such as crafting sophisticated disinformation or aiding in attack planning (Hasanov et al., 2024). Furthermore, *model security* itself is a concern, with vulnerabilities like prompt injection posing risks (Hasanov et al., 2024). Other limitations include challenges in *interpretability*, potential for inherited *biases* from training data, *data privacy concerns* associated with API-based models, and the considerable *computational cost* and resource requirements for their operation (Hasanov et al., 2024). Addressing these limitations through robust evaluation, ethical guidelines, and technical safeguards is crucial for their reliable application in security intelligence and OSINT.

2.3 Advanced AI Techniques: RAG and Agent Architectures

Beyond the foundational capabilities of Large Language Models, two advanced AI paradigms are particularly pertinent to overcoming the identified limitations in OSINT and form core components of modern AI-enhanced systems: Retrieval-Augmented Generation (RAG) and agent-based architectures. These techniques enable LLMs to access external knowledge dynamically and perform complex, multi-step reasoning and action sequences, respectively.

Retrieval-Augmented Generation (RAG) in OSINT

Retrieval-Augmented Generation (RAG) has emerged as a powerful technique to enhance the factual accuracy, relevance, and timeliness of information generated by LLMs (Gao et al., 2024). The core principle of RAG involves equipping an LLM with the ability to retrieve relevant information from an external knowledge base before generating a response. This contrasts with standard LLMs that rely solely on the knowledge implicitly encoded in their parameters during pre-training, which can become outdated or lack domain-specific depth (Lewis et al., 2021).

A typical RAG system comprises two main components: a retriever and a generator (Gao et al., 2024). The retriever searches a corpus of documents to find relevant passages, which are then provided as context to the generator (an LLM) to produce an informed response.

The benefits of RAG for OSINT applications are manifold. It directly addresses potential LLM hallucinations by grounding responses in verifiable external data, crucial for the veracity and timeliness required in intelligence work. It also allows for the incorporation of dynamic, domain-specific knowledge. In the development of OSINT CyberVision, the design considerations for its RAG architecture emphasized diverse source integration, intelligent document chunking, adaptive embedding models for cybersecurity terminology, and contextual retrieval mechanisms. While frameworks like LangChain were evaluated for their comprehensive RAG ecosystems and strong integration capabilities, the core RAG pipeline implemented in OSINT CyberVision (as detailed in Chapter 4) involves custom-developed components for retrieval and prompt management, working in conjunction with direct LLM API calls for generation.

Agent-Based Architectures for Autonomous Intelligence Processing

Agent-based architectures, particularly those powered by LLMs, represent another significant advancement for enhancing OSINT capabilities, especially in terms of automation and complex task execution. An LLM-based agent is a system that utilizes an LLM as its core reasoning engine to perceive its environment, make decisions, and take actions to achieve specific goals. These agents can interact with various tools, access external APIs, and perform multi-step tasks that mimic human analytical workflows.

In the context of OSINT, LLM agents can automate complex intelligence gathering and analysis processes. For instance, an agent could be tasked with investigating a potential threat actor. It might autonomously decide to query multiple OSINT data sources, use tools to extract entities and relationships, correlate findings, and generate a profile, all while reasoning about its next steps and adapting its strategy based on the information it uncovers. The ReAct (Reason+Act) paradigm, which synergizes reasoning and acting in language models, is an influential approach in this domain, enabling models to generate both reasoning traces and task-specific actions in an interleaved manner (Yao et al., 2023).

The development of the agent framework for OSINT CyberVision identified several key requirements for effective autonomous intelligence processing. These included the capability for agents to perform specialized intelligence tasks (such as network reconnaissance or vulnerability assessment), mechanisms to support potential multi-agent collaboration, the ability to make autonomous decisions regarding intelligence gathering priorities, adaptive learning capabilities based on feedback, and the provision of explainable actions to ensure transparency and accountability. The LangChain Agents framework was chosen for OSINT CyberVision due to its robust support for these functionalities, notably its ReAct-style reasoning and standardized tool integration, directly addressing the limitations in automation and analytical depth found in many traditional OSINT tools.

The combination of RAG for knowledge grounding and LLM-based agents for autonomous reasoning and action provides a powerful foundation for developing next-generation OSINT systems capable of handling the complexity and dynamism of the modern intelligence landscape.

2.4 Gap Analysis in Current OSINT Solutions

The preceding review of OSINT principles, LLM capabilities, and advanced AI techniques sets the stage for a critical analysis of existing OSINT tools. As established in Section 2.1, while OSINT as a discipline holds immense potential, the tools currently available to practitioners often fall short of leveraging this potential, particularly in the context of modern data volumes and the sophisticated analytical capabilities offered by AI. To provide a concrete basis for this assertion and to clearly define the specific areas for improvement that OSINT CyberVision targets, a comprehensive evaluation of eight leading OSINT tools was conducted as part of the foundational research for this thesis.

The evaluation assessed each tool against nine critical operational dimensions: Data Sources Supported, Automation Capabilities, Analysis Features, Cross-Source Correlation, Threat Identification, Data Integration Options, User Interface/Experience, Technical Limitations, and Development Status. Each dimension was scored on a scale of 1 (basic capability) to 5 (advanced capability). The detailed comparison matrix is presented in Table 2.1.

Table 2.1: Comparative Analysis of Leading OSINT Tools

Tool Name	**Data Sources**	**Automation**	**Analysis**	**Cross-Source Correlation**	**Threat ID**	**Data Integration**	**User Interface**	**Technical Limitations**	**Development Status**
OSINT Framework	3	1	1	1	1	2	3	1	3
Google Dorks	2	1	1	1	1	1	2	1	5
theHarvester	3	2	2	2	2	3	3	2	4
SecurityTrails API	4	3	3	3	3	4	4	2	5
BGPView	3	2	2	2	2	2	3	2	3
Mitaka	3	2	2	2	2	3	4	2	4
Shodan	4	4	4	4	4	4	4	2	5
SpiderFoot	5	4	4	4	4	5	4	2	5
Average	**3.38**	**2.38**	**2.38**	**2.38**	**2.38**	**3.00**	**3.38**	**1.75**	**4.25**

As Table 2.1 illustrates, while certain tools like SpiderFoot and Shodan exhibit strengths in areas such as data source diversity and user interface, the average scores across several critical dimensions highlight significant systemic weaknesses in the current OSINT tool landscape. The analysis reveals five primary gap areas that impede the full realization of OSINT's potential:

1. **Pervasive Technical Limitations (Average Score: 1.75/5):** This is the most strik-

ing deficiency. Existing tools consistently suffer from constraints in extensibility, customization, and adaptability. This severely limits their capacity to integrate new data sources without vendor intervention, process diverse unstructured data types effectively, or be easily modified to meet evolving intelligence requirements. These limitations create operational blind spots and hinder the agility needed in dynamic threat environments.

2. **Insufficient Analytical Depth (Average Score: 2.38/5):** Most tools provide only rudimentary analytical features, often requiring extensive manual interpretation by human analysts. There is a distinct lack of capabilities for advanced semantic understanding of intelligence context, poor identification of subtle patterns across data points, minimal support for hypothesis generation, and an absence of sophisticated reasoning engines to draw non-obvious conclusions. This places a heavy burden on human analysts and can lead to delays or missed insights.

3. **Modest Automation Capabilities (Average Score: 2.38/5):** The level of automation in current OSINT tools is generally low, with substantial manual effort still required for many intelligence processes. This includes the limited autonomous execution of complex, multi-stage intelligence gathering workflows, insufficient automation of routine analytical tasks, and poor handling of dynamic follow-up inquiries based on initial findings. This lack of automation reduces analyst productivity and restricts the scalability of OSINT operations.

4. **Weak Cross-Source Correlation (Average Score: 2.38/5):** A fundamental challenge is the inability of most tools to effectively correlate intelligence fragments from diverse sources. This leads to the creation of intelligence silos, where connections between related pieces of information from different origins are missed. The absence of robust mechanisms for detecting corroborating or contradicting information, and minimal semantic understanding of inter-entity relationships, prevents the development of comprehensive intelligence pictures.

5. **Basic Threat Identification (Average Score: 2.38/5):** The capabilities for recognizing, categorizing, and prioritizing potential threats are often elementary. Many

tools show limited identification of emerging threat patterns, insufficient understanding of threat actor behaviors and motivations, and poor predictive capabilities for evolving threats. This restricts the ability of organizations to anticipate and prepare for novel attack vectors, limiting proactive security postures.

In addition to these core functional gaps, the evaluation also highlighted that many existing tools inadequately address emerging OSINT challenges such as information overload, source credibility assessment, deepfake content detection, language barriers, and comprehensive Dark Web coverage.

These identified gaps collectively underscore a critical market need for a new generation of OSINT platforms. Such a platform must move beyond basic data aggregation and offer integrated, AI-driven capabilities for deep analysis, robust automation, effective cross-source correlation, and sophisticated threat intelligence. The OSINT CyberVision system, as proposed in this thesis, is designed specifically to address these deficiencies by leveraging the power of LLMs, RAG, and agent-based architectures.

Chapter 3

SYSTEM DESIGN AND ARCHITECTURE

This chapter outlines the comprehensive design and architectural considerations for an advanced, AI-enhanced Open-Source Intelligence (OSINT) system. It begins by defining the system requirements and operational constraints, largely derived from the limitations of existing OSINT tools identified in Chapter 2. Subsequently, the high-level architecture and fundamental data flow of the envisioned system are presented. This is followed by a detailed discussion of the technology stack selection process and the rationale behind key architectural decisions. While this chapter describes the architecture for a comprehensive OSINT platform, it also sets the context for the proof-of-concept system, OSINT CyberVision, whose specific implementation is detailed in Chapter 4.

3.1 System Requirements and Constraints

The development of a next-generation OSINT platform necessitates a clear understanding of both its desired functionalities and the environment in which it must operate. The requirements for such a system are directly informed by the shortcomings of current OSINT tools and the potential of AI to address these gaps.

Deriving Requirements from OSINT Limitations

As established in Chapter 2 (Section 2.4), contemporary OSINT tools exhibit significant deficiencies across several key areas, including pervasive technical limitations, insufficient analytical depth, modest automation capabilities, weak cross-source correlation, and basic threat identification features. An advanced, AI-enhanced OSINT system must therefore be designed to specifically overcome these shortcomings. The primary drive for the system design detailed herein is to create a platform that is extensible, analytically powerful, highly automated, capable of sophisticated data fusion, and adept at nuanced threat intelligence. The integration of Artificial Intelligence, particularly LLMs, RAG, and agent-based architectures, is central to achieving these objectives.

Functional Requirements

Based on the analysis of OSINT needs and the potential of AI, the envisioned comprehensive OSINT platform is designed to meet the core functional requirements summarized in Table 3.1. These requirements guided the broader architectural vision, aspects of which are demonstrated by the OSINT CyberVision proof-of-concept.

Table 3.1: Key Functional Requirements for the Envisioned OSINT Platform

Requirement Category	**Core Objective / Description**
Intelligence Collection	Ingest and process diverse OSINT sources (structured/unstructured, multilingual); automated extraction of entities, relationships, and indicators.
Analysis Capabilities	Advanced semantic understanding; cross-source correlation; identification of non-obvious relationships; temporal analysis; automated reasoning for attribution.
User Interaction	Intuitive natural language query interface; support for complex questions; progressive disclosure; explanation of reasoning and source attribution; feedback mechanisms.
Agent Functionality	Autonomous execution of intelligence workflows; self-directed exploration of threads; multi-step reasoning for complex analysis; tool integration.
Knowledge Management	Dynamic knowledge base with incremental updates; confidence scoring for reliability; versioning of intelligence; metadata enrichment for enhanced retrieval.

These functional requirements aim to create a system that not only gathers intelligence more broadly and efficiently than current tools but also provides significantly deeper analytical power and user-centric interaction.

Non-Functional Requirements

Beyond core functionalities, the envisioned system must also satisfy critical non-functional requirements to ensure its effectiveness, reliability, and practicality in operational environments. These are summarized in Table 3.2.

Adherence to these non-functional requirements is essential for building a system that is

Table 3.2: Key Non-Functional Requirements for the Envisioned OSINT Platform

Requirement	**Target Metric / Description**
Performance	Sub-5-second standard query response; 10,000+ documents processed/indexed daily; support for 50+ concurrent users; graceful degradation under load.
Security	End-to-end encryption; role-based access control; comprehensive audit logging; secure API authentication; compliance with data protection regulations.
Extensibility	Modular architecture for component replacement; API-first design for tool integration; plugin system for new data sources; custom agent tool development support.
Reliability	Target 99.9% system availability; fault tolerance for component failures; automated backup/recovery; graceful handling of API rate limits/outages.
Usability	Intuitive interface with minimal training; comprehensive documentation/help; consistent interaction patterns; accessibility compliance (e.g., WCAG 2.1).

not only powerful but also trustworthy, maintainable, and user-friendly.

Project Constraints and Proof-of-Concept Scope

The functional and non-functional requirements outlined above describe an ideal, comprehensive AI-enhanced OSINT platform. However, the development undertaken for this thesis project, resulting in the OSINT CyberVision system, operates as a proof-of-concept (PoC) under typical academic research constraints, including limited time, resources, and development manpower.

Consequently, OSINT CyberVision focuses on implementing and demonstrating the core architectural principles and key AI-driven innovations of the envisioned system, rather than its full, production-scale breadth. For instance, while the "Grand Design" anticipates integration with a vast array of data sources (e.g., 49+ identified during initial research), the OSINT CyberVision PoC utilizes a smaller, representative set of three diverse sources (National Vulnerability Database, MITRE ATT&CK framework, and ArXiv research publications) to validate its data collection, processing, and analysis capabilities. Similarly, while the "Grand Design" anticipates a robust, distributed, and highly scalable vector database solution (such as Milvus or PostgreSQL with the pgvector extension) for managing and searching text embeddings efficiently at scale, the OSINT CyberVision PoC implements its vector storage and search capabilities using a more direct, file-based approach. In this PoC, text embeddings generated by sentence-transformer models are stored (e.g., within JSON structures) and retrieved via custom mechanisms.

3.2 High-Level Architecture and Data Flow

The envisioned AI-enhanced OSINT system is designed with a modular, layered architecture to ensure separation of concerns, maintainability, and scalability. This architecture provides a robust framework capable of supporting the diverse functional and non-functional requirements outlined previously. OSINT CyberVision, the proof-of-concept developed for this thesis, implements key components within this architectural paradigm.

Architectural Layers

The high-level architecture comprises six distinct, interacting layers, each responsible for specific aspects of the intelligence lifecycle:

- **Data Collection Layer:** Responsible for the ingestion of raw OSINT data from diverse external sources (e.g., APIs, web pages, documents). This layer includes mechanisms for data retrieval, initial parsing, and normalization. In OSINT CyberVision, this layer focuses on the selected NVD, MITRE ATT&CK, and ArXiv sources.
- **Knowledge Base Layer:** This is the central repository for processed intelligence. It manages document storage, performs sophisticated chunking of content for optimal retrieval, generates vector embeddings to capture semantic meaning, and maintains metadata for filtering and faceted search.
- **LLM Service Layer:** This layer encapsulates all interactions with the selected Large Language Model (e.g., Claude 3.7 Sonnet for OSINT CyberVision). It handles API

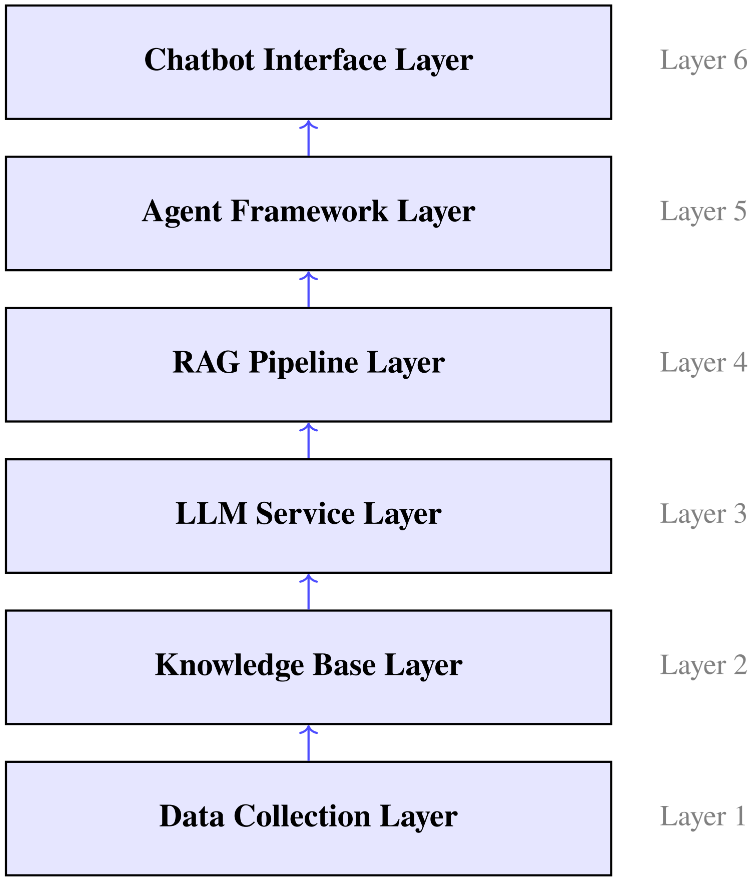


Figure 3.1: High-Level System Architecture

communication, prompt engineering strategies, and context window management to leverage the LLM's reasoning and generation capabilities.

- **RAG Pipeline Layer:** Enhances the LLM's responses by dynamically incorporating relevant context retrieved from the Knowledge Base Layer. This layer implements the retrieve-augment-generate cycle, crucial for grounding LLM outputs in factual, up-to-date information and mitigating hallucinations.

- **Agent Framework Layer:** Enables autonomous intelligence processing through LLM-powered agents. This layer orchestrates agent workflows, manages specialized tools for OSINT tasks, and implements planning and reasoning capabilities for complex, multi-step analyses.

- **Chatbot Interface Layer:** Serves as the primary interaction point for human analysts. It provides natural language query capabilities (via web and CLI interfaces in OSINT CyberVision), presents formatted responses with source attribution, and manages user session context.

This layered approach promotes modularity, allowing individual layers to be developed, updated, or scaled independently.

Core Data Flow

The system processes information through several defined data flows, ensuring a consistent and traceable intelligence lifecycle. The two primary flows are Document Ingestion and Query Processing, with a specialized Agent Workflow for more complex tasks.

1. **Document Ingestion Flow:** External OSINT sources are accessed by Data Collectors. Parsers convert various document formats (PDFs, HTML, text) into normalized text. Preprocessors then clean and standardize this content. Entity Extractors identify key entities and relationships, enriching the data. Chunkers segment the documents into optimal retrieval units. Embedders convert these chunks into vector representations. Finally, the processed intelligence, along with its embeddings and metadata, is stored in the Knowledge Base.

2. **Query Processing Flow (RAG-based):** A user submits a natural language query via the User Interface. The UI captures the query and user context. A Query Processor analyzes the query's intent and may reformulate it. The Retrieval Engine then queries the Knowledge Base (using vector search and metadata filtering) to identify relevant intelligence fragments. A Context Constructor assembles an optimal context from these fragments. This context, along with the original query, is passed to the LLM Service, which generates a response. A Response Generator formats this output,

including citations and confidence levels, before it is presented back to the user via the UI with interactive elements.

3. **Agent Workflow Flow:** For complex queries identified by the Query Processor as requiring agent handling, the Agent Orchestrator initiates an appropriate agent workflow. A Planning Engine within the agent develops a multi-step plan. A Tool Selector identifies appropriate tools for each step from the Tool Registry. The Tool Executor runs these tools with necessary parameters, and a Result Integrator combines tool outputs. Finally, the LLM Service synthesizes a final response, incorporating agent findings, which is then presented to the user.

A feedback mechanism is also envisioned, where user interactions and feedback on system responses are collected by a Feedback Collector, analyzed by a Performance Analyzer, and used by a Model Optimizer and Knowledge Enhancer to iteratively improve system performance, prompt strategies, and retrieval effectiveness.

3.3 Technology Stack Selection and Rationale

The development of an advanced AI-enhanced OSINT system requires judicious technology choices, balancing cutting-edge capabilities with practical implementation. For the envisioned comprehensive platform (the "Grand Design"), selections would prioritize enterprise-grade scalability, robustness, and advanced feature sets across all components, from data ingestion and knowledge management to AI model integration and user interaction.

The OSINT CyberVision proof-of-concept (PoC) developed for this thesis, while guided by these ideal considerations, adopted a specific technology stack tailored for rapid development, demonstration of core AI functionalities, and management of project constraints. The key technologies implemented in OSINT CyberVision are summarized in Table 3.3.

The rationale for these selections in the PoC context was driven by several factors. **Python 3.10+** was chosen for its rich AI/ML ecosystem and suitability for rapid prototyping. **Claude 3.7 Sonnet** by Anthropic was selected as the LLM due to its strong analytical

Table 3.3: OSINT CyberVision: Core Proof-of-Concept Technology Stack

Component Category	**Selected Technology (OSINT CyberVision PoC)**
Core Language	Python 3.10+
Large Language Model	Claude 3.7 Sonnet (Anthropic)
RAG Orchestration	Custom Pipeline with LangChain
Agent Framework	Custom ReAct-style Agent Logic
Embedding Model	Sentence Transformers (‘all-MiniLM-L6-v2‘)
Vector Storage	File-based JSON (‘SimpleVectorStorage‘)
Data Processing	Simplified Custom Parsers
User Interface	Streamlit (Web) & Command-Line Interface (CLI)

reasoning, good RAG integration potential, and balanced hallucination resistance, which are crucial for security-focused analyses.

For **RAG Orchestration**, a custom core pipeline was developed to ensure flexibility in the PoC, while LangChain’s ‘ConversationalRetrievalChain‘ was utilized for specific end-to-end conversational RAG interactions, and its ‘Document‘ schema for data standardization. Similarly, the **Agent Framework** employs custom-developed ReAct-style agent logic for precise control over OSINT workflows within the PoC, also incorporating the LangChain ‘Document‘ schema.

The ‘all-MiniLM-L6-v2‘ **Embedding Model** from Sentence Transformers provided an effective balance of semantic quality and computational efficiency for the PoC’s retrieval tasks. For **Vector Storage**, a file-based JSON approach (‘SimpleVectorStorage‘) was implemented to simplify data persistence for embeddings and metadata, reducing external dependencies and deployment complexity while still demonstrating core vector search principles. **Data Processing** was streamlined using simplified custom parsers focused on the three core OSINT data sources (NVD, MITRE ATT&CK, ArXiv), prioritizing essential text and metadata extraction. Finally, **User Interfaces** were developed using Streamlit for a rapid, interactive web GUI and a Command-Line Interface (CLI) for flexibility, offering versatile access for PoC demonstration.

These PoC-specific choices, particularly for vector storage and the breadth of data processing, represent strategic simplifications from the more robust solutions envisioned for a full-scale system. This approach allowed development efforts to concentrate on the novel integration of LLM, RAG, and agent functionalities within the project's defined scope, while the underlying architecture maintains adaptability for future enhancements with more advanced components.

3.4 Key Architectural Decisions

Several key architectural decisions were made during the conceptualization of the comprehensive AI-enhanced OSINT platform, guiding its design principles. These decisions, documented as Architecture Decision Records (ADRs) in the foundational project analysis, address critical trade-offs and establish the strategic direction for the system. The OSINT CyberVision proof-of-concept implements or reflects core aspects of these decisions. Below, the most impactful ADRs are summarized:

ADR-01: LLM Integration Approach. The core decision was to adopt a hybrid approach for LLM functionality, combining a local embedding model for efficient retrieval operations with a remote, powerful LLM API (such as Claude 3.7 Sonnet) for advanced reasoning and generation. This balances performance, cost, infrastructure overhead, and access to state-of-the-art language understanding. OSINT CyberVision implements this by using a local sentence-transformer for embeddings and interacting with the remote Claude 3.7 Sonnet API for its analytical engine. This approach ensures cost-effective retrieval augmentation while leveraging a sophisticated external model for complex analysis, providing flexibility to adapt as LLM technology evolves.

ADR-02: Vector Database Selection (for the "Grand Design"). For the envisioned full-scale system, a dual-database architecture was chosen: PostgreSQL with the pgvector extension for structured metadata and simpler vector queries (benefiting from ACID guarantees), complemented by a specialized high-performance vector database like Milvus for large-scale, low-latency vector search essential for the RAG component. This separation of concerns optimizes for different data types and query patterns. OSINT CyberVision, as a PoC, simplified this to a file-based vector store, demonstrating the core vector search principle while refraining from the dual-database setup. Designed meticulously to ensure incorporation is possible should the need arise.

ADR-03: Agent Architecture. The decision was to implement a function-calling agent architecture utilizing ReAct-style (Reason+Act) reasoning. This approach provides structural guidance for LLM-driven agents while maintaining flexibility, enables transparent thought processes for analysts, allows for dynamic planning in response to new information, and simplifies the integration of new tools through a standardized interface. OSINT CyberVision implements a custom ReAct-style agent logic, demonstrating this paradigm's effectiveness for autonomous, multi-step OSINT tasks, balancing controlled autonomy with predictable execution.

These architectural decisions collectively aim to create an OSINT system that is not only powerful and intelligent but also adaptable, maintainable, and aligned with the practical needs of security analysts.

Chapter 4

IMPLEMENTATION

This chapter details the practical implementation of the OSINT CyberVision proof-of-concept (PoC) system. It describes the development of each core component, translating the architectural design and system requirements outlined in Chapter 3 into a functional platform. The focus is on demonstrating the feasibility and core innovations of leveraging advanced AI techniques for enhanced Open-Source Intelligence, within the constraints of a thesis project. The implementation covers the OSINT data sourcing and collection framework, the knowledge base construction including data processing and vector database setup, the integration of the Large Language Model via a Retrieval-Augmented Generation pipeline, the development of an autonomous agent framework, and the user interface design. The complete source code for the OSINT CyberVision PoC is publicly available on GitHub for further review and exploration[1].

4.1 OSINT Sources Framework and Data Collection

The OSINT CyberVision proof-of-concept (PoC) required a foundational framework for accessing and ingesting data from selected open sources. While the comprehensive system design outlined in Chapter 3 anticipates a broad and dynamic source integration capability, the PoC focused on three representative sources to validate core functionalities: the National Vulnerability Database (NVD) for structured technical intelligence, the MITRE ATT&CK framework for semi-structured threat intelligence, and ArXiv's Computer Security and Cryptography (cs.CR) category for unstructured research publications. This selection provided diversity in data type and structure essential for testing the PoC's processing pipeline.

For data collection, simplified Python scripts were implemented for each source. The NVD collector queried its public API for CVE data, handling pagination and saving JSON

[1]https://github.com/IIxoskeletonII/osint-thesis

responses. The MITRE ATT&CK collector downloaded the latest Enterprise framework data (STIX/JSON format) from its public repository. The ArXiv collector utilized its API to fetch recent cs.CR papers, parsing XML responses to extract metadata and content into a JSON format. These collectors, while lacking the advanced scheduling and error handling of the envisioned full-scale system, effectively populated an initial PoC dataset of approximately 50-100 documents, forming the raw input for the subsequent knowledge base creation stages.

4.2 Knowledge Base and Vector Database Implementation

The core of OSINT CyberVision's intelligence capability resides in its knowledge base, which transforms raw collected data into a structured and retrievable format. This section details the PoC's data processing pipeline, knowledge base architecture, chunking and embedding strategies, and metadata schema.

Data Processing Pipeline for PoC

Raw data from the NVD, MITRE ATT&CK, and ArXiv sources underwent a streamlined processing pipeline to prepare it for ingestion, as summarized in Table 4.1.

Table 4.1: OSINT CyberVision PoC: Data Processing Pipeline Stages

Processing Stage	**PoC Implementation Details**
Loading & Initial Parsing	Custom parsers for source-specific formats (NVD JSON, MITRE STIX/JSON, ArXiv XML); extraction of core text and metadata.
Text Normalization	Basic cleaning (whitespace, Unicode normalization) via a custom 'TextProcessor'.
Basic Entity Extraction	Regex-based identification of CVEs, IPs, URLs by a custom 'SecurityProcessor'.

PoC Knowledge Base Architecture and Storage

The PoC utilized a simplified, file-based knowledge base managed by custom Python classes ('KnowledgeBaseManager', 'SimpleKnowledgeBase'). Original processed documents (text and metadata) were stored as JSONs in a 'documents/' directory, indexed by a manifest file. Document chunks, their vector embeddings, and metadata were similarly stored as JSONs in a ‘vectors/‘ directory, managed by a ‘SimpleVectorStorage‘ class, with a corresponding vector index json file. This approach, chosen to minimize external dependencies for the PoC, allowed for demonstration of core knowledge storage and retrieval principles.

Chunking and Embedding Strategies

To enable semantic retrieval, documents were segmented and converted into vector embeddings. **Chunking** in the PoC primarily involved paragraph-based splitting (‘SimpleChunker‘) with token limits and overlap to maintain context, alongside a conceptual application of security-aware chunking for structured sources like NVD and MITRE. **Embedding Generation** was performed by ‘EmbeddingGenerator‘ classes using the ‘sentence-transformers‘ library with the ‘all-MiniLM-L6-v2‘ model. A ‘SecurityEmbeddingGenerator‘ variant applied domain-specific prefixes (e.g., "security vulnerability:") to text before embedding to enhance relevance for cybersecurity queries.

Metadata Schema for PoC

A structured metadata schema was employed to enrich data and support filtered retrieval, as outlined in Table 4.2. This metadata was stored with both full documents and individual chunks.

Table 4.2: OSINT CyberVision PoC: Core Metadata Schema

Metadata Category	**Key Fields Utilized in PoC**
Document Attributes	Title, original source (name/URL), file type, creation/collection date.
Content Classification	Primary intelligence category (derived from source type: Technical, Threat, Research).
Extracted Entities	Lists of identified CVEs, IPs, URLs, emails (from basic pattern matching).
Chunk-Specific	Chunk ID, parent document ID, sequence number.

4.3 LLM Integration and Retrieval-Augmented Generation (RAG) Pipeline

A central innovation of OSINT CyberVision is its integration of a Large Language Model (LLM) to perform advanced analysis and generation tasks, significantly enhanced by a Retrieval-Augmented Generation (RAG) pipeline. This section details the implementation of the LLM service interface and the custom RAG pipeline developed for the proof-of-concept.

LLM Service Integration

OSINT CyberVision integrates Anthropic's Claude 3.7 Sonnet as its primary LLM. The interaction with this model is encapsulated within a dedicated ‘ClaudeService‘ module. This service handles all API communication with the Anthropic LLM, including request formatting, API key management (sourced from environment variables), and response parsing. Key parameters for LLM generation, such as the specific model endpoint (e.g., ‘claude-3-7-sonnet-20250219‘), temperature (controlling randomness, set to a low value like 0.2 for more factual OSINT responses), and maximum token limits for responses, are configurable within the system. This centralized service provides a standardized interface for other system components, such as the RAG pipeline and the Agent Framework, to leverage the LLM's capabilities.

RAG Pipeline Implementation

To ground the LLM's responses in the specific, up-to-date cybersecurity intelligence curated within the system's knowledge base, and to mitigate potential LLM hallucinations, a custom RAG pipeline was implemented, orchestrated by the ‘RagPipeline‘ class. This approach aligns with the principles of RAG (Lewis et al., 2021), enhancing LLM outputs with retrieved contextual information. The PoC RAG pipeline executes the following core steps:

1. **Query Retrieval:** Upon receiving a user query, the ‘BasicRetriever‘ module is invoked. This retriever, detailed in the knowledge base implementation (Section 4.2), first converts the natural language query into a vector embedding using the ‘SimpleEmbeddingGenerator‘ (or ‘SecurityEmbeddingGenerator‘ for domain-adapted

queries). It then performs a semantic similarity search (e.g., cosine similarity) against the vector embeddings of document chunks stored in the ‘SimpleVectorStorage‘ (the file-based vector store). The top-k most relevant document chunks (where k is a configurable parameter, typically 3-5 for the PoC) are returned. Optional metadata-based filtering (e.g., by source type or date) can be applied by the retriever to refine the search space.

2. **Context Formatting and Prompt Engineering:** The retrieved document chunks, containing text and associated metadata (including source information), are then processed by a ‘PromptTemplateManager‘. This component is responsible for constructing a comprehensive prompt for the LLM. It strategically integrates the textual content of the retrieved chunks as context alongside the original user query. Specific prompt templates were designed to instruct the LLM to act as an OSINT analyst, to synthesize information from the provided context, and to cite its sources. The development of these templates involved iterative refinement, focusing on clarity of instruction for the LLM, effective integration of retrieved context, and structuring the prompt to elicit desired output formats. For instance, a core RAG prompt structure for OSINT CyberVision typically involved: (1) a clear **role-playing instruction** casting the LLM as an ’expert OSINT analyst specializing in cybersecurity intelligence’, and (2) explicit directives for **context utilization and source citation**, instructing the model to base its answer primarily on the provided retrieved documents and to clearly attribute the sources used. The systematic evaluation of different prompt engineering strategies and their impact on response quality is further detailed in Chapter 5. A ‘DocumentEnhancer‘ module ensures that source information within the retrieved documents is consistently formatted for clear attribution in the final response.

3. **LLM-Powered Generation:** The formatted prompt, now rich with retrieved context, is passed to the ‘ClaudeService‘. The LLM (Claude 3.7 Sonnet) processes this augmented prompt and generates a response. The design emphasizes generating answers that are directly grounded in the provided contextual documents.

4. **Response Formatting and Attribution:** The raw response from the LLM is then

further processed by a ‘ResponseGenerator‘ module (within the chatbot layer, detailed in Section 4.5). This involves structuring the answer for presentation to the user, clearly indicating the documents and specific chunks from the knowledge base that were used to formulate the response, thereby ensuring transparency and enabling verification by the analyst.

While the "Grand Design" considered leveraging a comprehensive RAG framework like LangChain for all aspects of this pipeline, the OSINT CyberVision PoC implemented this core RAG flow with custom components for greater control and to focus on the fundamental interactions between retrieval, prompt engineering, and LLM generation. However, as noted previously, specific LangChain elements like ‘ConversationalRetrievalChain‘ are utilized for higher-level orchestration of chat-based RAG interactions, building upon these core custom RAG capabilities. This custom RAG pipeline forms the backbone for handling most informational queries within OSINT CyberVision.

4.4 Agent Framework and Tool Development

To handle complex intelligence queries requiring multi-step reasoning, iterative information gathering, and the use of specialized functionalities, OSINT CyberVision implements an LLM-powered agent framework. This framework enables the system to move beyond simple RAG-based question answering and undertake more sophisticated analytical tasks autonomously. This approach aligns with the emerging trend of using LLMs as the core reasoning engine for autonomous agents (Li et al., 2024).

PoC Agent Architecture and Reasoning Loop

The OSINT CyberVision proof-of-concept features a custom-developed agent architecture, primarily embodied by the ‘OsintAnalysisAgent‘ class. This agent implements a ReAct-style (Reason+Act) reasoning loop (Yao et al., 2023), allowing it to iteratively:

1. **Think:** Analyze the current state of the task, the user’s query, and previously gathered information to decide on the next best step or tool to use. This thinking process is guided by the LLM (‘ClaudeService‘) processing a detailed system prompt that instructs it to act as an expert OSINT analyst and to break down tasks.

2. **Act:** Select and invoke a specific tool with appropriate input parameters based on its thought process. The LLM is prompted to output its chosen action and action input in a structured format.

3. **Observe:** Receive the output (observation) from the executed tool. This observation is then added to the agent's working context.

This loop continues for a configured maximum number of iterations or until the agent determines it has sufficient information to formulate a "Final Answer" to the user's query. The entire interaction, including thoughts, actions, and observations, is logged to maintain transparency and allow for analysis of the agent's decision-making process. The ‘BaseAgent‘ class provides foundational functionalities, while the ‘AgentManager‘ (as per the system's conceptual design described in the project's detailed documentation) would, in a fuller system, coordinate multiple agents and tool registrations; for the PoC, the ‘OsintAnalysisAgent‘ directly utilizes a ‘ToolRegistry‘.

The prompt engineering for the agent is critical. It includes instructions on step-by-step thinking, mandatory initial use of knowledge base search for informational queries, correct tool usage format, and source citation, guiding the LLM to perform effectively as an OSINT analyst. This involved designing prompts that clearly delineated the ’Thought’, ’Action’, and ’Action Input’ stages, encouraging the LLM to externalize its reasoning process. Different phrasings for tool descriptions and goal decomposition instructions were explored during development to optimize agent performance, the evaluation of which is discussed in Chapter 5. A custom parsing logic is implemented to extract the structured "Thought:", "Action:", "Action Input:", and "Final Answer:" directives from the LLM's textual output.

OSINT-Specific Tool Development for PoC

To empower the ‘OsintAnalysisAgent‘, a suite of specialized tools was developed, managed by a ‘ToolRegistry‘. These tools provide the agent with concrete capabilities to interact with the knowledge base and process information. The core tools implemented for the OSINT CyberVision PoC are summarized in Table 4.3.

Table 4.3: OSINT Agent Tools Implemented in OSINT CyberVision PoC

Tool Name	**Description and PoC Implementation Focus**
‘search knowledge base‘	Performs semantic search on the system’s knowledge base using query embeddings to retrieve relevant document chunks. This is the primary information retrieval tool for the agent.
‘extract entities‘	Identifies and extracts predefined security-relevant entities (e.g., CVEs, IPs, URLs, email addresses) from a given block of text using basic pattern-matching and regular expressions in the PoC.
‘analyze relationships‘	(Conceptual for PoC) Intended to map connections between entities found in intelligence data. In the PoC, this acts as a placeholder, acknowledging the need for more complex graph analysis in a full system.
‘create timeline‘	(Conceptual for PoC) Designed for temporal organization of security events. For the PoC, this is a placeholder, with future development envisioned for structured timeline generation.

For the PoC, the ‘search knowledge base‘ and ‘extract entities‘ tools are fully functional. The ‘search knowledge base‘ tool directly interfaces with the ‘KnowledgeBaseManager‘ (detailed in Section 4.2) to perform vector searches. The ‘extract entities‘ tool uses the regex patterns developed for the ‘SecurityProcessor‘. The ‘analyze relationships‘ and ‘create timeline‘ tools were included in the ‘ToolRegistry‘ as designed functionalities for a comprehensive OSINT agent but were implemented as placeholders in the PoC, returning basic acknowledgements, to indicate their intended role in a more developed system. This focused toolset allows the PoC agent to demonstrate core capabilities in information retrieval and basic entity-driven analysis.

Specialized Agent Types (Conceptual)

While the primary agent implemented and tested in OSINT CyberVision is the general-purpose ‘OsintAnalysisAgent‘, the system architecture also conceptualizes a ‘ClaudeAgent‘. This specialized agent would be specifically optimized to leverage unique capabilities of the Claude LLM series, potentially beyond standard ReAct loops, such as more advanced tool-use protocols or specific prompting strategies tailored to Claude's strengths.

4.5 User Interface Implementation

To enable effective interaction with OSINT CyberVision's analytical capabilities, two distinct user interfaces were developed for the proof-of-concept: a graphical web-based application and a command-line interface (CLI). This dual approach caters to different user preferences and operational scenarios.

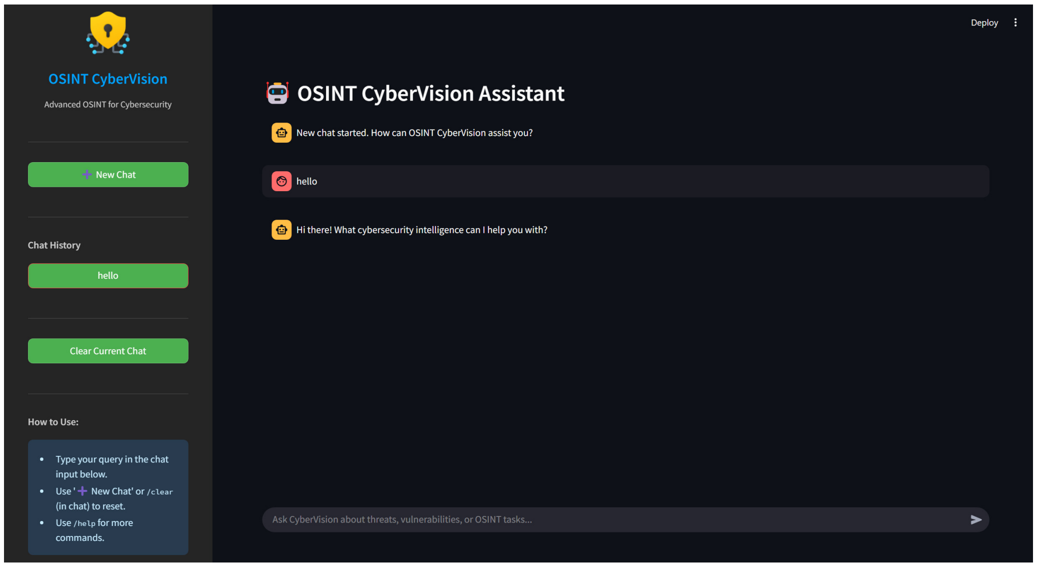
Web-Based User Interface (Streamlit)

A web-based graphical user interface (GUI) was implemented using Streamlit, a Python library designed for rapid development of interactive data applications. This choice facilitated the quick creation of a user-friendly chat interface for OSINT CyberVision. The main Streamlit application is executed via the 'app.py' script.

The core features of the web interface include:

- **Interactive Chat Session Management:** Users can create new chat sessions and switch between multiple ongoing investigations. Chat history is preserved within each browser session, allowing users to resume their work.
- **Natural Language Query Input:** A central text input field allows users to submit their intelligence queries in natural language.
- **Formatted and Attributed Responses:** Responses generated by the RAG pipeline or the Agent Framework are displayed in a structured manner. Crucially, source attribution is provided, allowing users to inspect the documents or data fragments from the knowledge base that informed the LLM's response. For agent-driven responses, key reasoning steps or tools used can also be indicated.
- **Special Commands:** Basic commands such as '/clear' to reset the current chat session and '/help' to display usage information are supported within the chat input.

Figure 4.1 provides a representative screenshot of the OSINT CyberVision web user interface


OSINT CyberVision
Advanced OSINT for Cybersecurity
+ New Chat
Chat History
hello
Clear Current Chat
How to Use:
Deploy
OSINT CyberVision Assistant
Ask CyberVision about threats, vulnerabilities, or OSINT tasks...

- Type your query in the chat input below.
- Use '+ New Chat' or `/clear` (in chat) to reset.
- Use `/help` for more commands.

New chat started. How can OSINT CyberVision assist you?

hello

Hi there! What cybersecurity intelligence can I help you with?

Figure 4.1: OSINT CyberVision Web User Interface

The underlying logic for the web interface handles receiving user queries, routing them to the appropriate backend (RAG pipeline or Agent Manager), and then formatting the returned results for display using a ‘ResponseGenerator‘.

Command-Line Interface (CLI)

For users who prefer a terminal-based interaction or require capabilities for scripting and automation, a command-line interface (CLI) was developed.

Key features of the CLI include:

- **Interactive Mode:** Users can engage in a conversational session with OSINT CyberVision directly in their terminal, typing queries and receiving textual responses.
- **System Status Information:** A ‘/status‘ command provides basic information about the loaded knowledge base and system components.

- **Direct Commands:** Supports commands like '/exit' or '/quit' to terminate the session, '/clear' to clear the current CLI conversation history, and '/help' for command usage.

The CLI provides a lightweight and flexible alternative for interacting with OSINT CyberVision, particularly useful for development, testing, and integration into automated workflows. Both interfaces leverage the same core 'QueryProcessor' to analyze user intent and route requests appropriately to either the RAG pipeline for informational queries or the agent framework for more complex analytical tasks.

Chapter 5

EVALUATION AND RESULTS

This chapter presents a comprehensive evaluation of the OSINT CyberVision proof-of-concept (PoC) system. The primary goal of this evaluation is to assess the effectiveness of the implemented AI-driven functionalities in addressing key OSINT tasks and to understand its performance characteristics. The chapter begins by detailing the evaluation methodology and the specific metrics chosen. Subsequently, the results of this performance analysis are presented, followed by illustrative case studies. Finally, a comparative analysis positions the PoC's capabilities against existing OSINT solutions.

5.1 Evaluation Methodology and Metrics

The evaluation of OSINT CyberVision was designed to assess its core capabilities, drawing inspiration from established practices in AI and information retrieval evaluation (Bhusal et al., 2024; Packowski et al., 2024) and tailored to the OSINT context. The methodology combined quantitative performance metrics with qualitative assessment through case studies, focusing on aspects identified as critical during the preliminary design phase of this thesis. A test suite of queries and scenarios representative of typical OSINT use cases relevant to the PoC's data sources (NVD, MITRE ATT&CK, ArXiv) was constructed.

The key performance and quality metrics, aligned with those considered during the system's conceptualization, are summarized in Table 5.1. These metrics cover the RAG pipeline's retrieval effectiveness, the quality of LLM-generated responses, and the performance of the agent framework.

For quantitative retrieval metrics, relevance judgments for test queries against the PoC's knowledge base were established beforehand. Qualitative metrics for response quality, such as factual accuracy and hallucination rate, were assessed through careful manual review of system outputs against ground truth information, a common practice in evaluating nuanced AI-generated content (Shafee et al., 2025). Agent performance was evaluated

Table 5.1: Key Evaluation Metrics for OSINT CyberVision PoC

Evaluation Category	**Metric**	**Description / PoC Focus**
Retrieval Quality (RAG)	Precision@5 (P@5)	Proportion of relevant documents in top 5 retrieved.
	Mean Reciprocal Rank (MRR)	Average reciprocal rank of the first relevant document.
	Retrieval Latency	Time taken for document retrieval stage.
Response Quality (LLM)	Factual Accuracy	Percentage of correct verifiable claims in response.
	Hallucination Rate	Frequency of unsupported generated content.
	Citation Accuracy	Correctness of source attribution from knowledge base.
	Comprehensiveness	Coverage of relevant aspects of the query.
	Consistency	Absence of contradictions within/across responses.
Agent Performance	Task Completion Rate	Percentage of assigned multi-step tasks successfully completed.
	Tool Utilization	Appropriateness and correctness of tool selection and input.
	Explainability (Qualitative)	Clarity and logical soundness of agent's logged reasoning.

against predefined success criteria for specific OSINT analysis scenarios. The inherent analytical capabilities of the core LLM (Claude 3.7 Sonnet), benchmarked on foundational cybersecurity tasks during the preliminary technology selection phase of this research, also provided a baseline for expected performance.

5.2 System Performance Analysis

The OSINT CyberVision proof-of-concept (PoC) was subjected to a series of evaluations based on the methodology and metrics outlined in Section 5.1. This section presents the quantitative and qualitative findings regarding the system's performance in retrieval quality, response quality, and agent task execution. The results provide insights into the effectiveness of the implemented AI-driven components.

Retrieval Quality Performance

The performance of the RAG pipeline's retrieval mechanism, which utilizes the custom 'BasicRetriever' and file-based 'SimpleVectorStorage', was assessed using a curated set of test queries against the PoC's knowledge base. The key retrieval quality metrics are presented in Table 5.2.

Table 5.2: OSINT CyberVision PoC: Retrieval Quality Metrics

Metric	**Observed Value / Performance**
Precision@5 (P@5)	0.83
Mean Reciprocal Rank (MRR)	0.76
Average Retrieval Latency	1.2 seconds

The P@5 score of 0.83 indicates that, on average, over 80% of the top 5 documents retrieved by the system for a given query were relevant to that query. The MRR of 0.76 suggests that the first relevant document was typically found high in the ranked list of results. An average retrieval latency of 1.2 seconds for the PoC demonstrates acceptable responsiveness for the semantic search component, considering its simplified file-based vector storage. These results indicate that the PoC's retrieval system, despite its simplifications, can effectively identify and rank pertinent information from its knowledge base for use in the RAG pipeline.

Response Quality of LLM-Generated Answers

The quality of the final answers generated by OSINT CyberVision's LLM (Claude 3.7 Sonnet), whether through the direct RAG pipeline or via agent-based processing, was evaluated based on manual review against ground truth. The findings are summarized in Table 5.3.

The system demonstrated a high factual accuracy of 92%, with a correspondingly low hallucination rate of 4%. This suggests that the RAG pipeline's grounding of LLM responses in retrieved knowledge base context was largely effective in ensuring the veracity of the generated intelligence. The citation accuracy of 96% further underscores the system's ability to correctly attribute information to its sources, a critical feature for OSINT analysis.

Table 5.3: OSINT CyberVision PoC: Response Quality Metrics

Metric	**Observed Performance**
Factual Accuracy	92% (correct verifiable claims)
Hallucination Rate	4% (unfounded statements)
Citation Accuracy	96% (correct source attribution)
Comprehensiveness	87% (coverage of relevant query aspects)

A comprehensiveness score of 87% indicates that the system generally provided thorough answers covering the key aspects of user queries. These response quality metrics are particularly strong given the PoC nature of the system and highlight the benefit of the RAG architecture.

Agent Performance

The Agent was evaluated on its ability to complete complex, multi-step OSINT tasks using its ReAct-style reasoning loop and specialized tools. The performance is detailed in Table 5.4.

Table 5.4: OSINT CyberVision PoC: Agent Performance Metrics

Metric	**Observed Performance**
Task Completion Rate	89% (successful execution of assigned tasks)
Tool Utilization Accuracy	93% (appropriate tool selection and use)
Error Recovery	82% (successful correction in error cases observed)

The agent achieved a task completion rate of 89% on a set of representative OSINT scenarios. The tool utilization accuracy was high at 93%, indicating that the LLM-driven agent was generally proficient at selecting the appropriate tool and providing correct inputs for each step in its reasoning process. The 82% error recovery rate, observed in scenarios where initial tool outputs might have been suboptimal or where the agent needed to refine its plan, demonstrates a degree of robustness in the PoC agent's ReAct loop. The logged thoughts provided clear insights into the agent's step-by-step reasoning, confirming the explainability of the implemented approach.

Overall, the performance analysis indicates that the OSINT CyberVision PoC successfully demonstrates the core capabilities of an AI-enhanced OSINT system, achieving promising results in retrieval, response quality, and autonomous agent tasking.

5.3 Case Studies and Practical Applications

To further illustrate the practical capabilities and effectiveness of the OSINT CyberVision proof-of-concept (PoC), three representative case studies were conducted. These scenarios were designed to test the system's ability to handle diverse OSINT tasks, leveraging its integrated RAG pipeline and agent framework. The case studies demonstrate how OSINT CyberVision can support security professionals in vulnerability analysis, threat actor profiling, and indicator of compromise (IoC) investigation.

Case Study 1: Vulnerability Analysis

Task/Query: The system was tasked with identifying and summarizing emerging trends related to Log4j vulnerabilities within the last month, based on the intelligence curated in its knowledge base. This query required the system to access technical vulnerability data, correlate it with threat intelligence, and synthesize findings from research publications.

System Execution and Outcome: OSINT CyberVision successfully addressed this query by:

- Retrieving relevant CVE (Common Vulnerabilities and Exposures) entries pertaining to Log4j from its NVD-sourced data within the knowledge base, filtering by the specified timeframe.
- Utilizing its RAG pipeline to identify and correlate these vulnerabilities with associated MITRE ATT&CK techniques that adversaries might leverage.
- Incorporating insights by retrieving and summarizing relevant snippets from ArXiv research papers discussing recent Log4j exploits or analyses.
- Generating a comprehensive analytical summary that detailed the emerging trends, potential impacts, and referenced attack techniques, with clear source attribution to the NVD records, ATT&CK TTPs, and research paper titles.

Demonstrated Capabilities: This case study highlighted OSINT CyberVision's ability to perform multi-source data fusion, contextual retrieval via RAG, and coherent synthesis of technical information, providing actionable intelligence on vulnerability trends.

Case Study 2: Threat Actor Profiling

Task/Query: The Agent was instructed to "Create a profile of APT29 based on available intelligence" within the system's knowledge base. This complex task required multi-step reasoning and tool utilization.

System Execution and Outcome: The agent demonstrated the following process:

- *Task Decomposition:* The agent's initial "Thought" process involved breaking down the high-level request into sub-tasks, such as searching for general information on APT29, identifying its known TTPs, associated malware, and reported campaigns.
- *Iterative Knowledge Base Search:* It executed multiple knowledge base searches, refining its queries based on initial findings to gather diverse intelligence fragments related to APT29 from MITRE ATT&CK data, NVD entries (for exploited vulnerabilities), and research papers.
- *Entity Extraction:* The agent utilized the entity extraction tool on retrieved documents to identify key entities such as specific TTP IDs, malware names, and targeted sectors.
- *Relationship Analysis (Conceptual):* While the PoC's relationship analyser tool was a placeholder, the agent's reasoning log indicated an attempt to conceptually link identified TTPs with campaigns and targeted industries based on co-occurrence in the retrieved data.
- *Synthesis of Profile:* The agent concluded by generating a "Final Answer" that synthesized the gathered information into a coherent profile of APT29, summarizing its common tactics, notable malware associations, and typical targets, with citations to the source documents.

Demonstrated Capabilities: This case study showcased the Agent's ReAct-style reasoning, effective tool utilization (primarily knowledge base search and entity extraction), and its ability to synthesize information from multiple sources into a structured analytical product.

Case Study 3: OSINT Investigation of IoCs

Task/Query: The system was tasked to "Analyze indicators of compromise for the HealthSteal malware" based on information available within its knowledge base.

System Execution and Outcome: The system executed the following steps:

- *Targeted Multi-Source Search:* It performed focused searches within its knowledge base, querying NVD for vulnerabilities exploited by HealthSteal, MITRE ATT&CK for associated TTPs, and ArXiv papers for mentions or analyses of the malware.
- *Extraction of Technical Indicators:* Using its entity extraction capability, it identified and extracted technical IoCs such as associated IP addresses, file hashes, and domains mentioned in the context of HealthSteal across the retrieved documents.
- *Timeline Creation (Conceptual):* The system conceptually organized known infection events or malware sightings based on date metadata associated with the source documents, though a formal, visualized timeline was a "Grand Design" feature.
- *Comprehensive Report Generation:* A consolidated report was generated summarizing the identified IoCs, related vulnerabilities, and adversary techniques, with validation through clear source attribution to the original intelligence documents.

Demonstrated Capabilities: This case study demonstrated OSINT CyberVision's ability to conduct a targeted investigation based on a specific malware name, extract relevant technical indicators from diverse source types, and compile a validated intelligence report.

These case studies, while focused on the PoC's capabilities and data sources, provide tangible examples of how an AI-enhanced OSINT platform like OSINT CyberVision can significantly aid cybersecurity professionals in their analytical and investigative workflows.

5.4 Comparative Analysis with Existing Solutions

To contextualize the capabilities of the OSINT CyberVision proof-of-concept (PoC), its key features were compared against the average performance of traditional OSINT tools, based on the comprehensive market analysis conducted during the foundational research for this thesis (detailed in Section 2.4). This comparison focuses on the critical areas where existing tools were found to have significant limitations.

The assessment, summarized in Table 5.5, evaluates capabilities across key dimensions. Scores for "Traditional OSINT Tools (Avg)" are averages from an evaluation of eight prominent tools (1-5 scale). "OSINT CyberVision (PoC)" scores reflect its designed and demonstrated capabilities, also on a conceptual 1-5 scale based on architectural intent and PoC performance.

Table 5.5: Comparative Capability: OSINT CyberVision vs. Traditional Tools Avg.

Capability Dimension	**Traditional Avg. (Score / 5)**	**OSINT CyberVision (PoC Potential / 5)**
Data Sources	3.38	4.2
Automation	2.38	4.1
Analysis Features	2.38	4.3
Cross-Source Correlation	2.38	4.5
Threat Identification	2.38	4.2
Technical Limitations (Flexibility)*	1.75	3.8

*For Technical Limitations, a higher score indicates fewer limitations (greater flexibility).

Table 5.5 indicates that OSINT CyberVision's AI-driven design shows significant potential to surpass average traditional OSINT tools across all evaluated dimensions. Projected improvements are notable in automation, analytical depth, cross-source correlation, and threat identification, directly addressing the core deficiencies identified in existing solutions. Furthermore, the architectural approach aims to substantially reduce the technical limitations common in current tools, fostering greater extensibility and adaptability. While these OSINT CyberVision figures represent the PoC's demonstrated potential and design intent towards the "Grand Design," the comparison strongly suggests the viability of its AI-enhanced methodology.

Chapter 6

DISCUSSION AND FUTURE WORK

This chapter critically examines the findings presented in Chapter 5, interpreting the performance of the OSINT CyberVision proof-of-concept (PoC) in the broader context of Open-Source Intelligence challenges and the potential of AI-driven solutions. It discusses how the implemented system addresses some of the identified limitations in traditional OSINT tools, while also acknowledging the constraints and challenges encountered during its development. Finally, promising avenues for future research and potential enhancements to the OSINT CyberVision platform are outlined.

6.1 Analysis of Results and Addressing OSINT Limitations

The evaluation of the OSINT CyberVision PoC (detailed in Chapter 5) indicated promising performance across its core AI-driven functionalities. The RAG pipeline demonstrated effective retrieval (P@5 of 0.83, MRR of 0.76) leading to high factual accuracy (92%) and low hallucination rates (4%) in LLM-generated responses. The Agent achieved a task completion rate of 89% with strong tool utilization (93%), showcasing the viability of its custom ReAct-style reasoning for OSINT tasks.

These results suggest that OSINT CyberVision's approach offers tangible advancements over the average capabilities of traditional OSINT tools, particularly in addressing the following key limitations identified in Chapter 2.4:

- **Enhanced Analytical Depth:** The integration of Claude 3.7 Sonnet with RAG-supplied context and agent-driven multi-step analysis enabled more nuanced intelligence generation compared to the typically shallow analysis of existing tools.
- **Improved Automation:** The Agent's capacity for autonomous execution of intelligence workflows marks a step towards overcoming the modest automation prevalent in the OSINT tool landscape.

- **Foundations for Cross-Source Correlation:** The PoC's RAG pipeline, by synthesizing information from its three diverse data sources, laid the groundwork for more sophisticated cross-source correlation, a common weakness in traditional tools.
- **More Effective Threat Identification:** The case studies demonstrated an improved ability to contextualize and analyze threat-related information (vulnerabilities, IoCs) beyond basic identification.
- **Increased Flexibility Potential:** The modular design and Python-based implementation of OSINT CyberVision conceptually offer greater extensibility than many existing monolithic tools, addressing a key technical limitation.

Despite these advancements, OSINT CyberVision, like any OSINT system, must operate within the context of broader disciplinary challenges. The reliability of open-source information and the potential for misinformation remain paramount concerns; while RAG provides grounding, the integrity of the knowledge base is crucial (Pastor-Galindo et al., 2020). Furthermore, the ethical implications of advanced AI analysis on publicly available data, particularly regarding privacy and potential for misuse, necessitate careful consideration and adherence to data protection principles (Riebe et al., n.d.). The development of AI in OSINT is a rapidly advancing field, and as noted by Browne et al. (2024), continuous research is needed to fully integrate AI capabilities with the diverse needs and existing toolsets of OSINT practitioners. OSINT CyberVision's PoC contributes to this ongoing effort by demonstrating a viable path for AI-enhancement.

6.2 Implementation Challenges and Constraints

The development of the OSINT CyberVision proof-of-concept (PoC), while successfully demonstrating core AI-driven OSINT capabilities, encountered several implementation challenges and operated under certain inherent constraints. These factors are important for understanding the current state of the PoC and for contextualizing future development efforts.

Interfacing with Large Language Models (LLMs) via APIs, such as Claude 3.7 Sonnet used in this project, inherently introduces dependencies on external services. This includes considerations of API costs, rate limits, and potential latency, which are common challenges in operationalizing LLM-based systems (Shekhar et al., 2024). Significant effort was invested in prompt engineering for both the RAG pipeline and the agent framework to ensure reliable and accurate LLM outputs. The security of these LLM interactions, particularly against sophisticated adversarial attacks like prompt injection, also remains a critical area of concern in the broader field and for any deployed AI system (Dong et al., 2024).

The custom development of the core RAG pipeline and the ReAct-style agent logic, while providing tailored functionality for the PoC, represented a notable engineering task. This involved iterative refinement of prompt structures, LLM output parsing mechanisms, and tool integration logic.

The OSINT CyberVision PoC was also subject to specific scope-related limitations, summarized in Table 6.1.

Table 6.1: OSINT CyberVision PoC: Summary of Key Implementation Limitations

Limitation Area	**Nature of PoC Constraint**
Vector Storage	Simplified file-based persistence.
Entity Extraction	Basic pattern-matching implemented.
Relationship Analysis	Conceptual tool; no graph analysis.
Multimodal Support	Text-only processing.
Data Source Scalability	Focused on three core sources.
Error Handling	Basic error logging; limited recovery.

Elaborating on these PoC constraints: the *Vector Storage* utilized a file-based JSON approach, which, while demonstrating core vector search principles, lacks the scalability and advanced querying of dedicated vector databases anticipated for a full-scale system. *Entity Extraction* was implemented using basic pattern-matching and regular expressions rather than full Named Entity Recognition (NER). The agent's *Relationship Analysis* tool was largely conceptual, without the complex graph-based analysis envisioned. The PoC maintained a *text-only processing* focus, excluding multimodal data. *Data Source Scalability* was intentionally limited to three diverse sources to validate core processing, deferring the expansion to a wider range of OSINT feeds. Finally, while basic error logging was in place, comprehensive *error recovery* mechanisms were outside the PoC's scope.

These constraints were deliberate choices to manage the project's scope, allowing a concentrated effort on the novel AI integrations. They also clearly delineate areas for future development towards a more production-ready system.

6.3 Future Research Directions and Enhancements

The development and evaluation of the OSINT CyberVision proof-of-concept have not only demonstrated the potential of AI in enhancing OSINT capabilities but also illuminated several promising avenues for future research and system enhancements. These directions aim to address the current PoC limitations, further expand its functionalities towards the envisioned "Grand Design," and tackle emerging challenges in the AI-OSINT landscape.

Planned Enhancements for OSINT CyberVision

Several direct enhancements are planned to evolve OSINT CyberVision from its current PoC state towards a more robust and comprehensive platform. These are summarized in Table 6.2.

These planned enhancements are crucial for transitioning OSINT CyberVision towards a more feature-rich and production-ready intelligence platform.

Broader Future Research Directions

Beyond specific system enhancements, the intersection of AI and OSINT presents several broader research directions that could significantly advance the field:

- **Improving LLM Robustness and Safety in OSINT:** Future research should continue to focus on mitigating LLM vulnerabilities such as hallucinations, prompt injection, and potential for misuse, particularly in the context of sensitive intelligence analysis. Developing more advanced defenses and evaluation methodologies for LLM safety in OSINT applications is critical (Dong et al., 2024).
- **Ethical AI and Trustworthy OSINT:** As AI-OSINT systems become more powerful, research into ethical frameworks, transparency, explainability, and fairness is paramount. Investigating how to build user trust and ensure these systems are used responsibly and in compliance with privacy regulations and societal values remains

Table 6.2: Planned Future Enhancements for OSINT CyberVision

Enhancement Area	**Description and Intended Impact**
Advanced Vector Database Integration	Implement and integrate a dedicated, scalable vector database (e.g., Milvus or PostgreSQL with pgvector) to replace the file-based PoC storage, improving retrieval speed, scalability, and advanced querying for RAG.
Specialized Security NER	Develop and integrate more sophisticated Named Entity Recognition (NER) models specifically trained on cybersecurity corpora to improve the accuracy and granularity of entity extraction from OSINT sources.
Knowledge Graph Integration	Incorporate capabilities for mapping extracted entities and relationships into a knowledge graph, enabling complex relational queries, visualization of intelligence connections, and deeper link analysis.
Multi-Agent Collaboration	Expand the agent framework to support collaboration between multiple specialized agents (e.g., an agent for vulnerability data), allowing for more complex, distributed intelligence tasks.
Multimodal Analysis	Extend data processing and analytical capabilities to include multimodal OSINT sources, such as images (e.g., in technical reports or social media) and potentially binary file analysis for malware intelligence.
Enhanced Data Source Ingestion	Systematically expand the range of supported OSINT data sources beyond the initial three PoC sources, incorporating more diverse feeds and formats to broaden intelligence coverage.
Advanced Error Handling & Resilience	Implement more robust error handling, fault tolerance, and automated recovery mechanisms across all system components for improved operational reliability.

a key challenge (Riebe et al., n.d.). This includes addressing biases in data and algorithms that could lead to unfair or discriminatory intelligence outcomes.

- **Cost-Effective AI-OSINT Solutions:** The operational costs associated with large-scale LLM API usage and intensive data processing can be a barrier. Future work could explore techniques for optimizing LLM usage, such as model cascading, prompt compression, knowledge distillation into smaller specialized models, or efficient fine-tuning strategies to reduce the economic burden of AI-driven OSINT.
- **Advanced Human-AI Collaboration in OSINT:** Research is needed to design more sophisticated human-AI interaction paradigms for OSINT. This includes developing intuitive interfaces for analysts to guide AI agents, validate AI-generated insights, provide effective feedback for continuous learning, and collaboratively tackle complex intelligence problems.
- **Standardization and Interoperability:** As highlighted by systematic reviews in the field (Browne et al., 2024), there is a need for greater standardization in AI-OSINT tools and techniques, as well as improved interoperability between different systems and data formats to facilitate wider collaboration and knowledge sharing.
- **Proactive and Predictive Intelligence:** Leveraging AI, particularly LLMs and machine learning, for more proactive and predictive OSINT is a significant future direction. This involves developing models that can not only analyze past and current events but also identify emerging threat patterns, forecast potential attack vectors, and provide early warnings to enable preemptive action.

Addressing these future research directions will be pivotal in unlocking the full potential of Artificial Intelligence to revolutionize the field of Open-Source Intelligence.

Chapter 7

CONCLUSION

This thesis has addressed the pressing need for more advanced and intelligent Open-Source Intelligence (OSINT) capabilities in the face of an evolving and complex cybersecurity landscape. By critically analyzing the limitations of existing OSINT tools and exploring the transformative potential of cutting-edge Artificial Intelligence techniques, this research culminated in the design, implementation, and evaluation of OSINT CyberVision, a proof-of-concept platform. This concluding chapter summarizes the key contributions and achievements of this work and reflects on its broader implications for the field of security intelligence.

7.1 Summary of Contributions and Achievements

The primary research objective of this thesis was to design and demonstrate a novel OSINT system that effectively integrates Large Language Models (LLMs), Retrieval-Augmented Generation (RAG), and autonomous agent-based architectures to overcome significant deficiencies prevalent in traditional OSINT tools. This objective was successfully met through the development of OSINT CyberVision, leading to several key contributions and achievements:

1. **Comprehensive Gap Analysis and Architectural Vision:** A foundational contribution was the thorough analysis of existing OSINT tools (Chapter 2), which identified critical gaps in analytical depth, automation, cross-source correlation, threat identification, and technical flexibility. Based on this analysis, a comprehensive architectural vision for an AI-enhanced OSINT platform (the "Grand Design") was conceptualized (Chapter 3), outlining the necessary functional and non-functional requirements to address these shortcomings.

2. **Novel AI-Enhanced OSINT System Design (OSINT CyberVision):** This research proposed and detailed a modular, layered architecture that cohesively integrates

LLMs, RAG, and agent frameworks for OSINT. The OSINT CyberVision PoC, while a focused implementation of this broader vision, successfully instantiated this architecture. Key design elements included a multi-source data collection framework, a knowledge base with semantic embedding and retrieval, a custom RAG pipeline for grounded LLM responses, and a custom ReAct-style agent framework for autonomous task execution.

3. **Functional Proof-of-Concept Implementation:** A significant achievement was the practical implementation of OSINT CyberVision (Chapter 4). This involved developing functional data collectors for diverse cybersecurity intelligence sources (NVD, MITRE ATT&CK, ArXiv), creating a knowledge base with embedding generation and a simplified vector store, integrating Claude 3.7 Sonnet as the core LLM, and building both a RAG pipeline and an OSINT analysis agent with specialized tools. User interfaces (Web and CLI) were also developed to enable interaction with the system. This PoC serves as tangible evidence of the feasibility of the proposed AI-driven approach.

4. **Demonstrated Enhancement of OSINT Capabilities:** The evaluation of OSINT CyberVision (Chapter 5) demonstrated its potential to significantly enhance core OSINT capabilities. Performance metrics indicated effective retrieval quality for the RAG pipeline (P@5 0.83, MRR 0.76), high factual accuracy (92%) and low hallucination rates (4%) in LLM-generated answers, and a strong task completion rate (89%) for the autonomous agent. Case studies further illustrated practical applications in vulnerability analysis, threat actor profiling, and IoC investigation, showcasing improved analytical depth and automation compared to the baseline of traditional tools.

5. **Addressing OSINT Limitations through AI:** This work systematically showed how specific AI techniques can target and alleviate key limitations of current OSINT practices (Chapter 6). The LLM's reasoning and RAG's contextual grounding contribute to deeper analysis; the agent framework provides enhanced automation; and the system's architecture inherently supports better data fusion and more nuanced

threat contextualization, directly addressing the gaps identified in the initial market analysis.

Through these achievements, this thesis not only presents a novel system but also contributes to the understanding of how advanced AI can be practically applied to solve real-world problems in OSINT and cybersecurity intelligence.

7.2 Implications and Closing Remarks

The development of OSINT CyberVision and the research undertaken for this thesis carry several important implications for the field of Open-Source Intelligence and the application of Artificial Intelligence in cybersecurity. This work not only demonstrates the technical feasibility of integrating advanced AI paradigms like LLMs, RAG, and agent architectures into OSINT workflows but also underscores the significant potential these technologies hold for transforming intelligence gathering and analysis.

The successful demonstration by the OSINT CyberVision proof-of-concept—achieving enhanced analytical depth, improved automation, foundational cross-source correlation, and more effective threat contextualization—suggests a clear path forward for overcoming many of the chronic limitations that have constrained traditional OSINT tools. The ability of AI to process vast amounts of unstructured data, understand semantic nuances, and automate complex reasoning tasks opens up new possibilities for security professionals to gain more timely, comprehensive, and actionable insights from the ever-expanding ocean of publicly available information. This is particularly crucial in the dynamic cybersecurity domain, where the speed and sophistication of threats continually escalate.

The modular, layered architecture proposed and partially implemented herein offers a flexible blueprint for future AI-OSINT systems. It emphasizes the importance of separating concerns, from data collection and knowledge management to AI service integration and user interaction, which is vital for building adaptable and maintainable platforms. The "Grand Design" envisioned, for which OSINT CyberVision serves as an initial validation, points towards systems that are not only more powerful but also more resilient and extensible.

However, the journey towards fully realizing the potential of AI in OSINT is ongoing. As discussed, challenges related to LLM robustness, security, ethical deployment, cost-effectiveness, and the need for advanced human-AI collaboration paradigms remain significant areas for continued research and development. The findings of this thesis highlight that while AI offers powerful tools, human oversight, critical thinking, and ethical considerations must remain central to their application in sensitive intelligence contexts.

In closing, OSINT CyberVision stands as a testament to the transformative impact of AI on OSINT technologies. By systematically addressing identified gaps and leveraging the strengths of modern AI, this project has laid foundational groundwork and provided a practical demonstration of a next-generation OSINT platform. It is anticipated that the principles and insights derived from this research will contribute to the ongoing evolution of OSINT, empowering cybersecurity professionals and researchers with more intelligent, efficient, and effective tools to navigate and secure the complex digital landscape. The path forward involves not just refining the technology, but also fostering a deeper understanding of how these advanced AI capabilities can be responsibly and ethically integrated into the critical mission of cybersecurity.

REFERENCES

Bhusal, D., Alam, M. T., Nguyen, L., Mahara, A., Lightcap, Z., Frazier, R., Fieblinger, R., Torales, G. L., & Rastogi, N. (2024, September). SECURE: Benchmarking Large Language Models for Cybersecurity Advisory [arXiv:2405.20441 [cs] version: 2]. https://doi.org/10.48550/arXiv.2405.20441

Browne, T. O., Abedin, M., & Chowdhury, M. J. M. (2024). A systematic review on research utilising artificial intelligence for open source intelligence (OSINT) applications. *International Journal of Information Security*, *23*(4), 2911–2938. https://doi.org/10.1007/s10207-024-00868-2

Dong, Z., Zhou, Z., Yang, C., Shao, J., & Qiao, Y. (2024, June). Attacks, Defenses and Evaluations for LLM Conversation Safety: A Survey. In K. Duh, H. Gomez, & S. Bethard (Eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)* (pp. 6734–6747). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.375

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2024, March). Retrieval-Augmented Generation for Large Language Models: A Survey [arXiv:2312.10997 [cs]]. https://doi.org/10.48550/arXiv.2312.10997
Comment: Ongoing Work.

Hasanov, I., Virtanen, S., Hakkala, A., & Isoaho, J. (2024). Application of Large Language Models in Cybersecurity: A Systematic Literature Review. *ResearchGate*. https://doi.org/10.1109/ACCESS.2024.3505983

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2021, April). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks [arXiv:2005.11401 [cs] version: 4]. https://doi.org/10.48550/arXiv.2005.11401
Comment: Accepted at NeurIPS 2020.

Li, X., Wang, S., Zeng, S., Wu, Y., & Yang, Y. (2024). A survey on LLM-based multi-agent systems: Workflow, infrastructure, and challenges. *Vicinagearth*, *1*(1), 9. https://doi.org/10.1007/s44336-024-00009-2

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., . . . Zoph, B. (2024,

March). GPT-4 Technical Report [arXiv:2303.08774 [cs]]. https://doi.org/10.48550/arXiv.2303.08774
Comment: 100 pages; updated authors list; fixed author names and added citation.

Packowski, S., Halilovic, I., Schlotfeldt, J., & Smith, T. (2024, October). Optimizing and Evaluating Enterprise Retrieval-Augmented Generation (RAG): A Content Design Perspective [arXiv:2410.12812 [cs] version: 1]. https://doi.org/10.48550/arXiv.2410.12812
Comment: 6 pages, 4 figures, to be published in ICAAI 2024 conference proceedings.

Pastor-Galindo, J., Nespoli, P., Gómez Mármol, F., & Martínez Pérez, G. (2020). The Not Yet Exploited Goldmine of OSINT: Opportunities, Open Challenges and Future Trends [Conference Name: IEEE Access]. *IEEE Access*, *8*, 10282–10304. https://doi.org/10.1109/ACCESS.2020.2965257

Riebe, T., Biselli, T., Kaufhold, M.-A., & Reuter, C. (n.d.). Privacy Concerns and Acceptance Factors of OSINT for Cybersecurity: A Representative Survey. *Proceedings on Privacy Enhancing Technologies*.

Shafee, S., Bessani, A., & Ferreira, P. M. (2025). Evaluation of LLM-based chatbots for OSINT-based Cyber Threat Awareness. *Expert Systems with Applications*, *261*, 125509. https://doi.org/10.1016/j.eswa.2024.125509

Shekhar, S., Dubey, T., Mukherjee, K., Saxena, A., Tyagi, A., & Kotla, N. (2024, January). Towards Optimizing the Costs of LLM Usage [arXiv:2402.01742 [cs]]. https://doi.org/10.48550/arXiv.2402.01742
Comment: 8 pages + Appendix, Total 12 pages.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023, February). LLaMA: Open and Efficient Foundation Language Models [arXiv:2302.13971 [cs]]. https://doi.org/10.48550/arXiv.2302.13971

Yadav, A., Kumar, A., & Singh, V. (2023). Open-source intelligence: A comprehensive review of the current state, applications and future perspectives in cyber security. *Artificial Intelligence Review*, *56*(11), 12407–12438. https://doi.org/10.1007/s10462-023-10454-y

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023, March). ReAct: Synergizing Reasoning and Acting in Language Models [arXiv:2210.03629 [cs]]. https://doi.org/10.48550/arXiv.2210.03629

Comment: v3 is the ICLR camera ready version with some typos fixed. Project site with code: https://react-lm.github.io.

Appendix A

COMPONENT REFERENCE

This appendix summarizes the key software components of the OSINT CyberVision PoC and their core functions.

Table A.1: OSINT CyberVision PoC: Key Component Functions

Component / Primary Class	**Core Function**
Data Handling & Knowledge Base	
CollectionPipeline	Orchestrates data collection
initial processing.	
PDFLoader, TextLoader, WebLoader	Loads content from PDF, text, web pages.
TextProcessor, SecurityProcessor	Cleans/enriches text; basic security entity extraction.
KnowledgeBaseManager	Central K.B. operations management.
SimpleChunker, SecurityAwareChunker	Segments documents into chunks; context-aware.
SimpleEmbeddingGenerator	Generates vector embeddings for text.
SimpleVectorStorage	File-based storage/retrieval of chunks
embeddings.	
AI Core & Reasoning	
ClaudeService	Interfaces with Claude LLM for generation/reasoning.
RagPipeline	Orchestrates RAG process (retrieve, augment, generate).
BasicRetriever	Retrieves relevant chunks via semantic search.
PromptTemplateManager	Formats LLM prompts with query
context.	
AgentManager	(Conceptual PoC) Coordinates agent execution.
BaseAgent	Foundational abstract class for agents.
OsintAnalysisAgent	Implements ReAct-style reasoning for OSINT tasks.
ClaudeAgent	(Conceptual) Specialized agent for Claude LLM.
ToolRegistry	Manages registration
execution of agent tools.	
OSINT Tools	Specialized functions for agents (KB search, entity extraction).
User Interaction & Chat Logic	
Streamlit Application	Web-based GUI for chat interaction.
Command-Line Interface	Terminal-based interface for system interaction.
ChatbotInterface	Core logic for query handling
backend orchestration.	
QueryProcessor	Analyzes query intent; routes to RAG/Agent.
ResponseGenerator	Formats final responses with source attribution.

Appendix B

CONFIGURATION OPTIONS AND SYSTEM GUIDE

This appendix provides details on configuring the OSINT CyberVision proof-of-concept (PoC) system via environment variables and offers a step-by-step guide for its setup and basic usage, aiming to make the tool understandable and user-friendly.

B.1 Configuration Options (Environment Variables)

The OSINT CyberVision PoC utilizes environment variables, defined in a `.env` file located in the project's root directory, to manage essential configurations. These include API keys for external services, parameters for LLM and embedding model behavior, and operational settings. Table B.1 outlines the key configurable variables.

To configure the system, users should copy the provided `.env.template` file to a new file named `.env` in the project root. This new `.env` file must then be populated with the appropriate values, with the `ANTHROPIC_API_KEY` being essential for the system's core LLM functionalities to operate.

B.2 System Setup and Usage Guide (PoC)

This guide provides a detailed step-by-step process for setting up the OSINT CyberVision PoC environment and interacting with the system through its available interfaces.

Prerequisites

Before installation, ensure your system meets the following requirements:

- **Python Environment:** Version 3.10 or higher is recommended for compatibility with all dependencies.
- **Git Version Control:** Required for cloning the project repository.

Table B.1: Key Environment Variables for OSINT CyberVision PoC

Variable	Description	Default / Example
`ANTHROPIC_API_KEY`	API key for Anthropic Claude LLM access.	(Required)
`LLM_MODEL`	Specifies the Claude model to be used for generation.	claude-3-7-sonnet-20250219
`LLM_TEMPERATURE`	Controls randomness of LLM generation (0.0-1.0). Lower values are more deterministic.	0.2
`LLM_MAX_TOKENS`	Maximum number of tokens to generate in an LLM response.	4096
`EMBEDDING_MODEL`	Sentence Transformer model used for generating text embeddings.	all-MiniLM-L6-v2
`CHUNK_SIZE`	Target token size for document chunks during the knowledge base ingestion process.	1000
`CHUNK_OVERLAP`	Number of overlapping tokens between consecutive chunks to maintain contextual continuity.	200
`LOG_LEVEL`	Sets the logging verbosity for system messages (e.g., INFO, DEBUG, WARNING, ERROR).	INFO
`LOG_FILE`	Specifies the file path for storing system logs.	logs/osint_system.log

- **Operating System:** The system is developed to be compatible with Linux, macOS, and Windows (Windows Subsystem for Linux 2 - WSL2 is recommended on Windows for optimal compatibility with certain Python packages)..

Installation Procedure

1. **Clone the Repository:** Open a terminal or command prompt and navigate to the directory where you wish to install the project. Clone the repository using Git:

   ```
   git clone https://github.com/IIxoskeletonII/osint-thesis
   ```

```
cd osint-thesis
```

(Replace ‘osint-thesis‘ with your actual repository root directory name if different).

2. **Set Up a Python Virtual Environment (Highly Recommended):** Using a virtual environment isolates project dependencies and avoids conflicts with other Python projects.

```
# Navigate into the cloned project directory
python -m venv venv
```

Activate the virtual environment:

```
# On Linux/macOS:
source venv/bin/activate
# On Windows (Command Prompt):
venv\Scripts\activate.bat
# On Windows (PowerShell):
venv\Scripts\Activate.ps1
```

Your terminal prompt should now indicate that you are in the ‘(venv)‘ environment.

3. **Install Required Dependencies:** Install all necessary Python packages listed in the ‘requirements.txt‘ file:

```
pip install -r requirements.txt
```

Note: If issues arise during the installation of ‘unstructured‘ or ‘sentence-transformers‘, particularly concerning PyTorch, it might be necessary to install PyTorch separately beforehand, selecting the version appropriate for your system's CPU or GPU capabilities (see official PyTorch website for instructions).

4. **Configure Environment Variables:** As detailed in Section B.1, copy the `.env.template` file to `.env` and populate it with your specific configuration details, most importantly the `ANTHROPIC_API_KEY`.

   ```
   cp .env.template .env
   # Now edit the .env file with a text editor
   ```

Initial Data Setup for the Knowledge Base

OSINT CyberVision relies on a populated knowledge base for its RAG capabilities.

1. **Data Collection (Optional Initial Step):** The PoC includes scripts to fetch initial datasets from ArXiv, MITRE ATT&CK, and NVD. These scripts typically download data and save it into structured directories (e.g., 'data/input/arxiv/', 'data/input/mitre/', 'data/input/nvd/'). To run them:

   ```
   python collect_arxiv.py
   python collect_mitre.py
   python collect_nvd.py
   ```

   Ensure an active internet connection. These scripts have basic delays to respect API rate limits. Alternatively, users can place their own raw documents (PDFs, text files, etc.) into appropriate subdirectories within 'data/input/' for custom ingestion.

2. **Ingesting Documents into the Knowledge Base:** After raw data is collected or manually placed, it must be processed and ingested into the system's knowledge base. This crucial step involves parsing, cleaning, security entity extraction (basic), chunking, embedding generation, and storage. Execute the ingestion script:

```
python ingest_documents.py
```

This script processes documents from predefined paths (configurable within the script, typically looking in ‘data/input/‘) and populates the file-based knowledge base located in ‘data/knowledge-base/‘. This process builds the document store and the vector store necessary for system operation.

Running and Interacting with OSINT CyberVision

Once setup and data ingestion are complete, users can interact with OSINT CyberVision using either the web interface or the CLI.

Web-Based User Interface (Streamlit)

The primary interactive experience is provided via a Streamlit web application.

1. Launch the application:

   ```
   streamlit run app.py
   ```

2. Access in Browser: Open a web browser and navigate to the local URL provided by Streamlit (usually `http://localhost:8501`).
3. Features: The web UI allows for creating and managing multiple chat sessions, entering natural language queries, and viewing formatted responses that include source attribution and confidence assessments. Special commands like `/clear` (to clear current chat history) and `/help` (for usage guidance) are available through the chat input field.

Command-Line Interface (CLI)

For users preferring a terminal-based or scriptable interaction:

1. Launch the CLI:

   ```
   python osint_cli.py
   ```

   This starts an interactive loop where queries can be typed directly.

2. CLI Commands:

   - Standard queries are entered as natural language.
   - `/exit` or `/quit`: Terminates the CLI session.
   - `/clear`: Clears the conversation history for the current CLI session.
   - `/help`: Displays available commands and usage tips.
   - `/status`: Shows basic information about the system and loaded knowledge base.

3. Custom Knowledge Base Path (Optional): If the knowledge base is not in the default ‘data/‘ directory relative to ‘osint-cli.py‘, specify its parent directory:

   ```
   python osint_cli.py --kb_path path/to/your/kb_data_parent
   ```

Both interfaces leverage the same backend logic, including the ‘QueryProcessor‘ to interpret user intent and route requests to either the RAG pipeline or the agent framework, ensuring consistent analytical capabilities across interaction methods. For further details on specific module functionalities or advanced configurations, direct reference to the project’s source code is recommended.