University of Burgos, Valladolid, and León

Master’s degree in

Business Intelligence and Big Data in Cyber-Secure Environments

Data Extraction from TOR-Based Ransomware Websites and Development of a Statistical Analysis Platform

Presented by:

Javier Martín Gallego

Supervised by:

Ángel Manuel Guerrero Higueras

Juan Delfin Peláez Álvarez

Valladolid, July 2025

Javier Martín Gallego

Data Extraction from TOR-Based Ransomware Websites and Development of a Statistical Analysis Platform

Resumen

El incremento del impacto de ciberataques, particularmente de tipo ransomware, supone una amenaza cada vez mayor para personas, organizaciones e infraestructuras. La gestión de amenazas convencional a menudo pasa por alto la información que se difunde en plataformas de la darknet. Este trabajo presenta el diseño y desarrollo de un sistema de web scraping orientado a la darknet, con el objetivo de extraer el contenido relacionado con ciberataques desde páginas pertenecientes a grupos de ciberdelincuencia. El scraper se centra principalmente en servicios ocultos basados en la red TOR, analizando contenido textual de webs de filtraciones donde los ciberdelincuentes publican sus ataques.

Además de la recolección de datos, se ha desarrollado un portal estadístico para almacenar, consultar y visualizar la información recopilada. Este portal permite a analistas de ciberseguridad e investigadores explorar tendencias, frecuencias y correlaciones en ataques de ransomware mediante interfaces gráficas, facilitando así análisis fundamentados sobre la evolución del panorama de amenazas cibernéticas.

La metodología incluye el rastreo de la darknet a través de la red TOR, la extracción estructurada de datos, el diseño de bases de datos y el uso de una arquitectura de desarrollo full stack para realizar consultas y generar paneles analíticos. Entre los principales desafíos abordados se encuentran el manejo ético de los datos y la mitigación de obstáculos como los mecanismos anti-bot.

En términos generales, este trabajo busca cerrar la brecha entre la inteligencia de fuentes abiertas y las redes ocultas donde gran parte del panorama de amenazas cibernéticas está en constante evolución.

Palabras clave

Web Scraping, Darknet, Ransomware, Ciberataques, Análisis

ABSTRACT

The increasing impact of cyberattacks, particularly ransomware incidents, poses a growing threat to individuals, organizations, and national infrastructure. The conventional threat intelligence often overlooks the information spread across darknet platforms. This thesis presents the design and development of a darknet web scraper aimed at extracting cyberattack-related content from cybercriminal webpages. The scraper focuses primarily on TOR-based hidden services, parsing textual content leak sites where threat actors post their attacks.

In addition to data collection, a comprehensive statistics portal has been developed to store, query, and visualize the harvested information. The portal enables cybersecurity analysts and researchers to explore trends, frequencies, and correlations in ransomware attacks through graphical interfaces, facilitating data-driven assessments of the evolving cyber threat landscape.

The methodology includes darknet crawling through the TOR network, structured data extraction, database design, and the use of a full stack architecture to query and generate analytical dashboards. Key challenges addressed include ensuring ethical data handling and mitigating scraping obstacles such as anti-bot mechanisms.

Overall, this work seeks to bridge the gap between open-source intelligence and the hidden networks where much of the cyber threat landscape is actively evolving.

Keywords

Web Scraping, Darknet, Ransomware, Cyberattacks, Analytics

LIST OF CONTENTS

Abstract........II

List of contents........III

List of Figures........V

List of Tables........VI

List of abbreviations........VII

Report........1

I Introduction........3

II Project objectives........4

III Theoretical concepts........5

1 Web scraping........5

2 Darknet........5

3 TOR network........5

4 ETL........6

5 NoSQL Databases........6

6 Orchestrator........7

7 Containers........7

8 Continuous Integration / Continuous Delivery........7

IV Techniques and tools........9

1 GitHub........9

2 Selenium........9

3 TOR........9

4 ChromeDriver........10

5 Python........10

6 MongoDB........10

7 Java........10

8 Spring Boot........11

9 React........11

10 Bootstrap........11

11 Typescript........11

12 D3.js 11
13 Nginx 12
14 Docker 12
15 Docker Compose 12
V Relevant aspects of the project 13
1 Web scraper design 13
1.1 Web scraper service 15
1.2 TOR service 16
1.3 ETL service 17
1.4 Database service 18
1.5 Orchestrator service 19
2 Web app design 20
2.1 Backend service 21
2.2 Frontend service 23
3 Ethical aspects 35
VI Related works 37
VII Conclusions and future worklines 40
Appendixes 46
Appendix A: Technical documentation 48
Introduction 48
Directory structure 48
Compilation, installation, and execution 48
Appendix B: Execution results 50
Introduction 50
Execution 50
Results 50
Bibliography 67

LIST OF FIGURES

Figure 1. System context diagram [24]. 13
Figure 2. Web scraper container diagram [24]. 15
Figure 3. Web Application container diagram [24]. 21
Figure 4. Web application’s header. 24
Figure 5. Web application's footer. 24
Figure 6. Cyberattacks interface 25
Figure 7. Cyberattack details interface. 26
Figure 8. Cybercriminal groups interface. 27
Figure 9. Number of cyberattacks time chart. 28
Figure 10. Number of cyberattacks by region. 29
Figure 11. Number of cyberattacks by target. 30
Figure 12. Frequency of cyberattacks by sector. 31
Figure 13. Cybercriminal group statistics section. 32
Figure 14. Number of cyberattacks by date 33
Figure 15. Number of cyberattacks by group. 33
Figure 16. Global statistics layout. 34
Figure 17. Directory structure overview. 48
Figure 18. project's building phase. 50
Figure 19. project's running phase. 50
Figure 20. Docker running containers. 50
Figure 21. mongo_backup_dag execution. 51
Figure 22. scraper_etl_pipeline execution. 51
Figure 23. Cyberattacks API endpoint. 62
Figure 24. Cyberattack details API endpoint. 63
Figure 25. Cybercriminals API endpoint. 64
Figure 26. Cybercriminal details endpoint. 65
Figure 27. Statistics API endpoint. 66

LIST OF TABLES

Table 1. General scraped data structure. 16

Table 2. ETL transformations applied. 18

Table 3. Database general scraped schema. 19

Table 4. Database general extra schema. 19

Table 5. Spring Boot dependencies [29]. 22

LIST OF ABBREVIATIONS

A

API: Application Programming Interface

ACID: Atomicity, Consistency, Isolation and Durability

B

BSON: Binary JSON

BiDi: BiDirectional

C

CI: Continuous Integration

CD: Continuous Delivery

CSS: Cascading Style Sheets

CAP: Consistency, Availability, Partition tolerance

CAPTCHA: Completely Automated Public Turing test to tell Computers and Humans Apart

CSRF: Cross-Site Request Forgery

CERT: Computer Emergency Response Team

CSIRT: Computer Security Incident Response Team

CLI: Command-Line Interface

CTI: Cyber Threat Intelligence

D

D3: Data-Driven Documents

DAG: Directed Acyclic Graph

DLS: Data Leak Site

E

ETL: Extract, Transform and Load

G

GUI: Graphical User Interface

H

HTML: HyperText Markup Language

I

I2P: Invisible Internet Project

IDE: Integrated Development Environment

J

JSON: JavaScript Object Notation

JSX: JavaScript XML

JVM: Java Virtual Machine

JDK: Java Development Kit

L

LOPDGDD: Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales

N

NoSQL: Not only SQL

R

RSS: Really Simple Syndication

REST: REpresentational State Transfer

RGPD: Reglamento General de Protección de Datos

S

SQL: Structured Query Language

SPA: Single-Page Application

SVG: Scalable Vector Graphics

SOCKS: SOCKetS

T

TOR: The Onion Router

U

UI: User Interface

URL: Uniform Resource Locator

W

W3C: Worl Wide Web Consortium

X

XML: eXtensible Markup Language

Y

YAML: YAML Ain’t Markup Language

REPORT

I INTRODUCTION

Since the origins of the internet in 1969, the digital infrastructure has evolved into a global ecosystem that supports communication, commerce, governance and research across every sector of society. Individuals, industries, academic institutions and government agencies have become increasingly dependent on interconnected systems. This digital expansion, however, has introduced vulnerabilities, making cybersecurity an essential component of modern digital life.

As digital technology grows, cyberattacks have evolved in scale, frequency, and sophistication. These attacks exploit weaknesses in software, networks, and human behaviour, targeting entities ranging from individuals to national governments. Among the various forms of cybercrime, ransomware has become one of the most damaging threats in recent years [1].

Ransomware is a type of malware that encrypts victim's data and demands a payment in exchange for the decryption key. This form of attack disrupts business operations, threatens data confidentiality, and can even impact critical infrastructure such as hospitals and transportation systems. The consequences are severe: financial losses, reputational damage, legal liability, and in some cases, threats to public safety.

The significance of ransomware is further amplified using the darknet, where many threat actors operate with relative anonymity. This underground ecosystem facilitates the monetization of cybercrime via cryptocurrencies and complicates efforts by law enforcement to trace criminals [2]. In this network, criminal groups leverage forums and marketplaces to distribute ransomware kits, recruit affiliates through Ransomware-as-a-Service (RaaS) programs, and publish successful attacks to pressure victims [3].

Given the central role of the darknet in facilitating ransomware operations and broader cybercriminal activity, it has become an important area of study for cybersecurity research. This project proposes the development of a web scraper specifically designed to monitor and extract information about cyberattacks from known cybercriminal darknet pages. By automating the collection of this data, the tool aims to provide a centralized statistics portal that offers real-time insights. This approach also seeks to bridge the gap between open-source intelligence and the hidden networks where much of the cyber threat landscape is actively evolving.

Despite the growing prevalence of ransomware, there is currently no accessible digital resource that aggregates and analyses cyberattacks affecting the Spanish territory. While international platforms exist, they often fail to provide detailed, localized insights to the context of Spain. This project addresses this void and contributes to build a stronger and more informed cybersecurity posture in the Spanish digital ecosystem.

II PROJECT OBJECTIVES

The initial foundation of the use case was guided by the following objectives:

- Build a web scraper capable of extracting relevant information from onion websites operated by ransomware groups.
- Implement anonymization and security techniques to ensure safe and private access to the darknet.
- Design and implement a scalable database to store collected data efficiently.
- Analyse key metrics related to ransomware activity.
- Build and deploy a web portal with interactive data visualization and statistical insights.

Following an in-depth research and design phase, the project was further refined into the following specific technical objectives:

- Automate data collection using Selenium integrated with the TOR network.
- Perform a complete ETL process to normalize the collected data using Python.
- Store the transformed data in a high-performance and scalable database such as MongoDB.
- Implement data querying and visualization through a robust full-stack architecture using Spring Boot, React and Typescript.
- Orchestrate and schedule the entire data pipeline using Apache Airflow.
- Containerize the project using Docker and Docker Compose to ensure smooth deployment across heterogeneous environments.

III THEORETICAL CONCEPTS

The theoretical concepts in which the project has been based are detailed in this section. All concepts are described in a detailed and simple way since this master's thesis can be aimed at technical and non-technical students.

1 Web scraping

Web scraping refers to a computer software technique used to automatically extract information from websites using software tools or scripts. Scrapers typically navigate through web pages, identify specific HTML elements and extract their content for storage or further analysis. While it is often associated with data analysis, marketing or academic research, it is also used in less transparent practices such as surveillance or competitive intelligence.

Legally, different jurisdictions approach it through varying combinations of copyright law, terms of service agreement, legality of the use of data, contracts and trade secrets [4].

Ethical considerations often impose stricter limits on web scraping practices. Although it is not enforceable by law, respecting a website's *robots.txt* file, avoiding harm to the data source and respecting individual privacy are good standards of ethical behaviour [4].

2 Darknet

The darknet is a hidden part of the internet that requires specific software, configurations, or authorization to access. Unlike the surface web, which is indexed by search engines like Google or Bing, the darknet operates within encrypted networks such as TOR, Freenet or I2P. These networks provide an extra layer of anonymization, privacy and security by routing traffic through multiple relays. While the darknet has legitimate uses such as anonymous trading, it is also a hotspot for illegal activities, including drug trafficking, weapon sales and cybercrime.

3 TOR network

The TOR network is based on the principle of onion routing, where data packets are encapsulated in multiple layers of encryption, enabling anonymous communication between a user and a destination. The path through the TOR network is composed of three primary relay types:

- Entry Guard Node: It is the first node in the circuit, which knows the user's IP but not the final destination.
- Middle Node: Forwards the encrypted data without knowing its origin or final endpoint.
- Exit Node: The last node decrypts the final layer and sends the traffic to its destination.

Each node in the path only knows its preceding and succeeding node, ensuring that no single relay can identify both the source and the destination. In this way, traffic obfuscation, location hiding and decentralization are achieved within the darknet.

4 ETL

ETL stands for Extract, Transform, Load and it represents a fundamental data pipeline process in any data project that ensures data quality facilitating further steps like visualization, statistical analysis and alerting systems. The ETL process involves the following three stages:

- The Extract phase consists of collecting raw data from various sources.
- The Transform phase processes the raw data into a structured and normalized format. It involves cleaning the data as well as converting data types to make the data meaningful.
- The Load phase the structured is inserted into the destination system, typically a data warehouse, where it can be efficiently queried for business intelligence and reporting.

This process can be done in batches or in real time, depending on the nature of the source of information and the user's needs.

5 NoSQL Databases

NoSQL databases differ from traditional relational databases in their ability to handle a wide variety of data types and structures. They are designed for scalability, flexibility and performance, especially for large-scale data storage and real-time web applications [5]. There are four main types of NoSQL databases, each optimized for different data models and access patterns:

- Document-Oriented databases

These databases store data in documents, typically using flexible formats like JSON, BSON or XML, providing a flexible schema suitable for hierarchical, semi-structured or rapidly changing data. *MongoDB* and *CouchDB* are examples of document-oriented databases.

- Key-Value stores.

Data is stored as a dictionary formed by a unique key and an associated value. These databases are simple and highly optimized for fast read and write operations. *Redis* and *Amazon DynamoDB* are examples of key-value stores.

- Column-Family stores.

Column-Family stores group related data in columns and rows, providing excellent write performance and high scalability for large-scale datasets. Examples of column-family stores would be *Apache Cassandra*, *HBase* or *ScyllaDB*.

- Graph databases.

Data is represented using nodes to represent entities and edges to define relationships. Therefore, these databases are efficient for highly connected data where the relationships are critical. *Neo4j* or *ArangoDB* belong to this type.

Distributed data stores such as NoSQL databases are related to the CAP theorem. This theorem states that only two out of the following three components can be guaranteed for a database at any given time [6]:

- Consistency. All reads fetch the latest version of the data.
- Availability. All reads contain data, even when some nodes are down.
- Partition tolerance. The system continues to operate despite network failures.

6 Orchestrator

An orchestrator in the context of data workflows is a tool that manages, schedules, and coordinates the execution of multiple interdependent tasks. For example, a scraping operation might need to be followed by data transformation, storage, and eventually visualization. Orchestrators ensure these processes occur in the correct sequence and can also retry tasks on failure, monitor progress, and log performance. This structured automation improves reliability, improves scalability and reduces the manual effort required to maintain data collection and processing over time making them widely used across domains like cloud infrastructure, machine learning and DevOps.

Orchestrators are commonly associated with containerization, where they manage multiple containers across clusters of serves. The most widely used orchestrators are *Kubernetes* and *Docker Swarm*.

7 Containers

Containers are a lightweight form of virtualization that package an application together with all its dependencies into a single isolated executable unit. This ensures that the software builds, ships and runs consistently across different environments such as a developer's laptop, a testing server, or a production cloud platform. This modular approach improves deployment speed, simplifies version control, and makes it easier to isolate and fix bugs.

This form of virtualization is widely used in microservices architecture, CI/CD pipelines, development or test environments and cloud deployments [7].

8 Continuous Integration / Continuous Delivery

Continuous Integration and Continuous Delivery are DevOps practices that improve the speed, quality, and reliability of software development. CI involves automatically testing and validating code whenever new changes are committed, ensuring bugs are caught early and code remains stable. CD takes this a step further by automating the deployment of tested code to production or staging environments. Together, CI/CD reduces the time between writing code and delivering it to users. Tools like *GitHub Actions* or *GitLab CI/CD* can be integrated to run tests, build containers, and deploy changes with minimal manual intervention, ensuring the system remains responsive to evolving threats and can scale efficiently.

IV TECHNIQUES AND TOOLS

The development tools used to carry out the project are presented in this section.

1 GitHub

Launched in 2008, GitHub is a web-based platform for version control and collaborative software development. It is built on top of Git, a distributed version control system that allows developers to track changes in code, collaborate on projects, and manage software versions efficiently. GitHub provides a cloud-based interface where developers can host public and private repositories, open issues, perform code reviews, and integrate CI/CD workflows [8].

2 Selenium

Selenium is an open-source framework used for automating web browsers. Primarily designed for automated testing, it is widely used in web scraping for interacting with dynamic websites that require user actions such as clicking, scrolling or entering data. It offers the following components [9].

- Selenium Webdriver. Selenium Webdriver is used to programmatically control major browsers like Google Chrome, Mozilla Firefox and Safari. It provides APIs in several programming languages such as Python, Java, JavaScript and C#.
- Selenium IDE. This is a browser extension for Google Chrome and Mozilla Firefox that allows users to record and playback interactions with the browser without programming.
- Selenium Grid. It allows users to execute tests on multiple machines and browsers in parallel. This is particularly useful for scaling up automated testing.

3 TOR

TOR is a decentralized network and software package that enables anonymous communication by routing internet traffic through a series of volunteer-operated servers, encrypting data at each step. This ensures high levels of privacy and anonymity for both users and servers. In cybersecurity research, TOR is essential for accessing darknet websites, many of which are not reachable through the regular internet. In addition, TOR provides the Tor Expert Bundle package to allow developers to bundle TOR with applications [10].

This network implements a layered encryption model. Each message is wrapped in multiple layers of encryption corresponding to the sequence of nodes it will pass through. As the message traverses the network, each relay removes one layer of encryption, revealing the address of the next node until it reaches its final destination. Therefore, no single node knows both the source and destination, preserving anonymity [11].

4 ChromeDriver

ChromeDriver is a standalone executable maintained by the Chromium project that enables programmatic control of the Google Chrome browser through the WebDriver protocol. It acts as a bridge between programmed test scripts and the actual browser, allowing automation scripts to simulate user interactions such as clicking, typing, scrolling, and navigation.

It is part of the WebDriver and the WebDriver BiDi standards of the W3C, which defines a language-neutral interface for controlling web browsers [12].

5 Python

First released in 1991 by Guido van Rossum, Python is a high-level, interpreted programming language known for its simplicity and versatility. Python is widely used in data science, web development, automation, and cybersecurity due to its rich ecosystem of libraries and frameworks. Its readability and strong community support make Python an ideal choice for rapid development and prototyping [13].

6 MongoDB

MongoDB is a widely used NoSQL created in 2007. It handles large volumes of unstructured or semi-structured data in BSON documents. This flexibility makes it well-suited for modern web applications, real-time analytics or content management. Its main features are the following [14]:

- Horizontal scalability splitting larger datasets across multiple distributed collections.
- High availability and stability through shard replication.
- Distributed and centralized multi-document ACID transactions.
- Indexing that support complex access patterns and can be created on demand.

7 Java

Java is a class-based object-oriented programming language and computing platform that enables developers to write software that runs on any device equipped with a JVM. Released by Sun Microsystems in 1995 and now maintained by Oracle, Java has become a fundamental technology in enterprise systems, mobile applications, web backends and scientific computers.

OpenJDK is the official open-source reference implementation of the Java Service Enterprise platform, led by the OpenJDK Community and governed by Oracle [15].

8 Spring Boot

Spring Boot is an open-source framework based on Java and the Spring framework, mainly used for building stand-alone, production-grade backend applications. It simplifies the configuration of the Spring framework and includes embedded servers like Tomcat, which eliminate the need for complex deployment setups. It ensures reliability, security, and scalability in handling data requests and interactions which makes it suitable for microservices, enterprise applications, APIs and middleware. Some of the key features of Spring Boot are listed below [16]:

- Automatically configure Spring and third-party libraries.
- Embed Tomcat, Jetty or Undertow directly.
- Provide production-ready features such as metrics, health checks and externalized configuration.

9 React

React is an open-source JavaScript library developed by Meta for building user interfaces for web and mobile applications. It uses a component-based architecture that allows developers to build reusable UI elements and efficiently manage application state through hooks. To boost performance, React only updates the elements that have changed between the DOM and a memory-stored representation of the DOM, called virtual DOM [17].

10 Bootstrap

Bootstrap is a popular open-source CSS framework originally developed by Twitter that provides a set of design templates and pre-built components to create responsive and mobile-first websites. It includes styles for buttons, forms, tables, modals, and more, ensuring a clean and consistent user experience without the need for custom CSS from scratch [18].

11 Typescript

TypeScript is an open-source programming language developed and maintained by Microsoft. It a superset of JavaScript that introduces static typing, type inference, interfaces and type aliases, which helps catch errors during development and improve code maintainability [19].

12 D3.js

Developed by Mike Bostock, D3.js is a powerful low-level JavaScript library for creating interactive and customizable data visualizations in web browsers using SVG, HTML, and CSS. It provides full control over how data is represented visually allowing the creation of interactive charts and graphs that help users explore data in a visual and intuitive manner [20].

13 Nginx

Nginx is an open-source, high-performance web server, reverse proxy and load balancer designed for speed, scalability and reliability. Originally developed by Igor Sysoev in 2002, Nginx performs optimally under heavy loads, making it widely adopted in both traditional and cloud-native environments [21].

14 Docker

Introduced in 2013, Docker is an open-source platform that enables developers to package applications and their dependencies into lightweight, portable and isolated environments known as containers. These containers run consistently across different environments, from development machines to production servers, especially in DevOps workflows and microservices architectures. The main concepts of Docker are the following [22]:

- The Docker Engine is the core component that runs and manage containers.
- Containers are standardized units of software that package code, libraries and dependencies into a single executable artifact. They run isolated from each other and the host system.
- Images are immutable blueprints for containers, created from a series of layers defined in a Dockerfile. They can be reused, shared and versioned.
- Volumes are persistent storage mechanisms that allow data to exist beyond the life of a container.
- Virtual networks allow containers to communicate.

15 Docker Compose

First released in 2014, Docker Compose is a tool that facilitates the definition and management of multi-container Docker applications. It provides a declarative way to describe multiple services, their networks, their volumes, etc. in a single YAML file, called docker-compose.yml [23].

V RELEVANT ASPECTS OF THE PROJECT

In alignment with the objectives outlined in this research, two core services have been developed. The first is a web scraper responsible for collecting data from the darknet within the TOR network. The second is a full-stack web application that provides a user-friendly interface for the analysis and visualization of the collected data. Both components are integrated through a shared database, which is periodically updated by the scraping service and queried by the web application.

Due to the limited information available in leak sites, it becomes imperative that a designated administrator will assume the responsibility of to providing more context for each registered cyberattack. This supplementary context is essential for the proper interpretation and analysis of the incidents.

To facilitate a clearer understanding of the operational environment, an abstract representation of the system architecture is illustrated in the figure presented below.

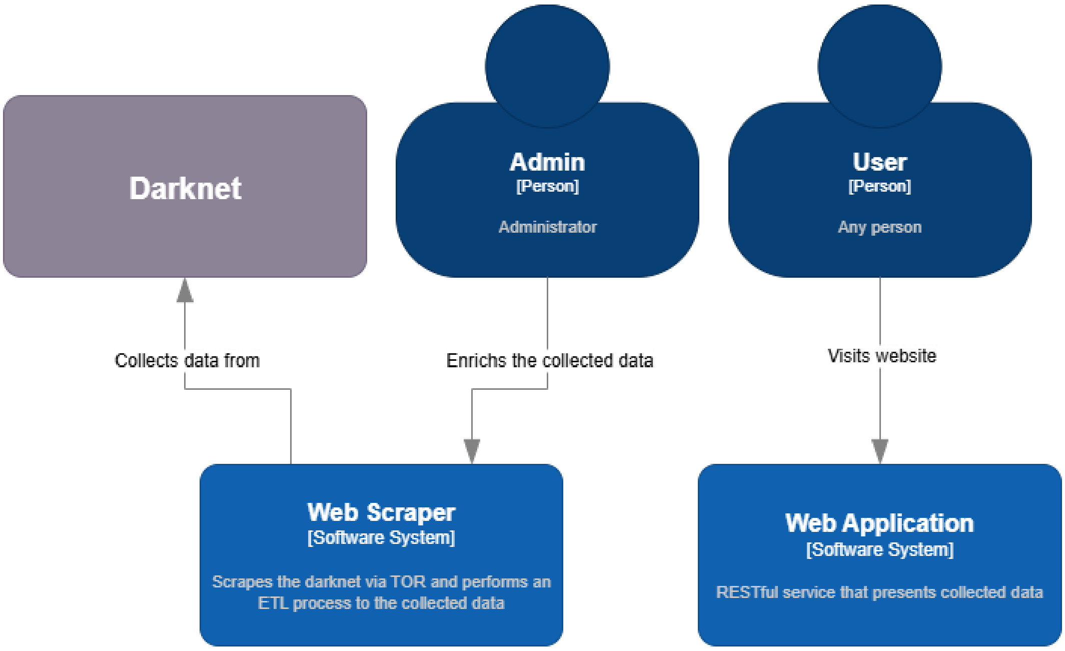

Darknet
Admin
[Person]
Administrator
User
[Person]
Any person
Collects data from
Enrichs the collected data
Visits website
Web Scraper
[Software System]
Scrapes the darknet via TOR and performs an ETL process to the collected data
Web Application
[Software System]
RESTful service that presents collected data

Figure 1. System context diagram [24].

Throughout the following sections, the design and the decision-making process of the proposed services are detailed.

1 Web scraper design

At the core of the subsystem are two primary components: the web scraper service and the TOR service. To enhance the system's overall functionality, the architecture further integrates two additional modules: an ETL process, a task orchestrator and a database backup service. The different services are detailed in the following sections.

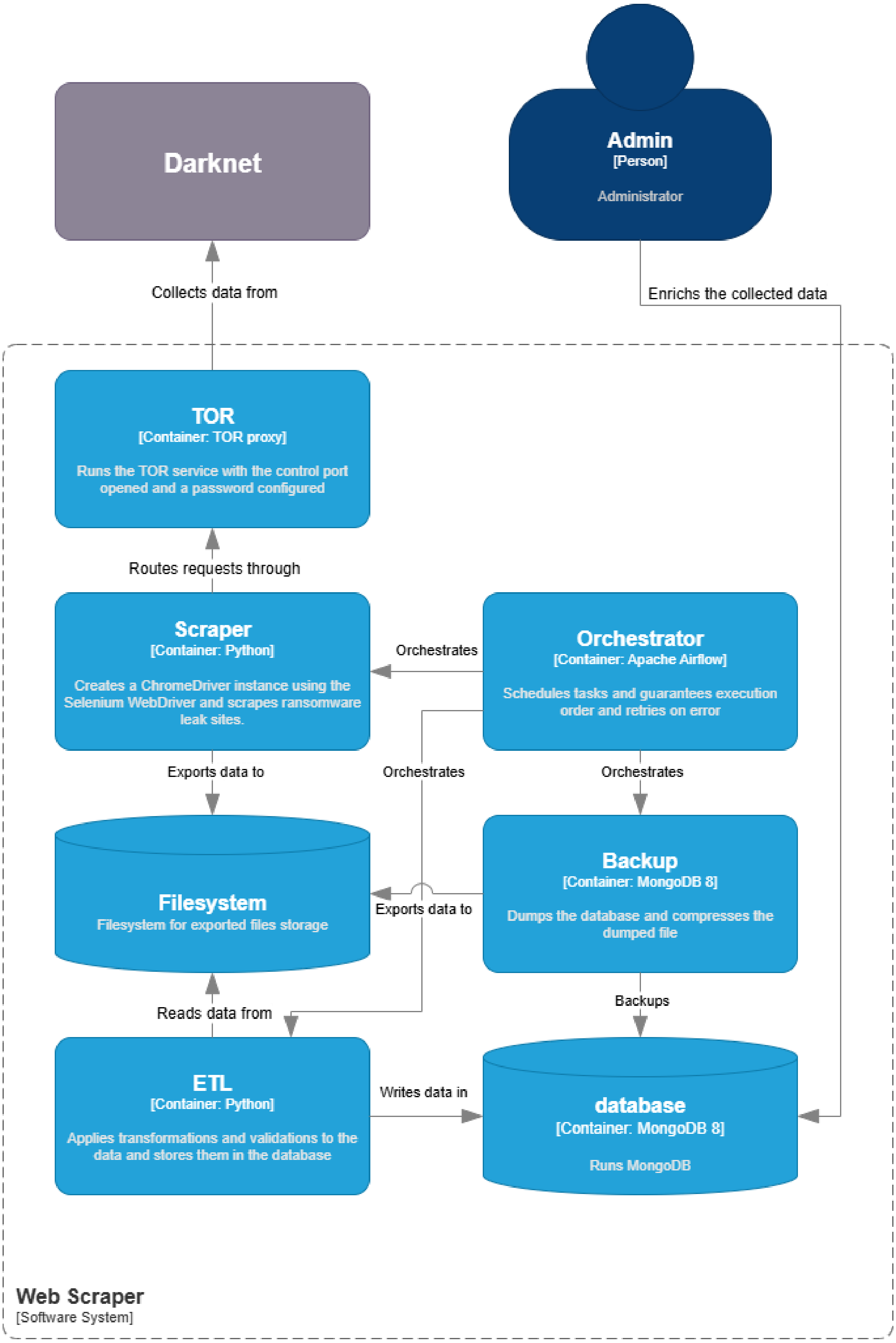

Darknet
Admin
[Person]
Administrator
Collects data from
Enrichs the collected data
TOR
[Container: TOR proxy]
Runs the TOR service with the control port opened and a password configured
Routes requests through
Scraper
[Container: Python]
Creates a ChromeDriver instance using the Selenium WebDriver and scrapes ransomware leak sites.
Orchestrates
Orchestrator
[Container: Apache Airflow]
Schedules tasks and guarantees execution order and retries on error
Exports data to
Orchestrates
Orchestrates
Filesystem
Filesystem for exported files storage
Exports data to
Backup
[Container: MongoDB 8]
Dumps the database and compresses the dumped file
Reads data from
Backups
ETL
[Container: Python]
Applies transformations and validations to the data and stores them in the database
Writes data in
database
[Container: MongoDB 8]
Runs MongoDB
Web Scraper
[Software System]

Figure 2. Web scraper container diagram [24].

1.1 Web scraper service

To obtain relevant and timely data on ransomware cyberattacks to Spanish companies, a set of targeted websites were selected for automated scraping. The sources were chosen among the ransomware leak sites maintained by ransomware groups to publish data from breached organizations. Since these sites are not indexed by traditional search engines and often use onion addresses, they were accessed through the TOR network.

A technical challenge inherent to scraping these sites is their reliance on JavaScript to dynamically render their content within the client's browser. This presents a challenge for traditional scraping tools, which are typically incapable of executing JavaScript. In response to this limitation, the scraping mechanism was implemented using Selenium, a robust browser automation framework well-known for its capability to emulate real-user behaviour and engage with JavaScript-driven web pages.

To accurately simulate a browser session, Selenium is configured to instantiate a session of Google Chrome via ChromeDriver, Google's standalone server for browser control. This setup allows the scraper to behave similarly to a human-operated browser, which is crucial for evading anti-bot mechanisms. The Selenium browser instance is routed through a SOCKS proxy, which is bound to a running TOR service. This configuration ensures that all web traffic is anonymized and encrypted according to the access protocols of the darknet.

The entire scraping logic, including the configuration of Selenium and the integration with the TOR network, is implemented using the Python programming language, which is widely used in web automation tasks due to its readability, ease of use and the extensive ecosystem of libraries.

However, some of these websites also include anti-bot protection. Therefore, a set of evasion techniques were employed to improve the robustness and stealth of the scraping process. These techniques aim to mimic natural human browsing behaviour and minimize the risk of detection or blocking. The following strategies were integrated:

- IP rotation via TOR.

The scraper periodically changes its TOR exit node managing TOR's control port with the stem library [25]. This approach cycles through different IP addresses preventing request patterns from being easily linked and blocked by target websites.

- Dynamic User-Agent.

Each browsing session is assigned a randomly selected user-agent string via python's fake-useragent library [26]. This simulates access from various browsers and operating systems avoiding detection mechanisms that flag unusual or repetitive user-agent headers.

- Randomized mouse movements, delays and window size.

Randomized mouse gestures and pseudo-random wait times between clicks were introduced using Selenium. These actions imitate a real user navigating a page, reducing the likelihood of being identified as a bot. Also, aleatory window size is implemented to simulate different devices.

- Non-headless mode.

Headless browsers are often more easily identified as automated tools due to missing or altered browser features such as window size, graphical rendering and plugin availability. This approach allows a more accurate execution of JavaScript-heavy content.

The scraped data consists of semi-structured records collected from the dark web that are exported to the filesystem in JSON format. The selection of JSON format is justified by its native integration with web technologies, support for nested and dynamic structures and interoperability with NoSQL databases.

Each record captures a different cyberattack published on a ransomware leak site. The general structure of a record includes, but is not necessarily limited to, the fields presented in the following table. Every scraped field is exported as a string and will be transformed during the ETL process.

Field	Description
cybercriminal.name	Name of the ransomware group
target.name	Target's name
target.revenue	Target's annual revenue
target.size	Target's number of employees
target.website	Target's website
disclosures	Completed and total disclosures of target's data
leakSize	Size of the target's data leak
leakFiles	Number of files leaked
scrapedAt	Timestamp of scrape

Table 1. General scraped data structure.

1.2 TOR service

The configuration of the TOR service was designed to facilitate controlled IP rotation. The TOR control port was enabled and configured as an interface through which the scraper can programmatically communicate with the TOR daemon to request new circuits and thus obtain new exit node IP addresses. Unauthorized access to the TOR control interface is prevented by configuring a secure authentication password.

1.3 ETL service

The ETL process is necessary to transform the semi-structured raw data acquired by the scraper into a normalized, database-compatible format. This transformation involves data cleaning, parsing, validation and standardization procedures that ensure consistency and accuracy of the stored information.

As only Spanish cyberattacks are scraped, the scale of the data processed in this pipeline is relatively modest. The frequency and volume of new records being scraped and the weight of the performed transformations do not justify the complexity or overhead of using distributed data processing frameworks. Therefore, Python was chosen due to its simplicity and maintainability. It is also particularly well-suited for working with JSON formats via the json library [27], and for integrating with MongoDB, which is also natively supported through the pymongo library [28].

The ETL process begins with the ingestion of scraped data in the form of JSON files from the filesystem that may contain a list of records. This data, once loaded, is parsed from JSON format into Python-native data structures, enabling the application of a set of normalization rules to ensure consistent datatypes. The cleaned and structured records are inserted into a MongoDB collection shared with the statistics portal application.

The normalization rules applied are listed below:

Field	Transformation
cybercriminal.name	Transformed to lowercase
target.revenue	Decomposed into its number value converted to euros, abbreviation, and coin
target.size	Casted to number type
target.website	Check whether constitutes a well-formed URL
disclosures	Separated into number of completed and total disclosures
leakSize	Separated into its number magnitude and its corresponding unit
leakFiles	Casted to numeric type after the removal of thousand separators

scrapedAt	Casted to Date type

Table 2. ETL transformations applied.

As part of the transformation phase within the ETL process, any field containing an empty, null, or invalid value is programmatically removed from the Python-native data structure to prevent its insertion into the database. This cleaning step ensures that the resulting records are free from non-informative fields from the final data representation.

1.4 Database service

The decision of the database system was driven by the heterogeneity of the scraped data, as individual records may contain differing fields, nested structures and optional data depending on the scraped web. Given these requirements, MongoDB was selected as the most suitable database solution for this context. MongoDB provides schema flexibility through its schema-less model, native JSON support and efficient handling of nested data, but also facilitates horizontal scalability through sharding mechanisms.

Although MongoDB offers robust support for horizontal scalability, in this project it was not feasible to implement a sharded cluster due to resource constraints in the development and deployment environment. As a result, the system currently operates using a single-node MongoDB instance, which is more lightweight and sufficient for the project's current scale and data volume.

Grounded in the structural characteristics of the scraped data and prioritizing the use of truthy values, each cyberattack also stores the information about both the cybercriminal and the target. However, as I mentioned earlier in this dissertation, the project fundamentally employs a schema-less data model to ensure that any field is optional and accepted by the database. In the table presented below, the general structure of the scraped data is gathered.

Field	Datatype
cybercriminal.name	string
target.name	string
target.revenue.value	number
target.revenue.abbreviation	string
target.revenue.coin	string
target.size	number

target.website	string
disclosures.completed	number
disclosures.total	number
leakSize.value	number
leakSize.unit	string
leakFiles	number
scrapedAt	Date

Table 3. Database general scraped schema.

As mentioned earlier, this data shall be enriched by the administrator. To provide context and enable further data analysis, the following fields shall be added to each cyberattack.

Field	Datatype
cybercriminal.email	String
cybercriminal.onion	string
cybercriminal.telegram	string
cybercriminal.wallet	String
target.region	string
target.sector	string
description	string
reputationalImpact	string

Table 4. Database general extra schema.

1.5 Orchestrator service

In this multi-component data pipeline, where the scraping and the ETL processes are separated into different services, orchestration becomes essential to ensure that each task is executed in the correct order, at the right time and under the right conditions. Moreover, it provides improved scalability, error handling and monitoring. To fulfil these orchestration requirements, Apache Airflow was selected based on its maturity, robust scheduling capabilities, integration and DAG-Based workflow modelling.

In addition to the core scraping and ETL processes, several supplementary validation and management tasks were designed to enhance the overall robustness of the pipeline and facilitate comprehensive error handling. The DAG is configured to run daily during nighttime with the ability to retry one more time on error. The complete DAG is composed of the following tasks, which are executed only after the competition of the one before:

1. run_scraper

Executes the web scraping process within a Docker container, mounting the raw data file for future processing. The exported filename is passed through an environment variable.

2. wait_for_scraped_file

Implements a sensor that monitors the presence of the scraper output file before running the etl process.

3. validate_scraped_file

Performs a basic validation of the JSON file to confirm its presence and its correct syntax.

4. run_etl

Initiates the ETL workflow inside a Docker container to load, transform, normalize and load the validated data into the target database. The input filename is facilitated using a environment variable.

5. cleanup_files

Cleans up historical data files beyond a specified retention period to optimize storage utilization.

Additionally, to ensure data durability and facilitate disaster recovery, automated backups of the MongoDB database were integrated into the system's workflow. A dedicated DAG was developed to perform monthly backups by executing backup commands in an isolated Docker environment. The backup files are exported to a mounted directory shared with the host system, where the dump is subsequently compressed.

2 Web app design

The selection of the full-stack architecture was based on scalability, maintainability and robustness, with a modular development approach. To this end, the web application architecture is divided into a backend service to query data from the database and provide an API to access it and a frontend service to visualize the data in a user-friendly and attractive way. This architectural separation promotes independent development, testing and scaling of each layer, thereby enhancing overall system flexibility and maintainability. The complete structure is shown in the following diagram. It is important to note that the database is the same explained earlier.

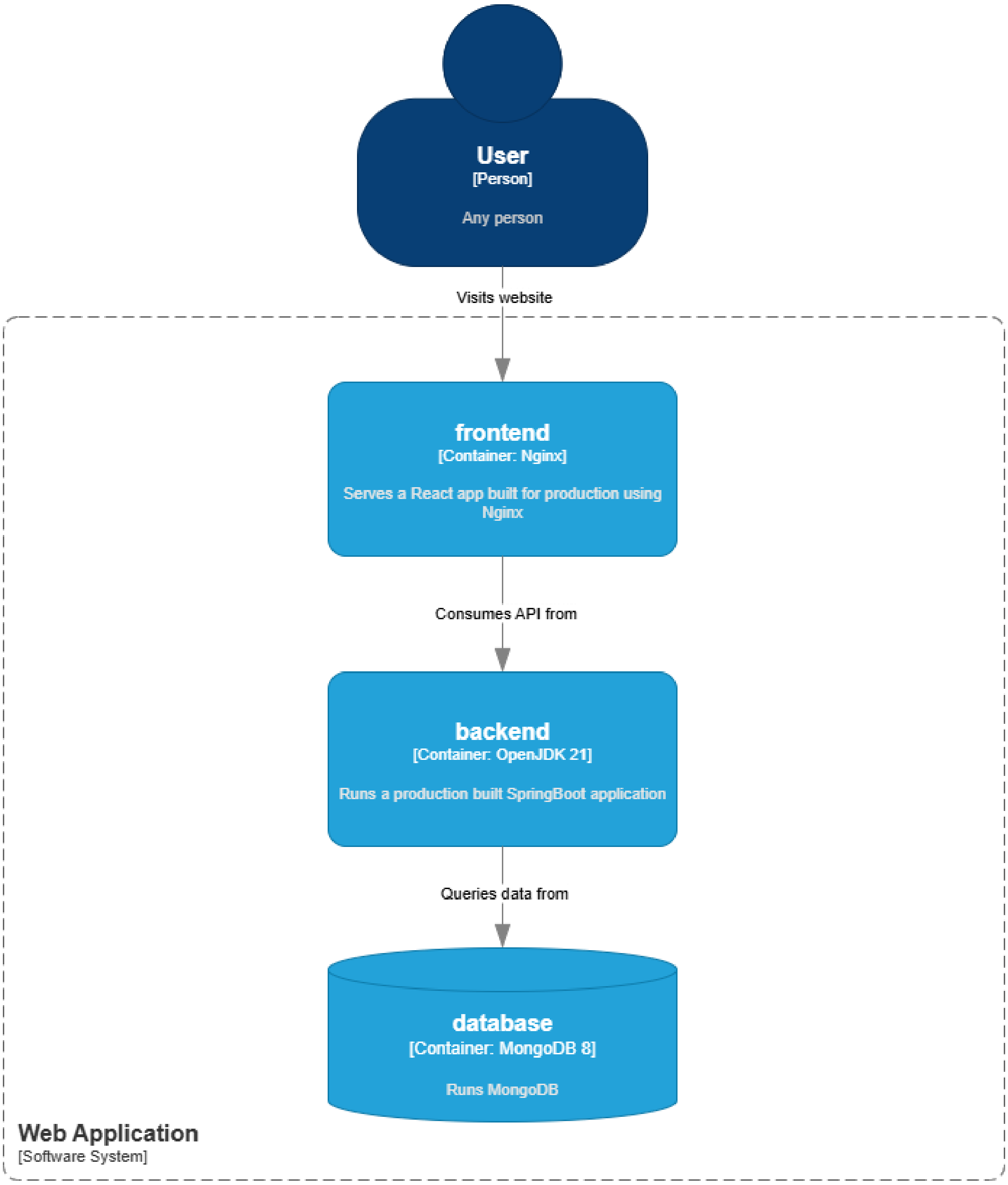


Figure 3. Web Application container diagram [24].

2.1 Backend service

The backend of the statistics portal was developed using Spring Boot. This technology is used as it provides an optimized production-ready Java framework that simplifies backend development. It also supports MongoDB natively, promotes code maintainability, promotes scalability and offers robust security. The dependencies used in the project are listed in the table below.

Spring Boot dependency	Feature
Spring Boot Starter Data Mongodb	Connects to MongoDB and uses Spring Data MongoDB
Spring Boot Starter Web	Starter for building web applications using Spring MVC
Lombok	Provides annotations to simplify Java development
Spring Boot Started Tomcat	Embeds Tomcat for development
Spring Boot DevTools	Provides developer tools for development

Table 5. Spring Boot dependencies [29].

Given that any information needs to be stored in session, a RESTful API was implemented ensuring scalability and simplicity. The API exposes a set of endpoints, detailed below, that were designed to facilitate efficient access to the cyberattacks data in alignment with the objectives of the project. The endpoints are specified in Spanish as it is the main audience target for this project.

- /api/health

This endpoint is enabled to support health monitoring by external systems, primarily intended for Docker's automated health checks. When invoked, the endpoint returns a lightweight response.

- /api/ciberataques

This endpoint enables the server to query and retrieve all the stored cyberattacks records. It returns a simplified data structure that shows the cybercriminal name and the target name together with additional information such as the leak size or the date of the attack detection.

- /api/ciberataques/{id}

Through this endpoint, access to the complete data structure of a specific cyberattack, identified by its unique id, as stored in the database.

- /api/ciberdelincuentes

This endpoint provides access to all registered cybercriminal entities. It returns the cybercriminal name and the computed total number of associated attacks to enable easy identification and a simple overview.

- /api/ciberdelincuentes/{name}

Fetching data from this endpoint will return detailed information about the cybercriminal group identified by the parameter name. Additionally, statistics and the simplified data structure of their last cyberattacks are also provided. The statistics data structure is detailed in the following section.

Given that any cybercriminal group may provide or utilize varying information across different cyberattacks, such as its cryptocurrency wallet addresses or contact emails, only the system is designed to aggregate and display only the unique values associated with each group.

- /api/estadisticas

A single statistical summary of the cyberattack data is returned from this endpoint. It is logically divided into two distinct sections: the first one provides aggregated counts across all cyberattacks whereas the second provides multiple substructures, each representing a specific category, such as region or sector, along with their corresponding occurrence counts within the data. Each substructure is also sorted in descending order based on frequency to facilitate clearer and more impactful data visualizations in the frontend application.

2.2 Frontend service

The frontend service functions as the primary interface between the user and the backend service, thereby the technological stack was selected with the goal of achieving high performance, modular development and robust user experience, particularly in the context of rendering dynamic, data-intensive content.

React was adopted as the core framework due to its component-based architecture, which promotes reusability and maintainability, but also due to its efficiency in rendering dynamic user interfaces and its maturity. React is integrated with TypeScript to provide static type checking at compile time, which significantly reduces runtime errors and enhances code robustness and developer productivity. The whole application is built into a Nginx service that provides high performance and security under heavy loads, facilitating future improvements like load balancing in the future.

This stack, combined with the D3.js library and Bootstrap ensures a frontend that is both functionally rich and visually responsive. D3.js enables the rendering of advance, interactive data visualizations, allowing users to explore complex cyberattack data with clarity and precision. Bootstrap, on the other hand, provides a responsive design framework that ensures consistent and adaptative layouts across a wide range of modern devices.

RansomWatch was selected as official title and brand identity of the web application. This name reflects the project's core objective: to monitor, aggregate and present data related to ransomware activities in a transparent and accessible manner. This name is displayed in both the header and footer of the frontend interface.

In addition to branding, the header is designed to facilitate user interaction by incorporating three navigation buttons, which allow intuitive access to the cyberattacks interface, the cybercriminals interface and the statistics interface.

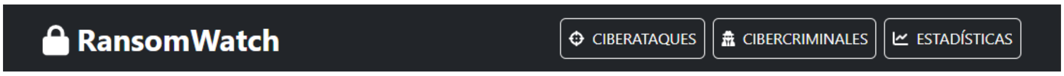


Figure 4. Web application's header.

Figure 5. Web application's footer.

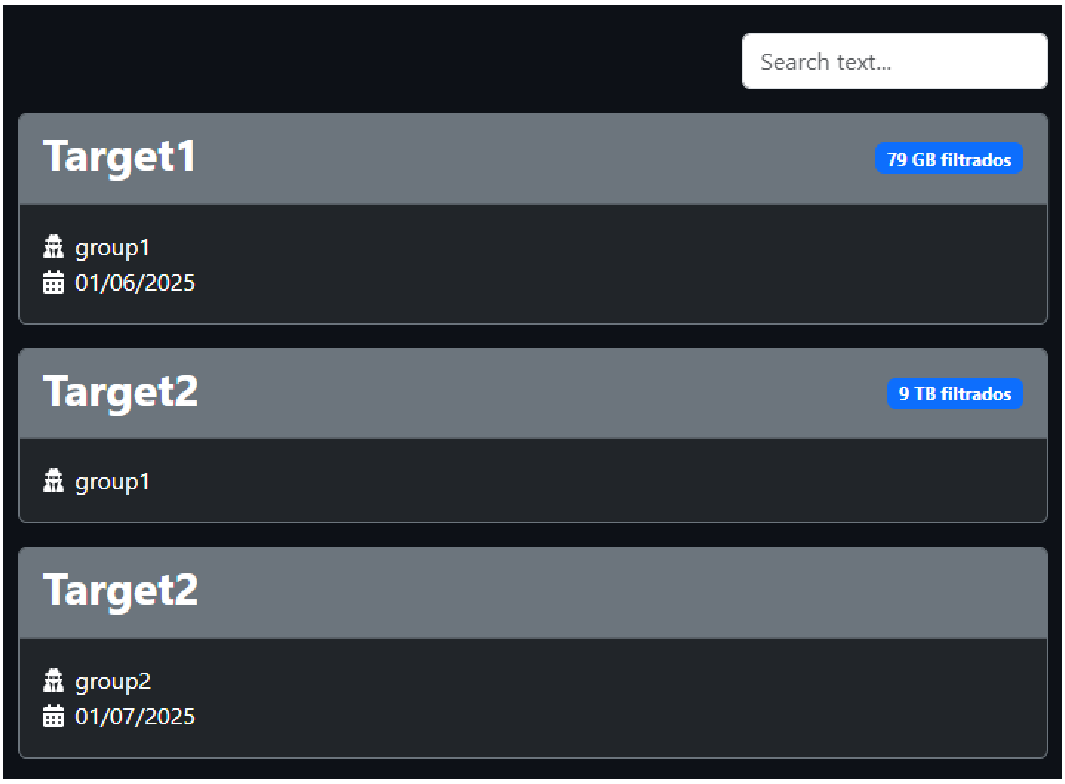
The frontend service consumes the RESTful API facilitated by the backend, dynamically fetching data from the endpoints previously outlined. It is important to note, however, that any field is optional within the database, thus null checks are always performed prior to render the content.

The service delivers the following user interfaces, each mapped to the same route with the prefix "/api" within the application:

- Cyberattacks interface

This interface, mapped to "/ciberataques", presents a summarized list of all recorded cyberattacks, each entry is presented as a Bootstrap card with the cyberattack target name as the title. The rest of the data is shown in the body of the card and the leak size, if present, is shown with a Bootstrap Badge to draw the user's attention.

To enhance the user experience, the number of cyberattacks displayed is limited to a fixed count on initial load to prevent excessive rendering. By scrolling, more cyberattacks are continuously rendered. To facilitate easy filtering, an input text was added to filter by cybercriminal name or target name.



Search text...

Target1 — 79 GB filtrados

group1
01/06/2025

Target2 — 9 TB filtrados

group1

Target2

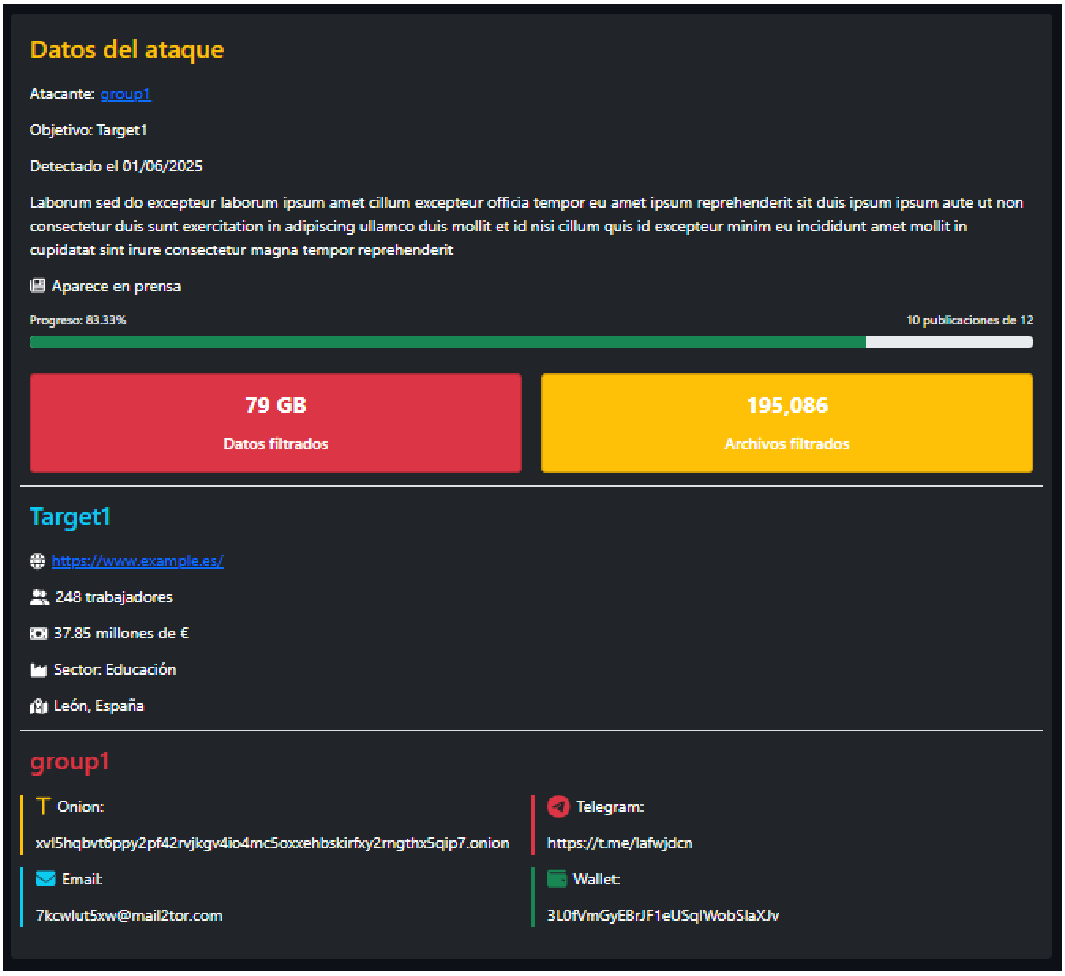
group2
01/07/2025

Figure 6. Cyberattacks interface

- Cyberattack details interface

This interface, mapped at "/ciberataques/{id}", renders the complete cyberattack data structure clean, colourful and user-friendly layout. The content is logically segmented into three main sections to improve readability. The sections are attack metadata, target information and cybercriminal group data.

The first section shows general information about the cyberattack such as the cybercriminal name, the target, the detected date, the description and other cyberattacks' data. The second shows in a user-friendly way known information about the breached organization and the third displays the stored data about the cybercriminal group.



Datos del ataque

Atacante: group1

Objetivo: Target1

Detectado el 01/06/2025

Laborum sed do excepteur laborum ipsum amet cillum excepteur officia tempor eu amet ipsum reprehenderit sit duis ipsum ipsum aute ut non consectetur duis sunt exercitation in adipiscing ullamco duis mollit et id nisi cillum quis id excepteur minim eu incididunt amet mollit in cupidatat sint irure consectetur magna tempor reprehenderit

Aparece en prensa

Progreso: 83.33% 10 publicaciones de 12

79 GB
Datos filtrados

195,086
Archivos filtrados

Target1

https://www.example.es/

248 trabajadores

37.85 millones de €

Sector: Educación

León, España

group1

Onion:
xvl5hqbvt6ppy2pf42rvjkgv4io4mc5oxxehbskirfxy2mgthx5qip7.onion

Telegram:
https://t.me/lafwjdcn

Email:
7kcwlut5xw@mail2tor.com

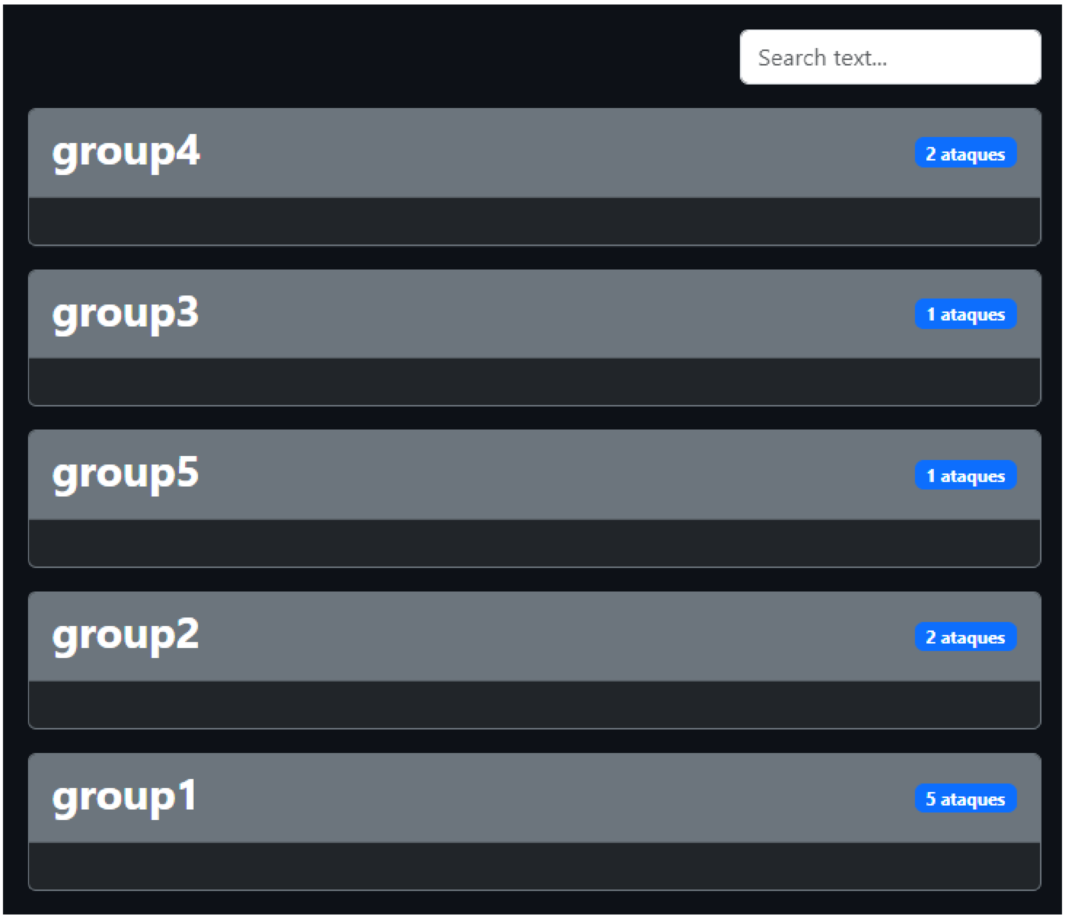
Wallet:
3L0fVmGyEBrJF1eUSqIWobSlaXJv

Figure 7. Cyberattack details interface.

- Cybercriminal groups interface

This interface is mapped to the "/ciberdelincuentes" endpoint. Here, each known cybercriminal group is displayed using a Bootstrap card, mirroring the design logic and animations used in the cyberattack list interface. Within this card, the cybercriminal name is used as the title and the number of attacks is shown with a Bootstrap Badge to highlight.

The scrolling and filtering techniques used in the Cyberattacks interface are also used within the Cybercriminal groups interface. However, data is only filtered by the cybercriminal group name in this case.



Search text...

group4 — 2 ataques

group3 — 1 ataques

group5 — 1 ataques

group2 — 2 ataques

group1 — 5 ataques

Figure 8. Cybercriminal groups interface.

- Cybercriminal group details

This interface is provided in the "/ciberdelincuentes/{name}" context. It is divided into three different sections according to the data returned by the backend. The first section shows the unique information about the cybercriminal group as in the cybercriminal group section within the cyberattack details interface. The last attacks are presented at the bottom of the page in the same way as the cyberattacks interface.

Finally, the statistics section, which is shown between the other two, is designed to ensure a coherent, intuitive and analytically effective user experience.

These visualizations were developed based on the data provided by the backend and using the D3.js library to ensure precision and interactivity, presenting the value of the corresponding section on hover. The design purpose of each chart is detailed below:

- Number of cyberattacks time chart

This time chart visualizes the temporal evolution of cyberattacks performed by the selected cybercriminal group. The X-axis denotes the time dimension and the Y-axis quantifies the number of cyberattacks per interval. The data points are connected via a continuous line to highlight the ransomware activity over time. To prevent extra text noise within the X-axis, only dates when a cyberattack was performed are displayed.

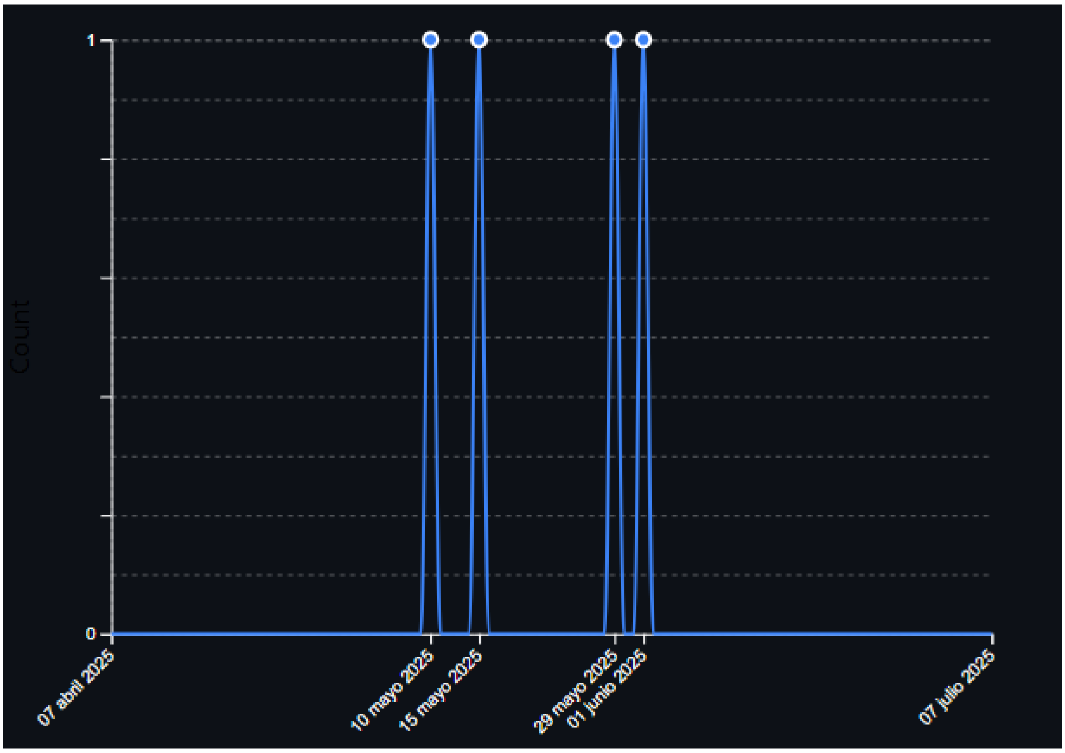


Figure 9. Number of cyberattacks time chart.

- Number of cyberattacks by region

Presented as a choropleth map, represents the distribution of ransomware attacks by region, with each geographic area shaded according to a gradient colour scale that reflects the corresponding number of attacks.

Regions with higher attack count are rendered in darker colours, providing an immediate visualization of global hotspots. However, a legend is also provided to facilitate the understanding.

Additionally, a zoom facility was configured for this project to enable the user to focus on the interested areas of the map.

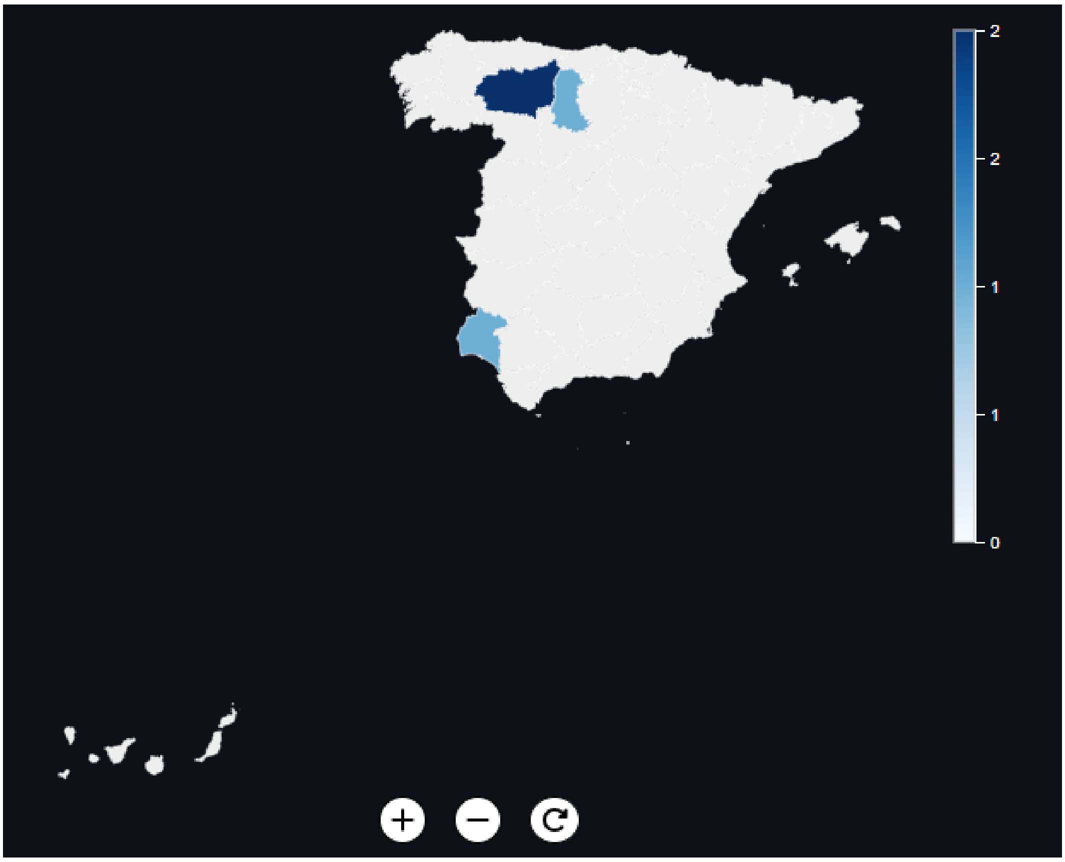


Figure 10. Number of cyberattacks by region.

- Number of cyberattacks by target

This bar chart represents the organizations most frequently targeted by ransomware groups ranked in descending order of attack count. The plot is displayed horizontally to improve label readability and enable easy comparison across the different target counts. This representation highlights concentration trends, such as repeated targeting of specific sectors or organizations.

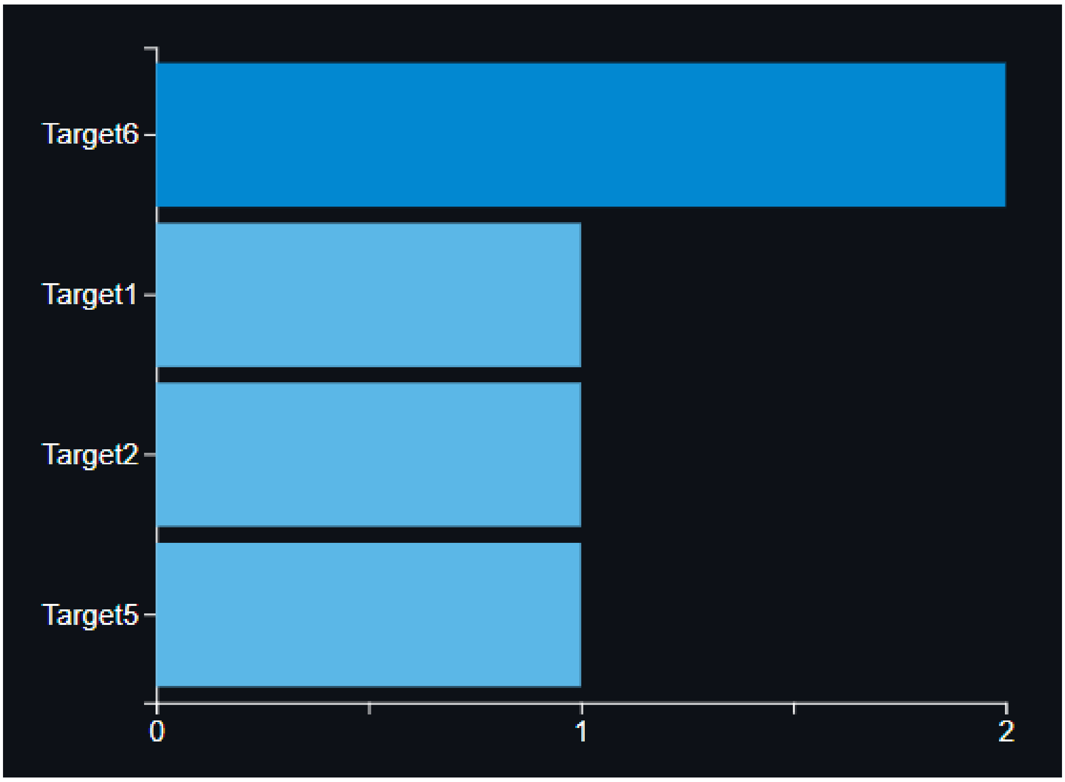


Figure 11. Number of cyberattacks by target.

- Frequency of cyberattacks by sector

This pie chart represents the percentage of cyberattacks executed among the different industrial sectors. The size of each slice is proportional to the relative frequency of attacks in that sector.

The pie chart enables quick understanding of sectoral distribution, illustrating which industries are most affected.

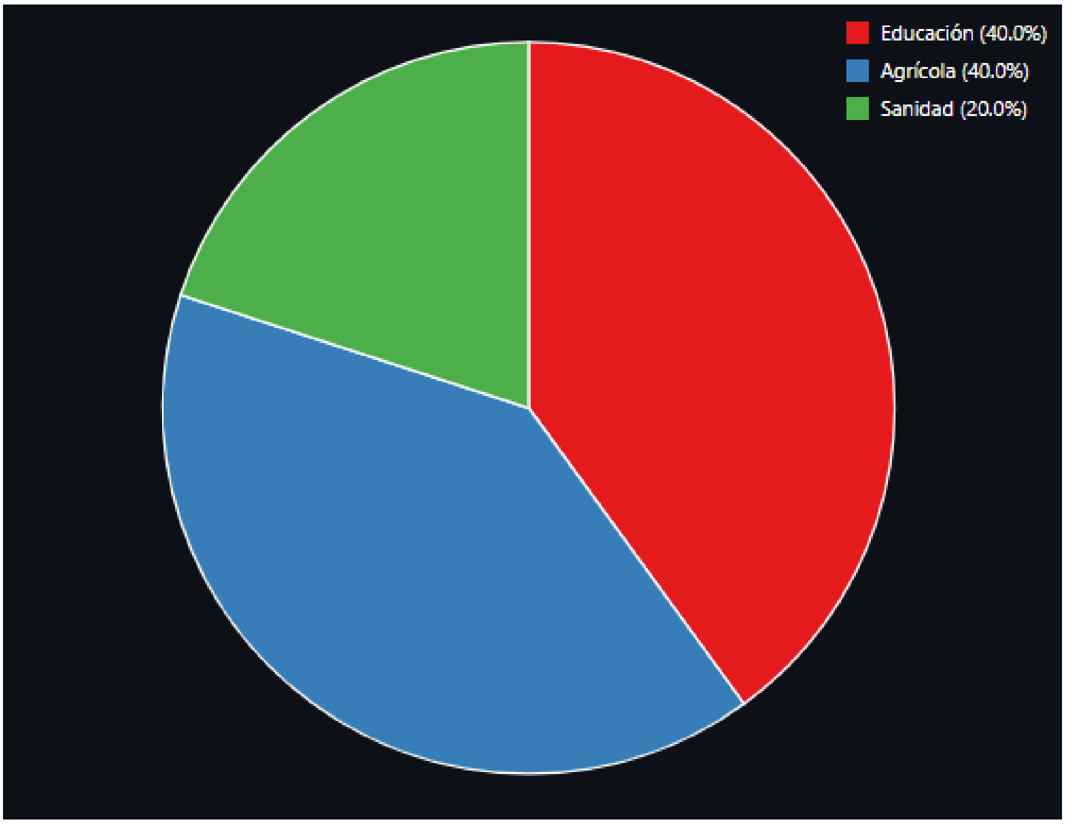
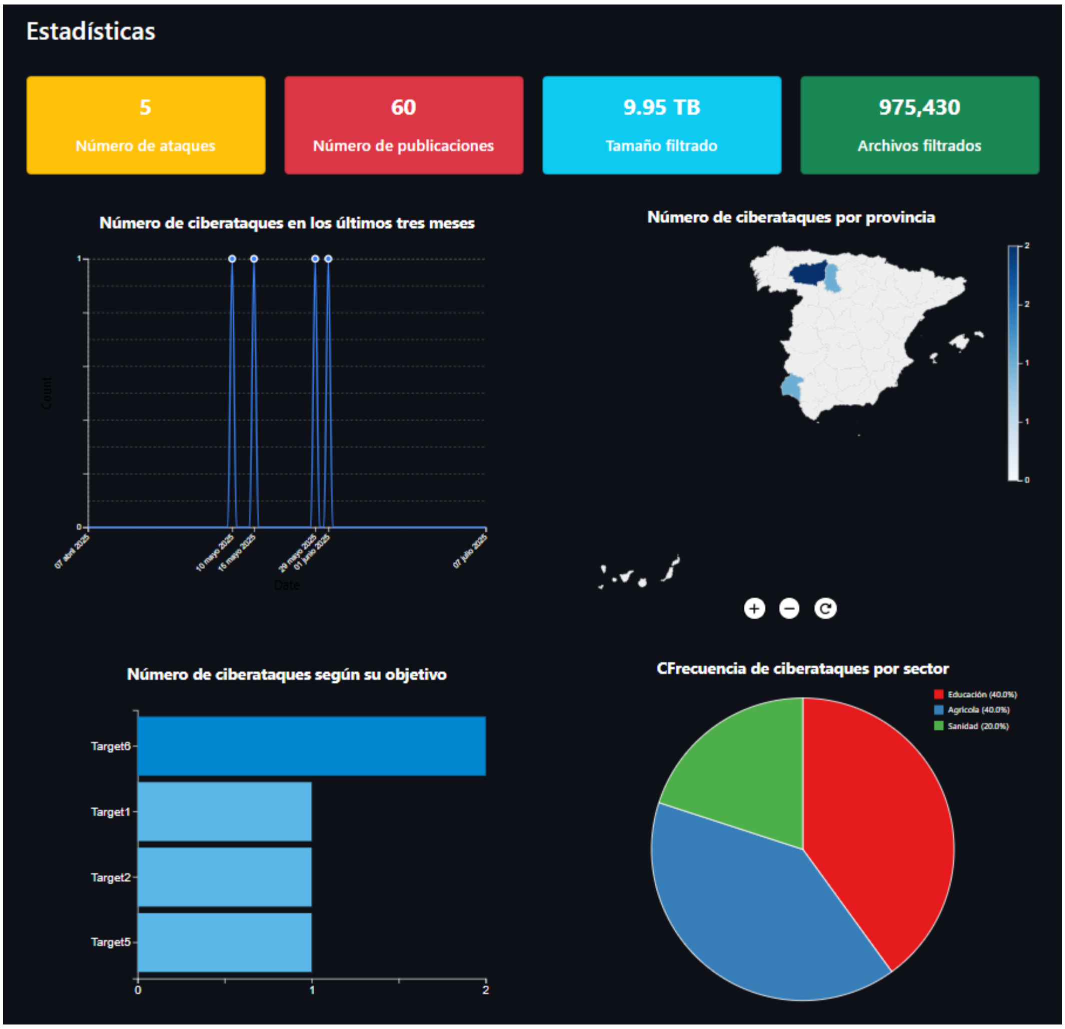


Figure 12. Frequency of cyberattacks by sector.

The layout was carefully designed with visual hierarchy in mind. It follows a top-down structure that first presents aggregated totals followed by a grid of visualizations organized into rows of two charts each. The summary statistics are displayed providing an immediate and high-level overview of the dataset whereas the grid facilitates comprehensive data analytics. This format was selected to optimize screen space by aligning charts and prevent users for excessive scrolling, certain space is also guaranteed between visualizations to ensure a clean view. The layout described can be visualized in the following figure.



Estadísticas

5 Número de ataques

60 Número de publicaciones

9.95 TB Tamaño filtrado

975,430 Archivos filtrados

Figure 13. Cybercriminal group statistics section.

- Global statistics interface

This interface is provided in the context "/estadisticas" and is similar to the statistics section of the cybercriminal details interface. However, this interface represents one chart differently and displays a different one.

- Number of cyberattacks by date

First, as the Y-axis of the time chart that represents the number of cyberattacks per date can get noisy, only the data of the last three months is represented in a choropleth calendar. This calendar enables the user to understand the very last information.

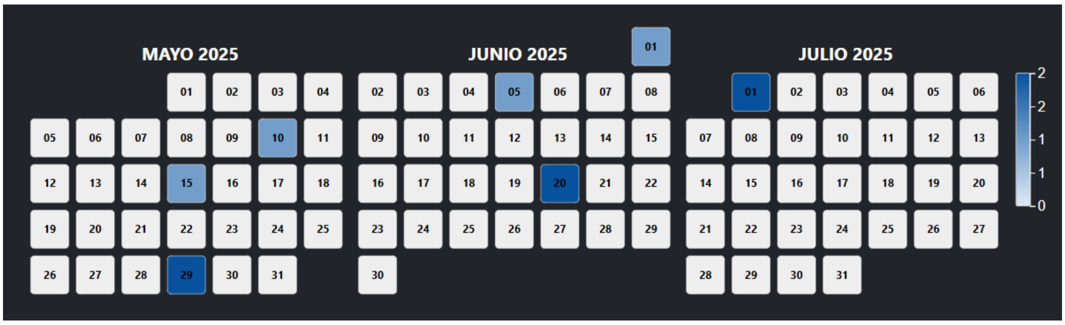


Figure 14. Number of cyberattacks by date

- Number of cyberattacks by cybercriminal group

A vertical bar chart is used to represent the groups that performed the most attacks.

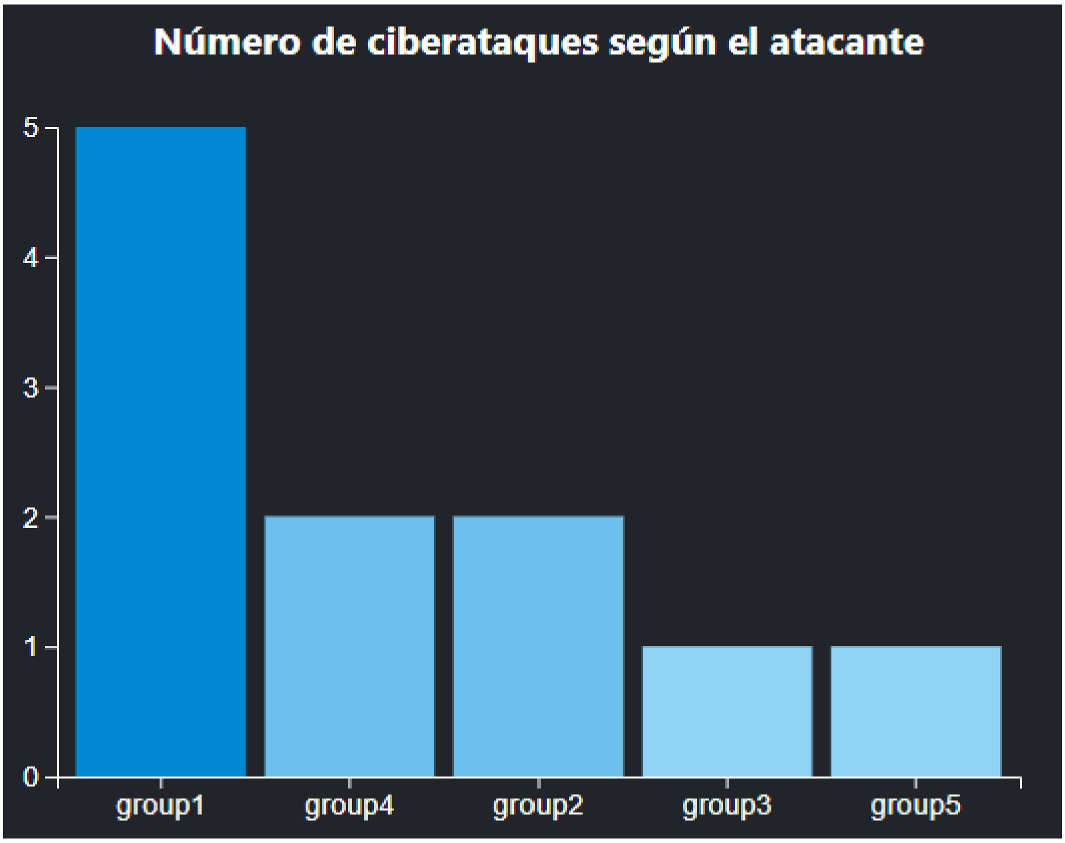
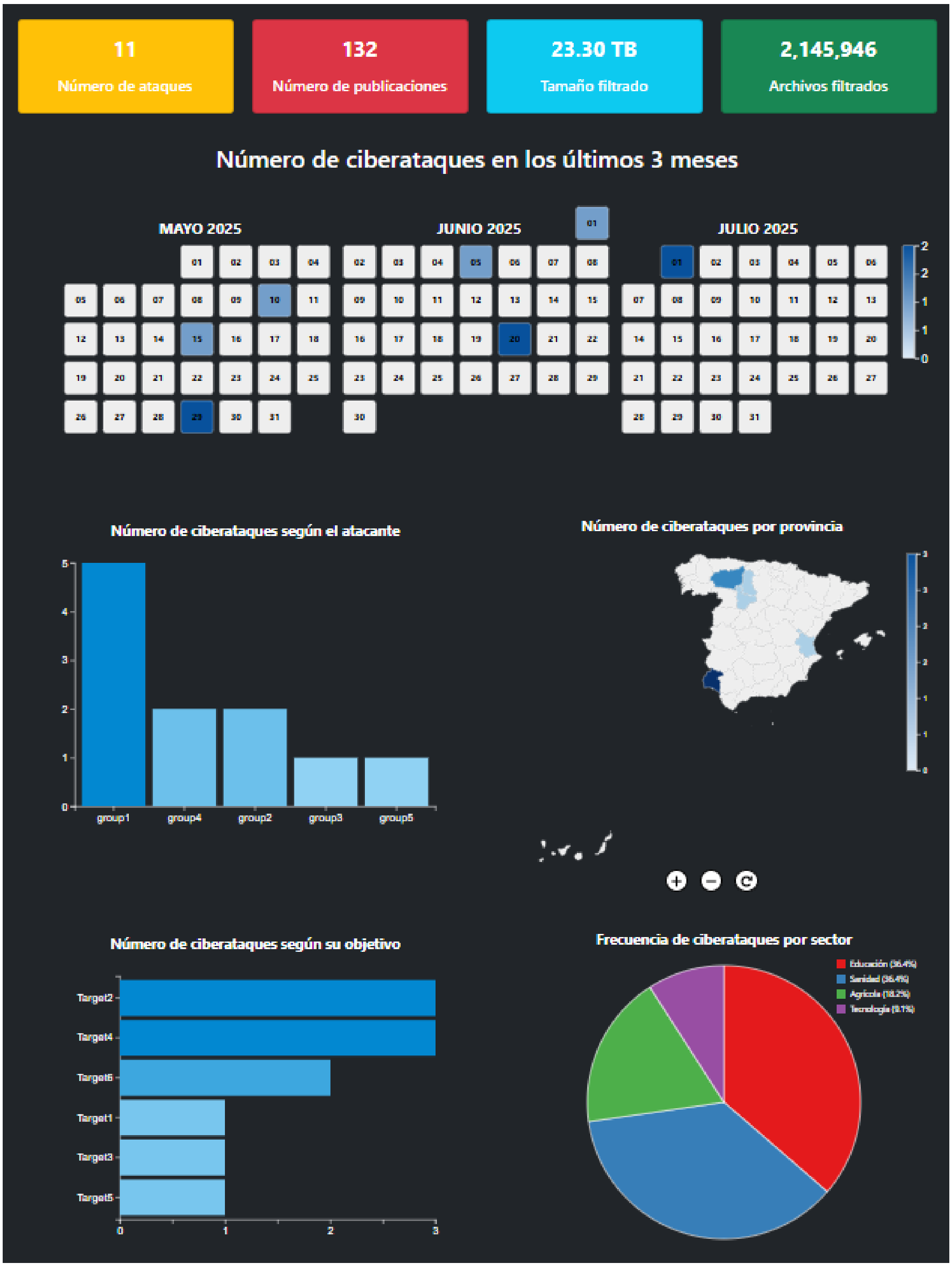


Figure 15. Number of cyberattacks by group.

The global statistics of the data and some modifications were made, both in the layout and the charts used. This time, the time chart is substituted with the bar chart and the time data was presented as a single row chart right below the aggregated statistics.



11 Número de ataques

132 Número de publicaciones

23.30 TB Tamaño filtrado

2,145,946 Archivos filtrados

Figure 16. Global statistics layout.

By analysing the processed and enriched data, several meaningful tendencies and cybercriminal group profiles can be observed. These patterns not only focus on the operational behaviour of these groups but also contribute to better threat intelligence and proactive defence strategies.

- Attack frequency and intensity

Certain cybercriminal groups may appear repeatedly across multiple attacks, suggesting a persistent and aggressive operational campaigns. Furthermore, by examining the timing of these attacks, potential seasonal peaks or specific temporal patterns can be identified, offering insights into the planning and execution cycles of these threat actors.

- Target or sector preferences

Analysis often reveals that specific sectors or organizations are targeted more frequently. This may be due to factors such as weaker cybersecurity measures, higher perceived ransom value or the availability of sensitive data. Recognizing these preferences allows for sector-specific recommendations and improved risk management strategies.

- Regional impact

Attacks may concentrate in specific geographic regions. This clustering could be influenced by language familiarity, regional geopolitics or economic motivations.

- Leak volume

The volume of leaked data together with the number of public disclosures are often correlated with extortion strategies. High leak volumes may indicate either broad exfiltration of organizational data or targeted extraction of sensitive information, both used to maximize pressure on victims. These metrics also reveal whether a group tends to focus on mass data harvesting or strategic compromise.

- Contact Methods

The variety of anonymous contact methods highlights the importance of anonymity and operational security for these groups. Monitoring these channels can help in tracking threat actor behaviour and potentially disrupting communication infrastructures.

These observed tendencies, derived from structured darknet data, support a better academic understanding of cybercriminal ecosystems.

3 Ethical aspects

The scraper was deliberately designed to avoid intrusive or illegal actions such as credential harvesting, brute-forcing or denial-of-service attacks. Only publicly accessible pages are visited, and the system's execution frequency is limited to prevent disruption of target sites. In addition, the project strictly adheres to the principles outlined in the Spanish adaption of the RGPD and the LOPDGDD, ensuring that no personal data is collected, processed or stored.

Specifically, the data targeted by the scraper consists exclusively of information voluntarily published by cybercriminal groups on the darknet, with no direct association to identifiable individuals. Furthermore, the collected data is also used exclusively for academic and research purposes, with no intent to identify individuals or exploit the information for commercial gain.

Administrators and developers managing this system must be knowledgeable about both RGPD and LOPGDD regulations. In the event that the system is extended to store or process personal data, it must comply with applicable legal requirements, including but not limited to the principles of lawfulness, fairness, transparency, purpose limitation, data minimization, accuracy, storage limitation, integrity and confidentiality as defined under Articles 5 and 6 of the RGPD and reinforced by Articles 4 to 8 of the LOPDGDD.

Should personal data need to be stores, it must be done under a legitimate basis, as outlined in Article 89 of the RGPD for statistical purposes, and with the appropriate technical and organizational measures to protect data subject’s right, such as pseudonymisation.

VI RELATED WORKS

- ransomware.live.

Ransomware.live is a personal initiative developed by Julien Mousqueton with the aim of increasing visibility into global ransomware activity. The platform serves as a transparent, accessible and continuously updated inventory of ransomware-related incidents offering valuable insights to cybersecurity professionals, researchers and the public. Unlike other platforms, ransomware.live is free of corporate influence, advertisements or subscription barriers.

The system tracks ransomware attacks by providing a transparent, accessible and continuously updated inventory of ransomware-related activity. The platform extracts data from publicly disclosed victim data from multiple ransomware groups' DLS and presents it in a clear, organized and actionable format.

It provides detailed information on recent cyberattacks, victims, ransomware groups, and statistics gathered by country, month, sector and ransomware group to facilitate the identification of trends. Beyond static data reporting, it also includes features such as screenshots of ransom notes and negotiation chats, offering deeper insights into threat actor behaviour and tactics.

Furthermore, the platform implements free notifications via an open-source mobile app, where the user stays informed about global, country-specific or sector-specific alerts. No account or personal information is required and the user can unsubscribe anytime [30].

- ransom-db.

Ransom-db is a free, publicly accessible web that aggregates and monitors information on ransomware attacks. Its core mission is to provide real-time updates on ransomware activity, making it a valuable resource for cybersecurity professionals, researches and organizations seeking timely intelligence on emerging threats.

The website collects and posts ransomware incidents published on Twitter and enriches it with basic information about live ransomware updates in real-time together with an AI summary, data about ransomware groups' status and statistics.

One of the standout features of Ransom-db is its ransomware decryption search, a free tool that assists users in identifying whether a decryption utility is available for specific ransomware variants. This can be especially helpful for affected entities attempting to recover encrypted data without yielding to ransom demands.

Beyond the public information, the ransom-db team has also developed an automated CTI platform. Acting as a digital reconnaissance team, they aim to deliver strategic advantage to protect digital assets. They facilitate cyberthreats and assets monitoring, access to their ransom API and real-time ransomware updates from different sources such as Telegram, the Dark Web or news. Through the mentioned API, users can integrate this intelligence directly into their security systems or workflows, enhancing automation and situational awareness in operational environments [31].

- ransomfeed.

Ransomfeed is an Italian cybersecurity project developed by Darío Fadda, dedicated to monitoring ransomware activity through automated scraping. The platform collects and stores ransomware claims in a permanent RSS feed, making it easily accessible to users seeking real-time insights into both Italian and international ransomware incidents.

The mission of ransomfeed is to provide a user-friendly interface that enables individuals, organisations and security teams to understand the evolving ransomware ecosystem. Through its continuously updated monitoring platform, the project supports threat identification, proactive defence and rapid response and mitigation, helping improve overall resilience against ransomware attacks.

The platform delivers detailed analytics through a comprehensive statistics interface, allowing users to explore current ransomware trends by country, sector, cybercriminal group, region of Italy and attack date. This granularity supports both strategic and operational decision-making, particularly within the Italian cybersecurity landscape.

In addition to the public dashboard, it integrates a range of services including CERT alerting, report mechanisms for victims and monthly summaries that provide a snapshot of recent ransomware activity.

Beyond the free and open-access website, Ransomfeed offers paid services for companies and cybersecurity professionals. These premium features include the generation of custom dashboards, automated reports and deeper trend analysis to support strategic planning. The team emphasizes client confidentiality and data protections as a core component of their services.

It is also recognized for its active presence in the digital threat intelligence ecosystem, sharing updates through various channels such as LinkedIn, Twitter, Bluesky, Telegram, Reddit and blogs. This multi-platform approach enhances visibility and promotes community engagement.

From a privacy standpoint, the platform is fully compliant with the RGPD. It clearly states that no personal data is collected beyond what is strictly necessary for the site's normal functionality, and that the collected data is never shared with third parties or external countries. The platform also designates a responsible data protection officer for any requests related to user privacy and data management [32].

- ransomlook.

Ransomlook is a free and open-source project maintained by Alexandre Dulaunoy and Fafner, designed to help users monitor and analyse ransomware-related activity across multiple platforms. The project consolidates intelligence from diverse sources such as blogs, forums, Bitcoin wallets, Telegram channels and Twitter feeds.

The platform provides a variety of intuitive interfaces designed for different analytical needs. These includes listings of recent cyberattacks, ransomware group profiles, ransom notes and posting frequency statistics.

A distinguishing feature of Ransomlook is its emphasis on monitoring cyberthreat communication channels and associated cryptocurrency wallets. This is particularly useful for researchers and incident response teams seeking to correlate financial transactions with specific ransomware families or groups [33].

VII CONCLUSIONS AND FUTURE WORKLINES

The development of this project has successfully fulfilled all the established objectives, both at the conceptual and technical levels. Through a modular and iterative design process, a comprehensive and robust system was implemented to collect, process, store, analyse and visualize data related to ransomware attacks published on darknet leak sites

To begin with, a fully automated web scraper was developed, capable of accessing onion websites, interacting with JavaScript-rendered content and extracting relevant information on cyberattacks within the Spanish territory. The integration of Selenium with the TOR network ensured not only the ability to scrape dynamic content but also provided significant anonymity and security through the implemented strategies that mimic human behaviour, meeting the project's second core objective.

The extracted data was processed through a complete ETL pipeline implemented in the Python programming language. This process normalized and cleaned the scraped data, preparing it for insertion into a high-performance NoSQL database such as MongoDB, which was selected due to its flexibility and native compatibility with semi-structured data and the JSON format. As mentioned earlier in this work, database sharding was not implemented due to constraints imposed by the development environment. Nevertheless, a set of proposed guidelines are presented in the corresponding Future Work section.

To facilitate data access and exploration, a RESTful API was implemented using Spring Boot, exposing structured endpoints that serve as a backend interface to the data. On the client side, a React and Typescript frontend was developed, featuring interactive data visualizations and dynamic interfaces powered by D3.js and Bootstrap. The application was designed to be responsive and user-friendly, enabling intuitive navigation and clear insight into ransomware groups profiling and statistics.

Additionally, the entire data pipeline was orchestrated using Apache Airflow, allowing for precise task scheduling, error handling and execution monitoring. The database backup pipeline operates once per month while the scraping process runs once per day.

To ensure platform independence and simplify deployment, all components were containerized using Docker and managed via Docker Compose, contributing to the system's scalability.

A total of seven cyberattacks were successfully extracted from the leak site, as the scraping process was limited to include only cyberattacks that targeted organizations within the Spanish territory.

In conclusion, the project has met all its initial and technical objectives by delivering a fully functional platform for the automated monitoring, processing and analysis of ransomware activity published in the darknet ecosystem. Specifically, this project offers a unique and detailed view of the ransomware ecosystem that targets the Spanish territory.

From a personal standpoint, the development of this project has not only been a technical and academic challenge, but also a rich experience. It revealed how vital proactive monitoring is in detecting such threats. Additionally, this system demonstrates how automation, open-source tools and strategic data enrichment can be combined to produce threat intelligence in near real time.

Through the project, it also became clear that both developers and administrators play a critical role in ensuring the correct data extraction and enrichment over time as leak sites often change domains, platforms or are voluntarily or forcibly shut down. In fact, the site targeted in this project was eventually closed by the cybercriminal group itself, demonstrating the need for agile system maintenance.

The development of a darknet web scraper for cyberattack intelligence and its corresponding statistics portal opens several improvements for future work. Some of the main limitations identified in this work include potential database bottlenecks, the presence of advanced anti-bot techniques on darknet sites, the lack of scalability of the scraping processes, the limited number of leak sites currently being scraped and the limited context data published by the ransomware groups. Therefore, expanding the current system would significantly enhance its utility, accuracy, robustness and contribution to the field of cyber threat intelligence. These challenges are addressed in the following subsections as work lines that can be pursued in subsequent iterations of this project.

- Distribution of the database service

As the system continue to collect cyberattack records on a daily basis, the volume of the stored data will significantly grow. Maintaining a centralized MongoDB instance through such conditions poses risks of performance degradation, limited scalability and potential single points of failure.

To address these limitations, it is proposed that the current database be transitioned to a sharded and distributed MongoDB deployment infrastructure. In this scenario, there are three different types of MongoDB services: the router services which acts as the interface between applications and the sharded cluster, the configuration servers which maintain metadata about the cluster's structure and the distribution of data across shards and the shard servers that store actual subsets of the dataset, replicated across nodes to ensure redundancy and fault tolerance.

A critical challenge in implementing sharding is the selection of an appropriate shard key, especially considering the schema-less and heterogeneous nature of the scraped data. The shard key must offer both high cardinality and a logical distribution pattern to ensure load balancing across shards.

Given these requirements and the schema-less nature of the scraped data, a compound shard key consisting of the cybercriminal group name and the breached organization name becomes a strong candidate. This key not only provides high cardinality but also aligns with the existing data access patterns of the system. In particular, the backend API frequently executes queries filtered by cybercriminal group name, suggesting that this field should not be hashed to improve performance and query efficiency.

By distributing the database in this manner, the system would gain a improved read latency under heavy data loads, high availability through replication and enhanced scalability.

- Scraping of new leak sites

An essential extension involves the incorporation of additional ransomware leak sites into the scraping pipeline in order to provide a broader ransomware group coverage, early cyberattack detection and improved trend analysis.

Integrating additional sources requires scalable adjustments to the scraper module. This includes the implementation of a custom group scraper and ETL processes.

This improvement also concerns the enrichment of the data model with supplementary information for each recorded cyberattack. The current schema captures essential data points, but many relevant attributes remain either uncollected or unstandardized. These points would significantly enhance the analytical depth of the statistics portal.

- Dashboard filters

Implementing dynamic filters in the statistics dashboard represents a valuable enhancement for future development. These filters would enable users to focus on certain data based on specific criteria such as time range, region, sector, target or cybercriminal group. By allowing users to interactively adjust the scope of the data, trends and correlations relevant to specific needs or areas of interest can be uncovered. Additionally, integrating such filters into the frontend requires efficient query handling and backend support to ensure performance and scalability when working with large datasets.

- Overcome advanced anti-bot strategies

Many anti-bot strategies were considered during the development of this project. However, certain presence of advanced anti-bot mechanisms deployed by certain leak sites are still a barrier to data extraction. As of now, the system is not equipped to effectively bypass CAPTCHAs and emerging anti-bot techniques. Future iterations of this project should explore methods to mitigate these countermeasures, with careful consideration of ethical and legal boundaries.

- Implementation of an administrator interface

Aiming the enhancement of operational usability and facilitate the administration of the stored data, a dedicated administrator interface is proposed. This module would enable authorized users to enrich, update or correct context-related information that cannot be automatically extracted through scraping, thereby ensuring greater data completeness.

The backend system should expose a strongly secured endpoints for managing cyberattack records. These endpoints should consist of one HTTP GET endpoint to provide the list of modifiable cyberattacks, one HTTP GET endpoint to provide the already-stored data and one HTTP PUT endpoint to enable cyberattack data inserting and updating.

The latter endpoint should implement validation and sanitization to prevent critical security vulnerabilities such as CSRF, MongoDB query injection and other malicious payloads.

Each update request would pass through an ETL pipeline designed in the backend, ensuring that modified data matches the same normalization rules applied to automatically ingested records.

A corresponding administrator interface shall be developed to interact with the mentioned endpoints. This interface should present a user-friendly and editable view of each cyberattack. To ensure secure access, the interface should implement strong passwords requirements with rate-limited login attempts, two-factor authentication or other strong security measures.

- Alerting system

An alerting mechanism should be integrated to notify administrators of newly scraped cyberattacks, enabling early data review and enrichment. These alerts could be delivered via email, notifications or any other messaging system.

- Support for additional anonymity networks

At present, the system focuses exclusively on the Tor network. However, extending support to other anonymity networks such as *I2P* or *ZeroNet* would enhance coverage and offer a more comprehensive view of cybercriminal activity. Each network presents unique technical challenges in terms of accessibility and scraping, but expanding the scope is essential to track a wider range of threat actors who may not rely solely on TOR-based platforms.

- Channel monitoring

Currently the system relies solely on scraping publicly accessible darknet leak sites. However, ransomware groups use a diverse range of communication channels to distribute victim information, threaten targets and recruit affiliates. These channels include darknet forums, surface web blogs, Telegram channels, Twitter accounts, RSS feeds and even blockchain records such as Bitcoin wallets.

By incorporating channel monitoring, the platform would evolve from a passive data aggregator to an active cyber threat observatory, aligning closely with the operational needs of a CERT or security operations team.

- Development a public API

To facilitate broader interoperability, the implementation of an API is proposed. This API would allow external threat intelligence platforms, research institutions, or security operation centres to query and ingest the data collected by the darknet scraper. It would provide endpoints for accessing real-time statistics, raw threat data, or summarized trend reports.

- Integration with Machine Learning for trend prediction

Future iterations of the system could incorporate machine learning models to analyse historical patterns and predict future trends in cyberattacks. Using techniques such as time series forecasting or natural language processing, the system could anticipate the rise of specific ransomware families, identify novel threat actor behaviours, or predict the geographic spread of cyber incidents. This predictive layer would enhance the decision-making capabilities support proactive cybersecurity strategies.

- Identification of Existing Decryption Tools

Another valuable feature would be the ability to detect and catalogue decryption tools associated with specific ransomware families. This functionality would benefit victims and security professionals by identifying potential means of data recovery without paying ransoms. It would involve scraping forums and repositories where such tools are shared and cross-referencing these tools with known ransomware strains.

- CERT notification

A valuable future enhancement involves the implementation of an automated CERT or CSIRTs notification mechanism, enabling direct communication of newly discovered cyberattacks to relevant authorities and response units. This integration would significantly contribute to proactive incident response and thereby early mitigation.

- Implementation of a Newsletter

Regarding public user experience and early notification, an automated newsletter service could be developed. This feature would allow users to subscribe to periodic updates on new ransomware reports.

APPENDIXES

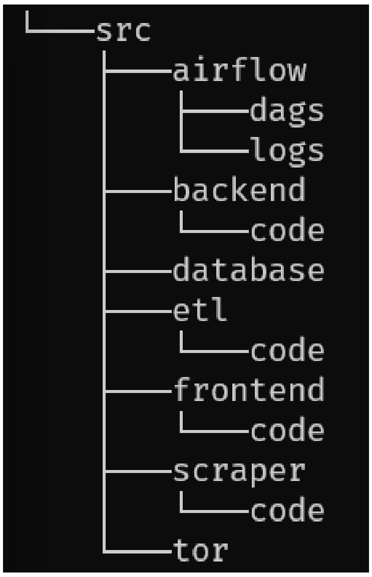
APPENDIX A: TECHNICAL DOCUMENTATION

Introduction

This section outlines the directory structure adopted in the implementation of the project and presents the principal technical considerations necessary for the successful execution of the project. The information provided serve as a practical reference for users seeking to deploy, extend or validate the system.

Directory structure

The directory structure has been designed with modularity, clarity and maintainability in mind. Each major service of the system is organized into its own dedicated directory. Within each one of these folders, the configuration files required for containerization and environment setup are separated from the main application logic, which is gathered into the code folder. An overview is shown in the following figure.



```
└───src
    ├───airflow
    │   ├───dags
    │   └───logs
    ├───backend
    │   └───code
    ├───database
    ├───etl
    │   └───code
    ├───frontend
    │   └───code
    ├───scraper
    │   └───code
    └───tor
```

Figure 17. Directory structure overview.

At the project's root level, a Makefile has been created to accelerate and simplify the most common development and deployment tasks, thereby reducing setup errors and improving efficiency.

Compilation, installation, and execution

To run the project, the following services should be installed in the host computer:

- Docker. The author used version 28.0.4.
- Docker Compose. The author used version 2.34.0.

It is important to note that the variables set in the .env file are specifically set to simplify local development, testing and debugging and shall not be used in production environments as they may introduce serious security vulnerabilities.

The project shall be deployed executing the following steps sequentially:

1. Clone the repository https://github.com/Elmerlusa/TFM.
2. Open a command prompt and move to the project's root folder.
3. Deploy the Docker application using the command 'make', make sure the docker daemon is running.
4. Wait until all the services have successfully started and are marked as healthy.
5. To stop the application the command 'make down' shall be executed.

Given the academic purpose of this work, all the main services are configured to expose their respective ports on the host machine to facilitate local development, testing and demonstration. The services and their access points are detailed below.

- Apache Airflow manager

Accessible at http://localhost:8080, this interface allows users to monitor, manage and manually trigger DAGs. It also provides a user-friendly dashboard to inspect task logs, statuses and execution times.

- MongoDB console

This service is bound to port 27017, the default MongoDB, allowing administrators and developers to connect using both GUI tools and CLI tools for database inspection, querying and debugging.

- Backend API

Hosted on http://localhost:8000, the backend service exposes the RESTful API endpoints described earlier. It facilitates debugging and allows real-time inspection of the data structures returned by the system.

- Web portal

The frontend service is served through http://localhost:3000, offering an accessible interface for end users to visualize the cyberattack data. This route loads the React-based interface that integrates with the backend to display statistical and real-time information.

Exposing ports on the host is useful during development, this configuration is not secure for production environments. In a real deployment, services should be properly isolated, protected by firewalls and accessed only through secure channels such as HTTPS.

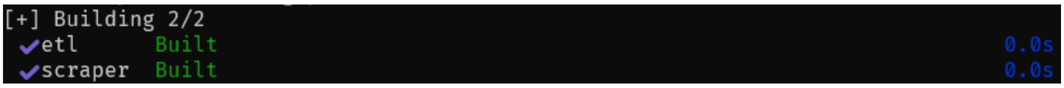
APPENDIX B: EXECUTION RESULTS

Introduction

This appendix is intended to demonstrate the successful deployment and operation of the project. It includes visual and technical evidence confirming that each component, ranging from the data scraping pipeline to the statistics web portal, is functioning as expected.

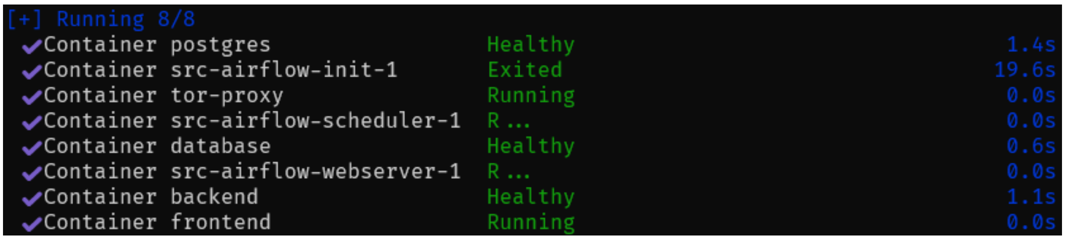
Execution

The correct Docker deployment of this work is shown in the figures below.



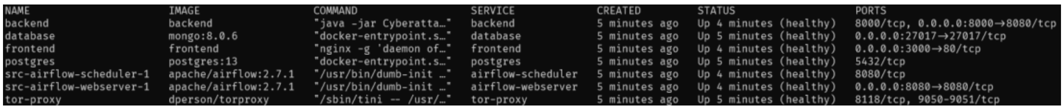
```
[+] Building 2/2
 ✓etl      Built                                                    0.0s
 ✓scraper  Built                                                    0.0s
```

Figure 18. project's building phase.



```
[+] Running 8/8
 ✓Container postgres                    Healthy                     1.4s
 ✓Container src-airflow-init-1          Exited                     19.6s
 ✓Container tor-proxy                   Running                     0.0s
 ✓Container src-airflow-scheduler-1     R ...                       0.0s
 ✓Container database                    Healthy                     0.6s
 ✓Container src-airflow-webserver-1     R ...                       0.0s
 ✓Container backend                     Healthy                     1.1s
 ✓Container frontend                    Running                     0.0s
```

Figure 19. project's running phase.



```
NAME                      IMAGE                   COMMAND                   SERVICE             CREATED         STATUS                    PORTS
backend                   backend                 "java -jar Cyberatta…"    backend             5 minutes ago   Up 4 minutes (healthy)    8000/tcp, 0.0.0.0:8000→8080/tcp
database                  mongo:8.0.6             "docker-entrypoint.s…"    database            5 minutes ago   Up 5 minutes (healthy)    0.0.0.0:27017→27017/tcp
frontend                  frontend                "nginx -g 'daemon of…"    frontend            5 minutes ago   Up 4 minutes (healthy)    0.0.0.0:3000→80/tcp
postgres                  postgres:13             "docker-entrypoint.s…"    postgres            5 minutes ago   Up 5 minutes (healthy)    5432/tcp
src-airflow-scheduler-1   apache/airflow:2.7.1    "/usr/bin/dumb-init …"    airflow-scheduler   5 minutes ago   Up 4 minutes (healthy)    8080/tcp
src-airflow-webserver-1   apache/airflow:2.7.1    "/usr/bin/dumb-init …"    airflow-webserver   5 minutes ago   Up 4 minutes (healthy)    0.0.0.0:8080→8080/tcp
tor-proxy                 dperson/torproxy        "/sbin/tini -- /usr/…"    tor-proxy           5 minutes ago   Up 5 minutes (healthy)    8118/tcp, 9050-9051/tcp
```

Figure 20. Docker running containers.

Results

Once the project is fully deployed and all services are running in a healthy state, both Apache Airflow pipelines were manually triggered via the Airflow UI. The execution of these pipelines is demonstrated in the subsequent sections.

- mongo_backup_dag

The correct execution of this pipeline, which runs monthly, is shown in the following figure.

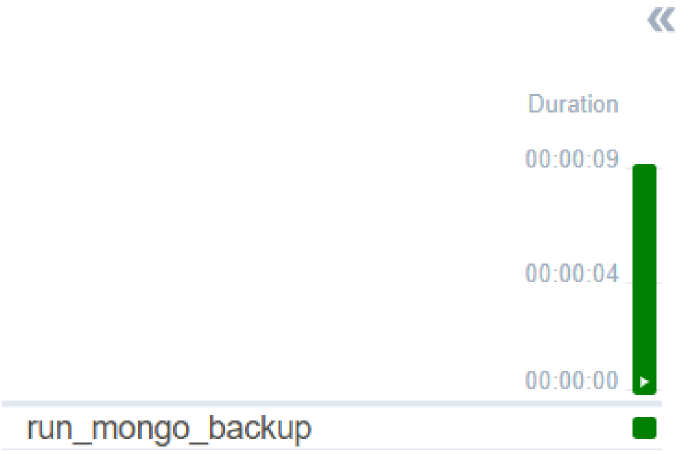


Figure 21. mongo_backup_dag execution.

As a result, a compressed file with the database dump is generated and named with format '<YYYY-MM-DD>.tar.gz'.

- scraper_etl_pipeline

The correct execution of this pipeline, which runs daily at 2 AM, is shown in the following figure.

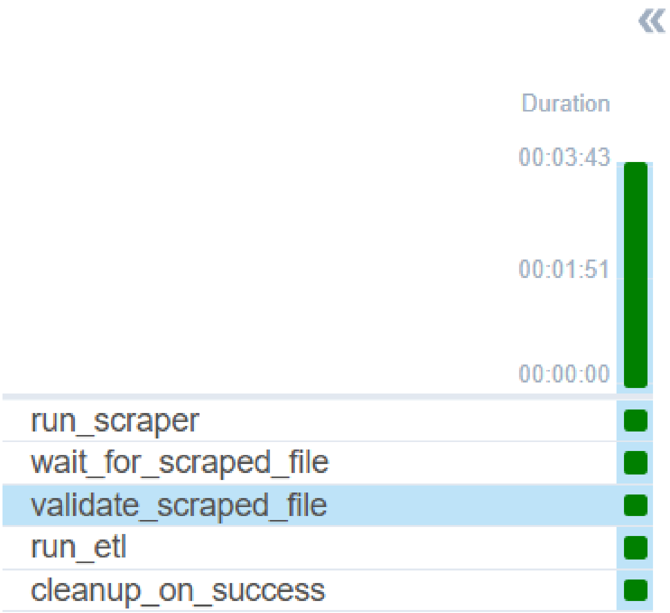


Figure 22. scraper_etl_pipeline execution.

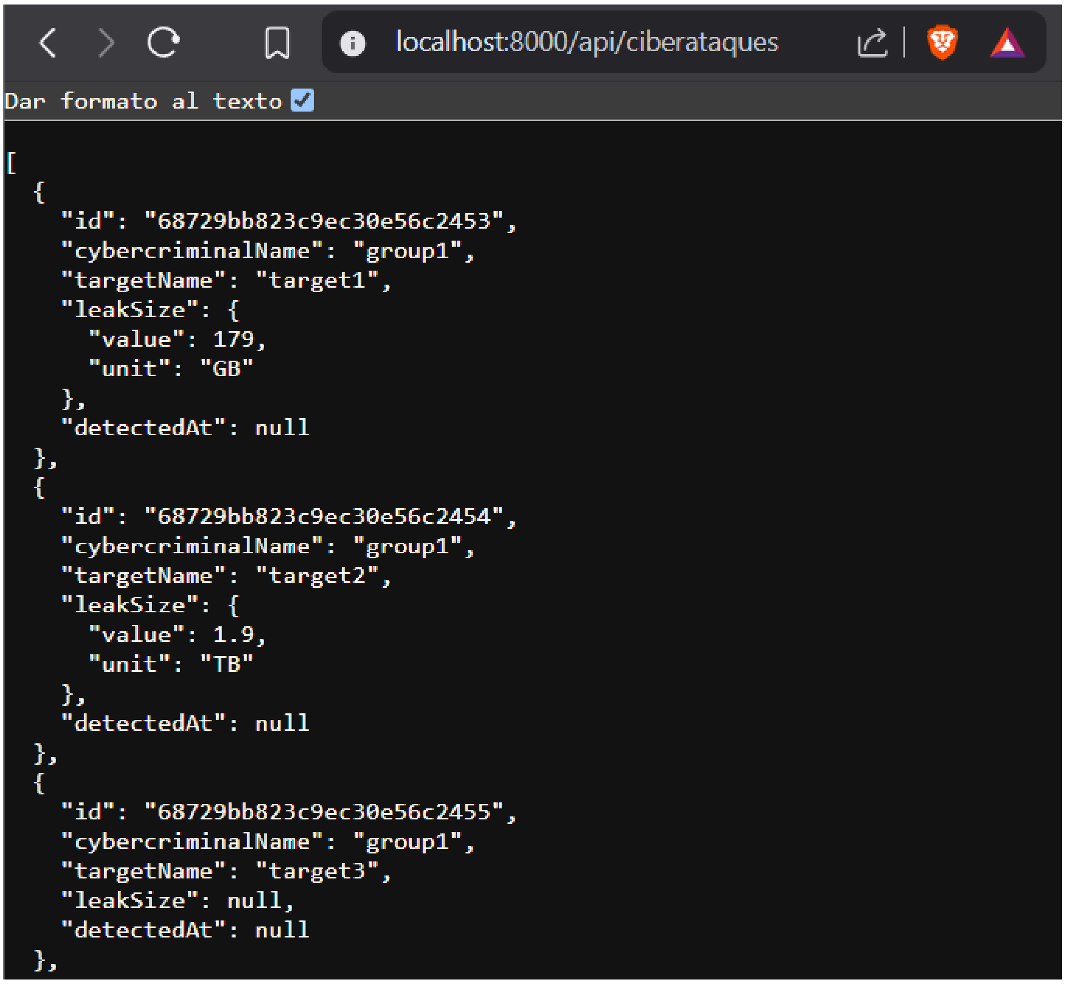
As a result of the scraping process execution, seven cyberattacks were scraped and exported in a JSON file named using the format ‘<cybercriminal>_scraped_data.json’. The file contains the data presented below. Note that entity names have been anonymized to guarantee privacy and ethical compliance.

```
[
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target1",
            "website": "https://www.target1.es/",
            "size": "248",
            "revenue": "$37.9M"
        },
        "disclosures": "1/1",
        "leakSize": "179 GB",
        "leakFiles": "195,086 files",
        "scrapedAt": "2025-07-01T13:13:41.786Z"
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target2",
            "website": "https://www.target2.es/",
            "size": "518",
            "revenue": "$289M"
        },
        "disclosures": "41/82",
        "leakSize": "1.9 TB",
        "leakFiles": "754,871 files",
        "scrapedAt": "2025-07-01T13:13:43.982Z"
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target3",
            "website": "https://www.target3.es/",
            "size": "142",
            "revenue": "$19.1M"
        },
        "scrapedAt": "2025-07-01T13:13:44.211Z"
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target4",
            "website": "https://www.target4.es/",
            "size": "302",
            "revenue": "$87.6M"
        },
        "disclosures": "1/1",
        "leakSize": "233 GB",
        "leakFiles": "554,821 files",
        "scrapedAt": "2025-07-01T13:13:46.362Z"
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target5",
            "website": "https://www.target5.es/",
            "size": "59",
            "revenue": "$29.4M"
        },
        "disclosures": "3/5",
        "leakSize": "12.2 GB",
        "leakFiles": "17,536 files",
        "scrapedAt": "2025-07-01T13:13:48.775Z"
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target6",
            "website": "https://www.target6.es/",
            "size": "148",
            "revenue": "$30.9M"
        },
        "disclosures": "3/5",
        "leakSize": "53 GB",
        "leakFiles": "858,621 files",
        "scrapedAt": "2025-07-01T13:13:49.997Z"
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "target": {
            "name": "target7",
            "website": "https://www.target7.es/"
        },
        "scrapedAt": "2025-07-01T13:13:51.234Z"
    }
]
```

Additionally, the ETL process parsed this JSON file and applied the previously defined data normalization rules. The transformed data was inserted into the MongoDB database and shown below.

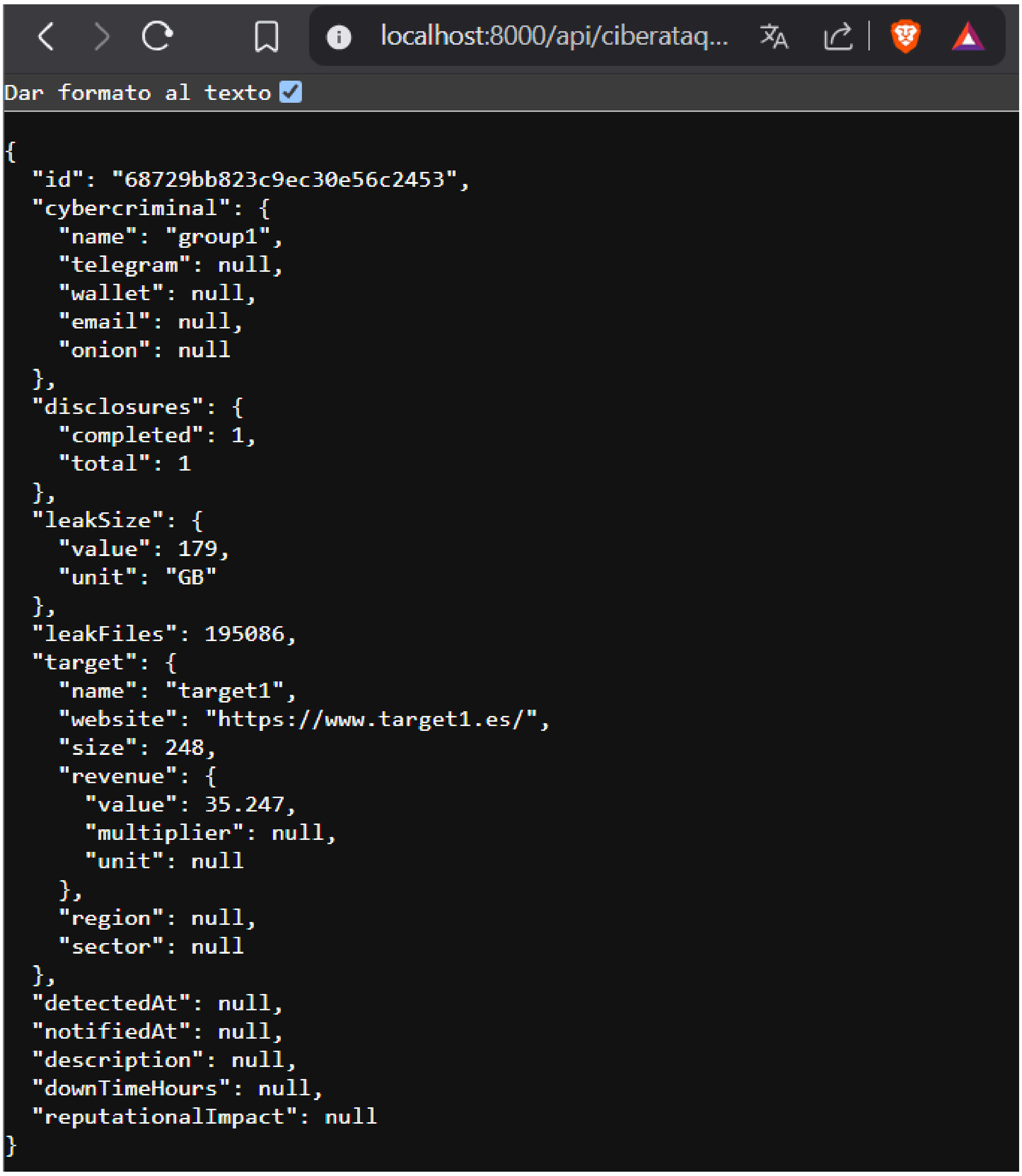
```
[
    {
        "cybercriminal": {
            "name": "group1"
        },
        "disclosures": {
            "completed": 1,
            "total": 1
        },
        "leakSize": {
            "value": 179.0,
            "unit": "GB"
        },
        "leakFiles": 195086,
        "scrapedAt": "2025-07-01T13:13:41.786Z",
        "target": {
            "name": "target1",
            "website": "https://www.target1.es/",
            "size": 248,
            "revenue": {
                "value": 35.247,
                "abbreviation": "M",
                "coin": "€"
            }
        }
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "disclosures": {
            "completed": 41,
            "total": 82
        },
        "leakSize": {
            "value": 1.9,
            "unit": "TB"
        },
        "leakFiles": 754871,
        "scrapedAt": "2025-07-01T13:13:43.982Z",
        "target": {
            "name": "target2",
            "website": "https://www.target2.es/",
            "size": 518,
            "revenue": {
                "value": 268.77000000000004,
                "abbreviation": "M",
                "coin": "€"
            }
        }
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "scrapedAt": "2025-07-01T13:13:44.211Z",
        "target": {
            "name": "target3",
            "website": "https://www.target3.es/",
            "size": 142,
            "revenue": {
                "value": 17.763,
                "abbreviation": "M",
                "coin": "€"
            }
        }
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "disclosures": {
            "completed": 1,
            "total": 1
        },
        "leakSize": {
            "value": 233.0,
            "unit": "GB"
        },
        "leakFiles": 554821,
        "scrapedAt": "2025-07-01T13:13:46.362Z",
        "target": {
            "name": "target4",
            "website": "https://www.target4.es/",
            "size": 302,
            "revenue": {
                "value": 81.468,
                "abbreviation": "M",
                "coin": "€"
            }
        }
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "disclosures": {
            "completed": 3,
            "total": 5
        },
        "leakSize": {
            "value": 12.2,
            "unit": "GB"
        },
        "leakFiles": 17536,
        "scrapedAt": "2025-07-01T13:13:48.775Z",
        "target": {
            "name": "target5",
            "website": "https://www.target5.es/",
            "size": 59,
            "revenue": {
                "value": 27.342,
                "abbreviation": "M",
                "coin": "€"
            }
        }
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "disclosures": {
            "completed": 3,
            "total": 5
        },
        "leakSize": {
            "value": 53.0,
            "unit": "GB"
        },
        "leakFiles": 858621,
        "scrapedAt": "2025-07-01T13:13:49.997Z",
        "target": {
            "name": "target6",
            "website": "https://www.target6.es/",
            "size": 148,
            "revenue": {
                "value": 28.737000000000002,
                "abbreviation": "M",
                "coin": "€"
            }
        }
    },
    {
        "cybercriminal": {
            "name": "group1"
        },
        "scrapedAt": "2025-07-01T13:13:51.234Z",
        "target": {
            "name": "target7",
            "website": "https://www.target7.es/"
        }
    }
]
```

The inserted data was successfully consumed by the backend API, as illustrated in the following figures. This process validates the full integration between the ETL module, the MongoDB database and the backend service.



```
[
  {
    "id": "68729bb823c9ec30e56c2453",
    "cybercriminalName": "group1",
    "targetName": "target1",
    "leakSize": {
      "value": 179,
      "unit": "GB"
    },
    "detectedAt": null
  },
  {
    "id": "68729bb823c9ec30e56c2454",
    "cybercriminalName": "group1",
    "targetName": "target2",
    "leakSize": {
      "value": 1.9,
      "unit": "TB"
    },
    "detectedAt": null
  },
  {
    "id": "68729bb823c9ec30e56c2455",
    "cybercriminalName": "group1",
    "targetName": "target3",
    "leakSize": null,
    "detectedAt": null
  },
```

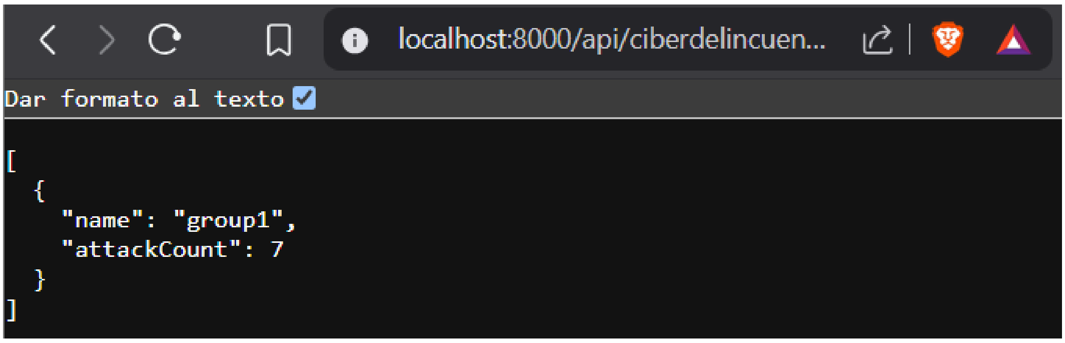
Figure 23. Cyberattacks API endpoint.



Dar formato al texto ☑

```
{
  "id": "68729bb823c9ec30e56c2453",
  "cybercriminal": {
    "name": "group1",
    "telegram": null,
    "wallet": null,
    "email": null,
    "onion": null
  },
  "disclosures": {
    "completed": 1,
    "total": 1
  },
  "leakSize": {
    "value": 179,
    "unit": "GB"
  },
  "leakFiles": 195086,
  "target": {
    "name": "target1",
    "website": "https://www.target1.es/",
    "size": 248,
    "revenue": {
      "value": 35.247,
      "multiplier": null,
      "unit": null
    },
    "region": null,
    "sector": null
  },
  "detectedAt": null,
  "notifiedAt": null,
  "description": null,
  "downTimeHours": null,
  "reputationalImpact": null
}
```

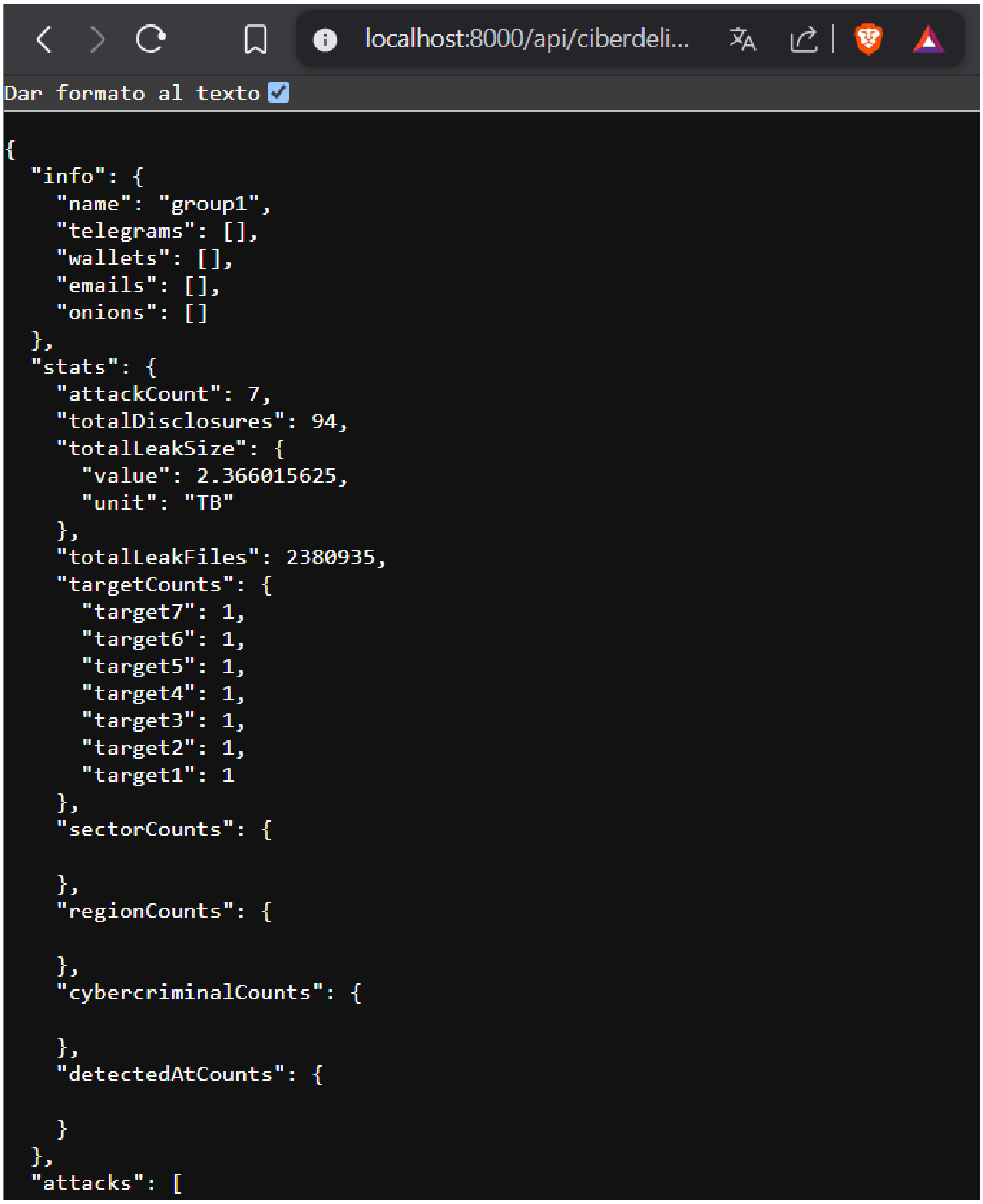
Figure 24. Cyberattack details API endpoint.



Dar formato al texto ☑

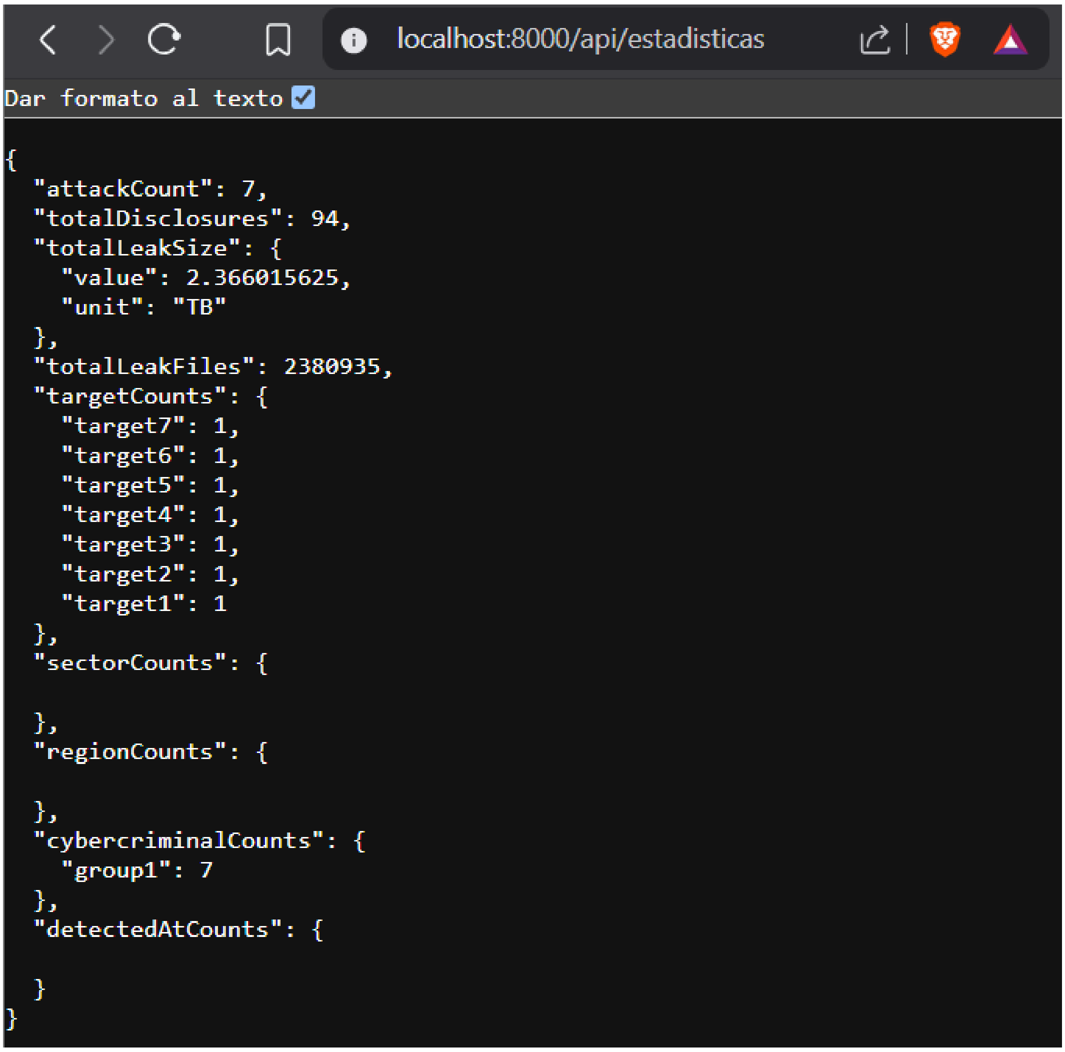
```
[
  {
    "name": "group1",
    "attackCount": 7
  }
]
```

Figure 25. Cybercriminals API endpoint.



```
{
  "info": {
    "name": "group1",
    "telegrams": [],
    "wallets": [],
    "emails": [],
    "onions": []
  },
  "stats": {
    "attackCount": 7,
    "totalDisclosures": 94,
    "totalLeakSize": {
      "value": 2.366015625,
      "unit": "TB"
    },
    "totalLeakFiles": 2380935,
    "targetCounts": {
      "target7": 1,
      "target6": 1,
      "target5": 1,
      "target4": 1,
      "target3": 1,
      "target2": 1,
      "target1": 1
    },
    "sectorCounts": {

    },
    "regionCounts": {

    },
    "cybercriminalCounts": {

    },
    "detectedAtCounts": {

    }
  },
  "attacks": [
```

Figure 26. Cybercriminal details endpoint.



localhost:8000/api/estadisticas

Dar formato al texto ☑

```
{
  "attackCount": 7,
  "totalDisclosures": 94,
  "totalLeakSize": {
    "value": 2.366015625,
    "unit": "TB"
  },
  "totalLeakFiles": 2380935,
  "targetCounts": {
    "target7": 1,
    "target6": 1,
    "target5": 1,
    "target4": 1,
    "target3": 1,
    "target2": 1,
    "target1": 1
  },
  "sectorCounts": {

  },
  "regionCounts": {

  },
  "cybercriminalCounts": {
    "group1": 7
  },
  "detectedAtCounts": {

  }
}
```

Figure 27. Statistics API endpoint.

As demonstrated earlier in this work, the frontend service allows well-structured visualization of the stored data, that shall be enriched to facilitate further analysis. The figures previously displayed were taken from the deployed frontend itself.

BIBLIOGRAPHY

[1] H. Oz, A. Aris, A. Levi and A. Selcuk Uluagac, “A Survey on Ransomware: Evolution, Taxonomy, and Defense Solutions,” *ACM Journals,* 2022.

[2] H. Thanh Luong, “Foundations and trends in the darknet-related criminals in the last 10 years: a systematic literature review and bibliometric analysis,” *Security Journal,* 2024.

[3] F. Teichmann, “Ransomware attacks in the context of generative artificial intelligence - an experimental study,” *International Cybersecurity Law Review,* 2023.

[4] V. Krotov, L. Redd Johnson and L. Silva, “Tutorial: Legality and Ethics of Web Scraping,” *CIO Magazine,* 2020.

[5] IBM, “What Is a NoSQL Database?,” 12 December 2022. [Online]. Available: https://www.ibm.com/es-es/think/topics/nosql-databases. [Accessed 10 June 2025].

[6] IBM, “What is the CAP theorem?,” 20 December 2022. [Online]. Available: https://www.ibm.com/think/topics/cap-theorem. [Accessed 2025 June 10].

[7] IBM, “What are containers?,” 9 May 2024. [Online]. Available: https://www.ibm.com/think/topics/containers. [Accessed 10 June 2025].

[8] GitHub, “GitHub,” [Online]. Available: https://github.com/.

[9] Selenium, “Selenium,” [Online]. Available: https://www.selenium.dev/.

[10] TOR, “The Onion Router,” [Online]. Available: https://www.torproject.org/.

[11] B. Skerritt, “How does Tor actually work?,” 1 March 2019. [Online]. Available: https://hackernoon.com/how-does-tor-really-work-5909b9bd232c. [Accessed 11 June 2025].

[12] Google, “What is ChromeDriver?,” [Online]. Available: https://developer.chrome.com/docs/chromedriver.

[13] Python, “Python,” [Online]. Available: https://www.python.org/.

[14] MongoDB, “MongoDB,” [Online]. Available: https://www.mongodb.com/.

[15] Oracle, “Java,” [Online]. Available: https://www.java.com/.

[16] VMWare, “Spring-Boot,” [Online]. Available: https://spring.io/projects/spring-boot.

[17] Meta, “React,” [Online]. Available: https://react.dev/.

[18] Bootstrap, “Bootstrap,” [Online]. Available: https://getbootstrap.com/.

[19] Microsoft, “Typescript,” [Online]. Available: https://www.typescriptlang.org/.

[20] Observable, “D3,” [Online]. Available: https://d3js.org/.

[21] I. Sysoev, “nginx,” [Online]. Available: https://nginx.org/.

[22] Docker, “Docker,” [Online]. Available: https://www.docker.com/.

[23] Docker, “Docker Compose,” [Online]. Available: https://docs.docker.com/compose/.

[24] S. Brown, “C4 model,” [Online]. Available: https://c4model.com/.

[25] D. Johnson, “Stem,” TorProject, [Online]. Available: https://stem.torproject.org/.

[26] M. van den Berf, “fake-useragent,” Pypi, [Online]. Available: https://pypi.org/project/fake-useragent/.

[27] Python, “json,” python, [Online]. Available: https://docs.python.org/es/3/library/json.html.

[28] MongoDB, “pymongo,” pypi, [Online]. Available: https://pypi.org/project/pymongo/.

[29] Maven, “MVN Repository,” [Online]. Available: https://mvnrepository.com/.

[30] J. Mousqueton. [Online]. Available: https://www.ransomware.live/.

[31] Ransom-DB, "Ransom-DB," [Online]. Available: https://www.ransom-db.com/.

[32] D. Fadda. [Online]. Available: https://ransomfeed.it/.

[33] "RansomLook," [Online]. Available: https://www.ransomlook.io/.