

# RED : A REputation Dashboard that displays the geolocation of bots and malicious computers, utilizing OSINT data.

*Konstantinos Spyridakis*

Thesis submitted in partial fulfillment of the requirements for the  
*Masters' of Science degree in Computer Science and Engineering*

University of Crete  
School of Sciences and Engineering  
Computer Science Department  
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Prof. *Evangelos P. Markatos*

---

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).



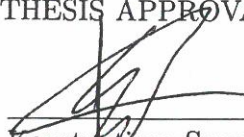
UNIVERSITY OF CRETE  
COMPUTER SCIENCE DEPARTMENT

**RED : A REputation Dashboard that displays the geolocation of bots  
and malicious computers in each country.**


Thesis submitted by  
**Konstantinos Spyridakis**  
in partial fulfillment of the requirements for the  
Masters' of Science degree in Computer Science


THESIS APPROVAL

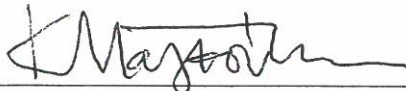
Author:

  
Konstantinos Spyridakis

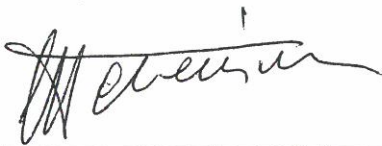
Committee approvals:

  
Evangelos P. Markatos  
Professor, Thesis Supervisor

  
Ioannis Tzitzikas  
Professor, Committee Member

  
Kostas Magoutis  
Associate Professor, Committee Member

Departmental approval:

  
Polyvios Pratikakis  
Associate Professor, Director of Graduate Studies

Heraklion, July 2023



# **RED : A Reputation Dashboard that displays the geolocation of bots and malicious computers, utilizing OSINT data.**

## **Abstract**

OSINT community's active involvement and innovative approaches have played a pivotal role in the evolution of modern systems. Leveraging freely available online sources, developing tools and techniques, have a significant influence on information access, cost-effective solutions and collaboration between individuals and organizations. This work, focused on developing a tool that effectively utilizes OSINT data to extract IP addresses of computers engaged in shady or cybercriminal activities. Adapting to the difficulties of visualizing such content, we introduce an IP Reputation Dashboard (RED) that displays the geolocation of bots and malicious computers across the world. Throughout this research, we explored and implemented methodologies for information acquisition, content analysis, and adept utilization of these approaches to effectively visualize the gathered data on the dashboard. RED's user interface offers interactive visualizations, including world maps, charts, graphs presenting geospatial data pertaining to poor IP address credibility, facilitating quick understanding and interpretation of fetched OSINT information. Furthermore, the design of the tool aligns with the needs of developers seeking to expand or integrate acquisition and visualization techniques into their development environment, providing them with streamlined processes and abstract implementation models. RED empowers users to make data-driven decisions, identify emerging attack trends, track entities, enabling organizations to gain actionable intelligence from the wealth of publicly available information.



## **RED (Reputation Dashboard): Ένας πίνακας ελέγχου αξιοπιστίας που εμφανίζει τη γεωτοποθεσία των ρομπότ και κακόβουλων υπολογιστών, χρησιμοποιώντας δεδομένα από πηγές ανοιχτού λογισμικού.**

### **Περίληψη**

Η ενεργή συμμετοχή της κοινότητας ανοιχτών διαδέσιμων δεδομένων και οι καινοτόμες προσεγγίσεις έχουν καθοριστικό ρόλο στην εξέλιξη των σύγχρονων συστημάτων. Η αξιοποίηση ανοιχτών διαθέσιμων διαδικτυακών πηγών, η ανάπτυξη εργαλείων και τεχνικών, έχουν σημαντική επίδραση στην πρόσβαση σε πληροφορίες, σε οικονομικά αποδοτικές λύσεις και στη συνεργασία μεταξύ ατόμων και οργανισμών. Αυτή η εργασία, επικεντρώθηκε στην ανάπτυξη ενός εργαλείου που χρησιμοποιεί αποτελεσματικά ανοιχτών διαθέσιμα δεδομένα για την εξαγωγή διευθύνσεων IP υπολογιστών που εμπλέκονται σε ύποπτες ή εγκληματικές δραστηριότητες στο διαδίκτυο. Προσαρμοζόμενοι στις δυσκολίες της οπτικοποίησης τέτοιου περιεχομένου, παρουσιάζουμε έναν Πίνακα Φήμης Διευθύνσεων IP που εμφανίζει τη γεωτοποθεσία των ρομπότ και κακόβουλων υπολογιστών σε όλο τον κόσμο. Κατά τη διάρκεια αυτής της έρευνας, διερευνήσαμε και εφαρμόσαμε μεθοδολογίες για την απόκτηση πληροφοριών, την ανάλυση περιεχομένου και την έμπειρη χρήση αυτών των προσεγγίσεων για την αποτελεσματική οπτικοποίηση των δεδομένων που συγκεντρώθηκαν στον πίνακα. Η Διεπαφή Χρήστη του εργαλείου, προσφέρει διαδραστικές οπτικοποιήσεις, συμπεριλαμβανομένων παγκόσμιων χαρτών, γραφημάτων και διαγραμμάτων που παρουσιάζουν γεωχωρικά δεδομένα σχετικά με τη χαμηλή αξιοπιστία των διευθύνσεων IP, διευκολύνοντας τη γρήγορη κατανόηση και ερμηνεία των ανοιχτών διαθέσιμων πληροφοριών που αποκτήθηκαν. Επιπλέον, ο σχεδιασμός του εργαλείου ευθυγραμμίζεται με τις ανάγκες των προγραμματιστών που επιθυμούν να επεκτείνουν ή να ενσωματώσουν τεχνικές απόκτησης και οπτικοποίησης στο περιβάλλον ανάπτυξής τους, παρέχοντάς τους απλοποιημένες διαδικασίες και αφαιρετικά μοντέλα υλοποίησης. Το RED επιτρέπει στους χρήστες να λαμβάνουν αποφάσεις βασισμένες στα δεδομένα, να ανιχνεύουν αναδυόμενες τάσεις επιθέσεων, να παρακολουθούν οντότητες, επιτρέποντας στις οργανώσεις να αποκτήσουν νοημοσύνη από τον πλούτο των δημόσια διαθέσιμων πληροφοριών.





## Acknowledgements

I would like to express my sincere appreciation and gratitude to all those who have contributed to the completion of this research project.

First and foremost, I would like to extend my deepest gratitude to my supervisor Prof. Evangelos Markatos for his guidance, expertise and support throughout both my Bachelor's and Master's studies. He introduced me to the fundamental mechanism of asking simple yet correct questions to address a complex problem in the research field, tearing it down and keeping it simple (KISS). Furthermore, I would like to express my gratitude to my friend and colleague Konstantinos Anemozalis, for his excellent collaboration and irreplaceable contribution to the depth of this work. I am deeply grateful to DiSCS Lab of FORTH, including colleagues and friends that enriched my development experience by discussing and sharing their insights to the breadth of the project.

I am also indebted to FORTH and CC-DRIVER for their financial support, which made this research possible. Their commitment to advancing knowledge and their belief in the importance of research have played a crucial role in the successful completion of this project. I am deeply grateful to my family for their unwavering love, encouragement and support. Being there for any ups and downs during this work, and most important, giving me the ability to attend to University of Crete and complete my Master's Degree. I want to acknowledge my gratitude to Ioanna, for her understanding and belief in my aspirations. Lastly, I want to thank my friends for supporting me in every step.



στους γονείς μου,



# Contents

<b>Table of Contents</b>	<b>2</b>
<b>List of Tables</b>	<b>3</b>
<b>List of Figures</b>	<b>5</b>
<b>1 Introduction</b>	<b>7</b>
1.1 The OSINT Community . . . . .	7
1.2 Research . . . . .	8
1.2.1 Threat Intelligence . . . . .	8
1.2.2 Software development . . . . .	8
<b>2 Threat Intelligence Sources</b>	<b>11</b>
2.1 Overview . . . . .	11
2.2 Datasets . . . . .	11
2.3 Data Source Structure . . . . .	12
2.4 Feed Categorization . . . . .	13
<b>3 System Overview &amp; Architecture</b>	<b>17</b>
3.1 Parsing System . . . . .	18
3.1.1 Macro-Architecture . . . . .	18
3.1.1.1 Controller . . . . .	18
3.1.1.2 Web Parsers . . . . .	19
3.1.1.3 GeoJSON Feature . . . . .	19
3.1.1.4 Extensibility . . . . .	20
3.2 Visualization Dashboard . . . . .	22
3.2.1 Live Dashboard . . . . .	22
3.2.1.1 Dashboard Macro Architecture . . . . .	23
3.2.1.2 Functionality . . . . .	23
3.2.2 Globe Map . . . . .	25
3.2.2.1 Clustering . . . . .	26
3.2.3 User Interface . . . . .	26
3.2.3.1 Maps . . . . .	26
3.2.3.2 Search . . . . .	28

3.2.3.3	Statistics . . . . .	29
3.2.3.4	Threat Exchange . . . . .	30
<b>4</b>	<b>Statistics</b>	<b>35</b>
4.1	Overview . . . . .	35
4.2	Findings . . . . .	35
4.2.1	Attack Distribution per Capita . . . . .	35
4.2.2	IP Occurrence . . . . .	36
4.2.3	Attack Category Distribution . . . . .	37
4.2.4	Attack Origin . . . . .	37
<b>5</b>	<b>Related Work &amp; Limitations</b>	<b>41</b>
5.1	Limitations . . . . .	41
<b>6</b>	<b>Conclusion</b>	<b>43</b>
<b>7</b>	<b>Future Work</b>	<b>45</b>
	<b>Bibliography</b>	<b>49</b>

# List of Tables

1.1	Dataset of collected OSINT blocklists. . . . .	8
2.1	Attack sources categorized. Index column indicates the number of blocklist fetched from each source. . . . .	15





# List of Figures

2.1	Blocklist contribution. . . . .	13
3.1	Architecture Overview. . . . .	18
3.2	AlienVault parser example. . . . .	19
3.3	GeoJSON feature schema . . . . .	20
3.4	Generic parser schema . . . . .	21
3.5	Dashboard macro-architecture . . . . .	22
3.6	Clicking on a cluster. . . . .	27
3.7	Popup example. . . . .	28
3.8	Marker on map. . . . .	29
3.9	Cross validation of IP reputation through alternative reputation services. . . . .	30
3.10	Attack additional information, when clicking a marker. . . . .	31
3.11	Check/Uncheck boxes to show/hide chosen attack categories. . . . .	31
3.12	Inspecting search module. . . . .	32
3.13	Inspecting search results. . . . .	32
3.14	Inspect attack distribution worldwide. . . . .	32
3.15	Inspect attack distribution of a chose country, Greece. . . . .	33
3.16	OSINT data exchange module. . . . .	33
4.1	Choropleth map of attack distribution. . . . .	36
4.2	Attack distribution per capita. . . . .	37
4.3	Attack distribution of the Virgin Islands. . . . .	38
4.4	IP address occurrence. . . . .	39
4.5	Attack category distribution. . . . .	40



# Chapter 1

## Introduction

There has been an increasing interest in open-source intelligence data (OSINT), in various research fields. While more companies, organizations or individuals contribute to the OSINT community, the pool of available data grows. As a result, the need of efficient collection techniques of these data raises. We would like to share some background knowledge and methodology techniques that helps in data acquisition from public available sources. At the same time, data types are complex and differ from one publisher to another, so representation becomes difficult and analysis is impossible especially for non-familiar users. Trying to resolve these issues, led us to implement a two-layer system, collecting and visualizing public datasets including suspicious or malicious IP addresses. This background is provided to help readers understand the importance of the contribution of the OSINT community to mitigate cyberattacks. Furthermore, we provide steps towards building RED system and introduce the architecture design and its expansion capabilities. Acknowledging the impact of this research, will provide readers the ability to comprehend the statistical results, provided in the latest chapters. Finally, we discuss limitations and future work of this study in Chapters 5 and 7, respectively.

### 1.1 The OSINT Community

The Open Source Intelligence (OSINT) community plays a crucial role in today's interconnected world. By harnessing the power of publicly available information, the OSINT community empowers individuals, organizations, and government agencies to gather, analyze, and utilize valuable insights for various purposes. One of the key advantages of OSINT is its accessibility, as it taps into a vast range of publicly available sources such as social media, news articles, public records, and online forums. The OSINT community facilitates the detection of emerging trends, identification of potential risks, and the generation of actionable intelligence. It aids in supporting investigations, monitoring security threats, informing policy decisions, and facilitating disaster response efforts. To this day, the OSINT community faces a major issue, trust. Trust serves as the foundation for the OSINT

Name	Maintainer	List	Category	Date	Description	Result	State
Rescure	<a href="https://rescure.fruxlabs.com">https://rescure.fruxlabs.com</a>	-	-	-	-	FAIL	Outdated
Icewater	<a href="https://github.com/SupportIntelligence/icewater/icewater">https://github.com/SupportIntelligence/icewater/icewater</a>	-	-	-	12,805 Free Yara rules...	FAIL	Outdated
FireHOL IP Lists	<a href="http://iplists.firehol.org">http://iplists.firehol.org</a>	-	various	-	400+ publicly available IP Feeds	PASS	-
Emerging Threats Firewall Rules	<a href="http://iplists.firehol.org/">http://iplists.firehol.org/</a>	-	attacks	-	A collection of rules for several types ...	PASS	-
Bambenek Consulting	<a href="http://osint.bambenekconsulting.com/feed/c2-ipmasterlist.txt">http://osint.bambenekconsulting.com/feed/c2-ipmasterlist.txt</a>	-	c&c	-	A feed of known C&C IP addresses...	PASS	-

Table 1.1: Dataset of collected OSINT blocklists.

community, enabling collaboration, source evaluation, data protection, informed decision-making, and the overall credibility and reliability of the intelligence generated. Hence, researching the trustworthiness of OSINT data is essential to ensure that the information used is reliable, accurate, and obtained from trustworthy sources.

## 1.2 Research

### 1.2.1 Threat Intelligence

In this study, OSINT data were obtained only from publicly-available sources. These sources consist of IP blacklists, corporate threat exchanges (APIs, structured data) and Firewall rules. Determining the trustworthiness of each source required research on its type, publishing rate (consistency) and data usability. This research was a joint work of European project, CC-DRIVER, which was the sponsor and also gave access to sources in the name of research. Each source was maintained and published by a certain group of people. The selection procedure for choosing trustworthy sources started by monitoring their publicly available feed. After a couple of weeks of stable publishing rate, we moved on with evaluation of the overall reputation of the blacklisted IP addresses.

Given the existence of an extensive array of online blacklists with diverse characteristics, evaluation and testing required specific analysis on each one, making this process the most demanding part of our work.

As part of this work, we introduce a dataset holding available sources, information on retrieval and whether are valuable or not. We provide a snippet of this dataset in Figure 1.1. For each evaluated data source we accumulate, among other things, its name, maintainer, category as well as a short description. Finally, we manually review its data source based on the collected information and conclude whether it will be utilized in this work (see column “Result” in Figure 1.1).

### 1.2.2 Software development

The selection of the appropriate OSINT sources, was the first step of the parsing system. In order to perform daily fetching and analysis on those sources, we developed an automated parsing system in charge of checking a source’s availability, fetching non-obsolete IP addresses, categorizing each incident and last, producing structured data compatible to the representation mechanism. Hence, research was conducted to construct a lightweight parsing model built in Python3, distributed

and fault-tolerant, capable of handling web-parsing error scenarios. In this study, the final task was to create a visualization environment for the aforementioned IP sources. Such environment should be friendly and straightforward for any kind of user, cybersecurity aware or not. Thus, we develop a dashboard that maps IP addresses of potential attackers to geographical locations. To this end, finding the appropriate tools to build the dashboard and then displaying it live was the final stage of this work. The client-server model was specifically developed to handle millions of IP addresses in case of high volume sources. More precisely, we used clustering and filtering modules in order to retrieve data faster while not congesting the processing and memory units of the client.



## Chapter 2

# Threat Intelligence Sources

### 2.1 Overview

Datasets from the OSINT community are the main source of information of our system. They come in various formats and provide information such as a description of the attack, network configuration, timestamps, reporter of the incident, attack method, attack timespan etc. Performing data collection revolves around identifying trustworthy organizations or teams that provide accurate and well-maintained datasets. The source evaluation was completed with constructing a multidimensional array containing seen sources and details about them. Fetching OSINT sources led to a collection of blacklists with various formats and metadata. Unfortunately, publishers of blacklists do not share a common publishing format. Furthermore, the mechanisms of data publication differ across organizations. Some observed publishing techniques were through APIs, rules, text files, IP lists and RSS feeds. All blacklisting sources collected for this work can be found on the CC-DRIVER-BACKLIST. The published dataset contains important information such as the distinct name, a short description, purpose of existence or the usage of each blacklist. We chose to accept blacklists that report a fine number of Indicators of Compromise.

### 2.2 Datasets

To find public IP blacklists, we utilized various online services and resources that provide blacklisting information, like MalwareWorld, Maltiverse, VirusTotal etc. Then, we navigated to the origins of those reports. A lot more sources came across while digging in collaborators and related services. We used 34 distinct sources of blocklists for our analysis. These blocklists contain entries called IOCs (Indicators of Compromise), consisting of a unique identifier involved in a cyberattack. Parsing public data based on different APIs can vary in difficulty depending on several factors.

**Documentation:** The availability and quality of API documentation greatly influence the ease of parsing public data.

**API Design:** The design of the API itself affects the parsing process. APIs with clean and consistent data structures, standardized formats (such as JSON or XML), and intuitive naming conventions are generally easier to parse.

**Authentication and Access:** Some APIs, although free of use, require authentication mechanisms, such as API keys, tokens, or OAuth, to access their data.

**Data Complexity:** The complexity of the data itself can influence parsing difficulty.

**Data Volume and Pagination:** If the public data is extensive and paginated, parsing can become more complex. Paginated data requires handling pagination links, managing multiple API requests, and merging or appending data from different pages, which can increase the parsing effort.

**Error Handling:** Robust parsing requires handling various error scenarios, such as rate limiting errors, authentication failures, or malformed data responses. Implementing error handling mechanisms to ensure the reliability and fault tolerance of the parsing process can add complexity.

Overall, the difficulty of parsing public data based on different APIs can vary significantly based on these factors. The need of various parsing schemes, led to the creating of an adaptive parsing model.

## 2.3 Data Source Structure

During this study, we came across several types of public datasets. IOC extraction, for each of those, required a different mechanism. Starting with the simpler and yet most common type of dataset, raw files. Many reporters, in terms of simplicity, publish IOCs by writing them in files and serving them to the public community through their original website. In this work, we collect and process CSV, Text, JSON, XML, Ipsets, Netsets, Rules and TAXII files.

Next, we used API services of organizations, that offered IOCs through specific endpoints. Some examples of such APIs are AlienVault, PulseDive, ThreatFox, Maltiverse. These services offer various IOCs through exposed endpoints. Fetching data from those services requires following guidelines and policies from the original documentation, like token authentication, maximum downloads/period, rate limits.

The Last category of dataset parsing, consists of OSINT tools and projects. Fetching IOCs in this category, requires deployment, run and updating of third-party tools and importing them as modules to our parsing model. FireHOL, Maltrail and OpenCTI offer services through their installation and deployment, responsible for at least 60% of total IOCs parsed.

Please note that each reporter followed a different file structure, closer to their needs, concluding that yet there is not a common format for threat exchanging. This issue is further discussed in Section 5.1.



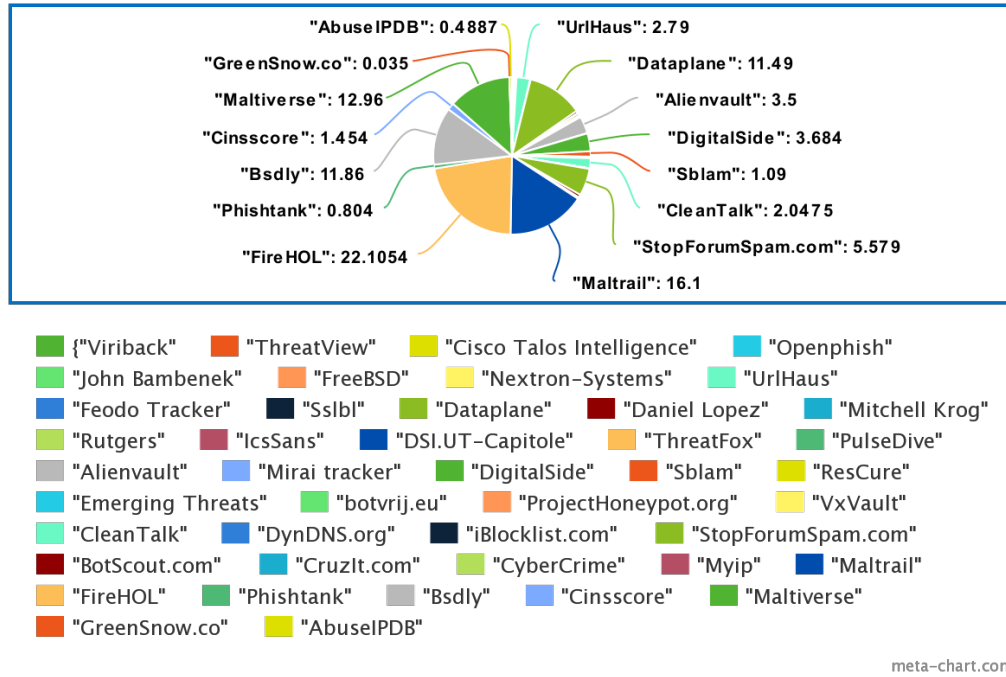


Figure 2.1: Blocklist contribution.

## 2.4 Feed Categorization

Table 2.1 contains the blacklist maintainers classified into four categories based on the attack type they monitor and the number of blacklists they maintain. Our dataset includes popular blacklists such as FireHOL [1], Digitalside [2], Cinsscore [3], Alienvault [4], Abuse.ch [5], Emerging Threats [6].

Figure 2.1 shows the blacklist size distribution in the dataset. On the one hand, we have large blacklists(15.76%) listing more than 500,000 addresses and on the other, we have small blacklists (19.56%) which list less than 1,000 addresses.

**Attack Classification** We classify attacks based on the extra information provided by each maintainer.

- Attacks : General category of attacks (undefined method).
- Abuse : Abusing any mechanism, form of communication (e.g. ports flooding, html form abuse etc.)
- Spam : Abusing communication message protocols.
- Phishing : Targets contracted by email, forms, online calls.
- Malware : Delivery/ Download of infected documents, programs.

- Command and Control : Orchestrator of botnets.
- Bruteforce : Endless connections tries in order to get authentication or slow down the system.

Name	Category	Index
abuseipdb	Attacks	1
alienvault	Varied	1
bambenek	Command&Control	1
benkow	Varied	1
blackhole	Attacks	1
bruteforceblocker	Bruteforce	1
bsdly pop3gropers	Abuse	1
cinsscore	Attacks	1
daniellopez tweetfeed	Varied	1
dataplane	Attacks	1
digitalside	Malware	1
dsi ut capitol	Malware	1
emergingthreats	Command&Control	1
feodotracker	Malware	1
firehole	Varied	36
maltiverse	Malware	3
maltrail	Varied	1
mirai	Malware	1
mittchellkrog	Phishing	1
myip	Attacks	1
nextron	Spam	1
openphish	Phishing	1
phishtank	Phishing	1
pulsedive	Varied	1
rescure	Attacks	1
rutgers	Bruteforce	1
sans	attacks	1
sblam	Spam	1
sslbl	Command&Control	1
talosint	Attacks	1
threatfox	Varied	2
threatview	Attacks	1
urlhaus	Varied	1
viriback	Malware	1

Table 2.1: Attack sources categorized. Index column indicates the number of blocklist fetched from each source.



## Chapter 3

# System Overview & Architecture

Our objective is to provide knowledge and insights about the newest cyberattacks. RED is designed to empower people at storing, visualizing and analyzing cyberattacks.

The primary purpose of RED is to simplify the term “Source IP address” of the cyberattacker. To make this possible, we map IP addresses fetched by the Parsing System to geographic locations. Next, we proceed with the visualization development. Using Leaflet maps, we created an interactive visualization environment for all types of users visiting the platform. Navigating through the map, helps users identify the source of the cyber-attacks. The map is enriched with pinned markers (geolocation points) which provide additional information of the attack incident. The description of each attack is available with a simple click on the pinned marker. Additionally, we utilize several APIs from distinguished organizations to further validate the results of this study. Furthermore, the dashboard offers a search mechanism in order to search for specific IP addresses and check involvement in cybersecurity incidents. Last, we provide cyberattack statistics for each country for a 2-month period. We go through the system’s architecture, analyze the dashboard mechanisms in the Section 3.2. We close this chapter with a short demo, a basic scenario of using this dashboard.

The second goal of the project, is the usability of the parsers that perform IP address collection from OSINT resources. These mechanisms are suggested mostly for developers, in order to deploy this parsing system and enrich it with future available sources. Going through the parsing system’s model, we discuss the importance of fault tolerance regarding web parsers. Another important consideration, during the parsing system development, was the distribution of the downloaded content. We designed this system in Python3 to simplify the deployment without the need of any specific hardware. At last, there is available functionality using configuration files. A new developer, without any Python knowledge, can enable, disable or add a source to the parsing system by editing text configuration

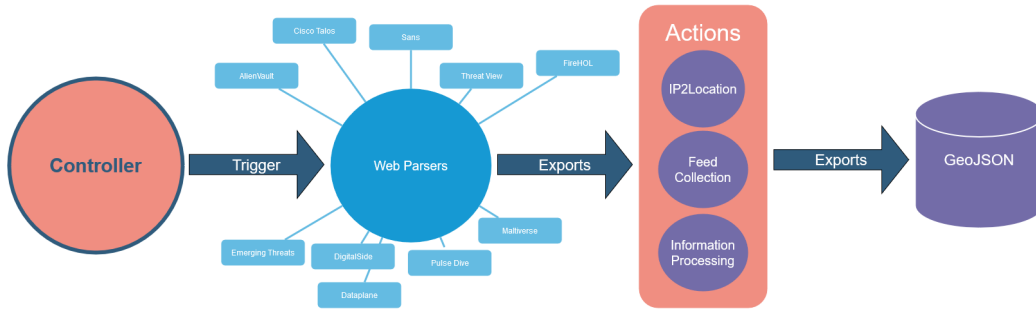


Figure 3.1: Architecture Overview.

files, making the parsing process even easier.

### 3.1 Parsing System

An automated web parsing model based on Python scripting, using OSINT sources with extensibility standards. This system is responsible for collecting open-source intelligence and other available data of computers engaged in shady or cybercriminal activities. After processing the data, it produces a list of verified IP addresses possibly belonging to cybercriminals, with additional information attached to each case.

#### 3.1.1 Macro-Architecture

In this section, we tear down the parsing system to identify its components and the relationships between them. The structure of the system mainly depends on the communication between these components. Isolating and distributing individual tasks on each component enables the system to distribute the load, stabilize easier after a potential failure, and as a result provide extensibility standards for future developers.

##### 3.1.1.1 Controller

As illustrated in Figure 3.1, the parsing system model consists of four main entities. In a multimodular system, orchestration is always a good solution. Starting of, we introduce the Controller. Controller is responsible for adjusting system initialization given the appropriate configuration guidelines. The main purpose of the Controller, is to trigger the fetching process of our second entity, the Web Parsers. Controller handles Web Parsers as standalone processes that depend on the configuration that it provides. In this way, Web Parsers run in parallel, independent of one another. Achieving asynchronous fetching is the key for both distributing downloaded content and stabilizing the system after a potential error scenario. Web parsing error scenarios are common, while online services face downtimes,

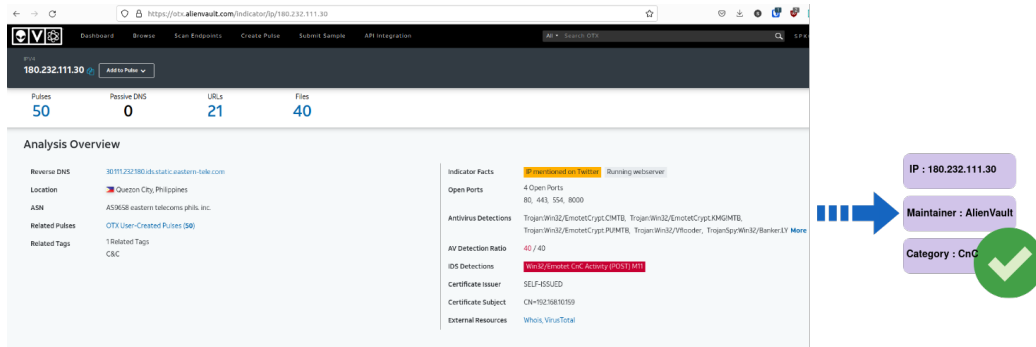


Figure 3.2: AlienVault parser example.

updates etc. The system remains stable after a Web Parser crash and is able to recover with just killing the Web Parser's process and restarting again based on the configuration policies.

### 3.1.1.2 Web Parsers

Moving on to the second main entity, we have the Web Parsers. The term entity is used to describe a group of Python scripts, each of which is built to fetch a specific type of OSINT source. In this study, we managed to successfully fetch thirty-four (34) different sources for over a 3-month period. Every Web Parser is a script, which given a specific configuration will fetch OSINT data. Each one of them are implemented based on how data is structured and delivered. Parsers utilize links, APIs or exposed documents to retrieve data from the Web, using the fetch function. Controller is importing the parsers as Python modules, makes a different call of their fetch function, and finally logs the outcome of the process. The fetching functionality is achieved by analyzing the data downloaded. Downloading published content in any form from AlienVault parser as shown in Figure 3.2. The first step is the verification of the IOCs, by validating the IPv4 and IPV6 format. IP addresses that pass the validation are queried in the IP2Location local database to check their existence and return the geolocation translation. The next step is the creating of the GeoJSON entry. GeoJSON analysis will be on the next section. At last, each parser merges Entries that contain same IP addresses and writes the entries in a Dictionary formatted file using IP addresses as keys. In that way, we speed up the process of merging all lists fetched with duplicate IP addresses.

### 3.1.1.3 GeoJSON Feature

When creating a visualization map, we must consider the amount of input given. We simply cannot overload the map with thousands of pinned locations. Using Supercluster, we can make clusters of locations extremely fast. Supercluster is designed to receive GeoJSON objects as input. So, to prevent map and hence

```

"type": "object",
"properties": {
  "location": {"type": "string"},
  "country": {"type": "string"},
  "ip": {"type": "string"},
  "maintainerURL": {"type": "string"},
  "maintainer": {"type": "string"},
  "category": {"type": "string"},
  "description": {"type": "string"},
  "source": {"type": "string"}
},
"geometry": {
  "type": "Point",
  "coordinates": [
    "number",
    "number"
  ]
}

```

Figure 3.3: GeoJSON feature schema

client overloading, we utilized GeoJSON features as shown in Figure 3.3. GeoJSON features are JSON objects that have an additional field called geometry that contains a geolocation point (latitude, longitude) of a specific location on a map. Supercluster module constructs aggregated points or clusters. Further discussion about Supercluster module in the next chapter. Spectating this object, we can see all the necessary information that needs to be sent to the Dashboard.

#### 3.1.1.4 Extensibility

The end of this chapter is a discussion regarding the need of extensibility. As cyber awareness raises, the cyberattack community becomes larger. As a result, more organizations will publish potential threats to prevent cyberattacks. That being said, increasing the number of the IP Blacklists improves the functionality of tools such as RED. We designed RED web parsers to be easy readable, so that the deployment of a new parser would be smoother. By having more OSINT sources, improve cross-verification, data integrity and in the end provides a richer dataset that demonstrate the current cyberattack behavior closest to reality.

The simpler the implementation, the easier it is for the developer to construct an additional parser. In this example, we spectate the BruteforceBlocker parser that is only 27 lines of beginner level Python3.

```
features={}
```



```
{
    "url": "string",
    "type" : "string",
    "delim" : "string",
    "ip_pos" : "number",
    "category": "string",
    "maintainer": "string",
    "maintainerURL": "string",
    "description": "string",
    "comments": "string"
}
```

Figure 3.4: Generic parser schema

```
__url__ = 'https://danger.rulez.sk/projects/
    bruteforceblocker/blist.php'
__comments__ = []
__maintainter__ = "FreeBSD"
__maintainter_url__ = "https://www.freebsd.org/"
__category__ = "bruteforce"
__description__ = "Ip_address_listed_performed_SSH_
    bruteforce_attack(s)."

def fetch():
    __content__ = csv.reader(fetch_content(__url__, [])
        [0],delimiter=' ')
    for record in __content__:
        if not record: continue
        record = ip2info(ioc_xtr(record[0]))
        if not record :continue
        obj = Entry.createEntry(record,
            __maintainter_url__,__maintainter__,
            __category__,__description__,__url__)
        features[obj.properties.get("ip")] =
            add_or_update(obj,features)
    write2file(features)
    return len(features)
```

Assuming the same published content format, changing the field `__url__` and the rest of the information based on each source, we can create a brand new functional parser. Finally, we introduce a generic web parser, that functions based on a configuration file. So future devs, would not mess up with Python at all. As shown in Figure 3.4, a developer can enhance the functionality of RED, by

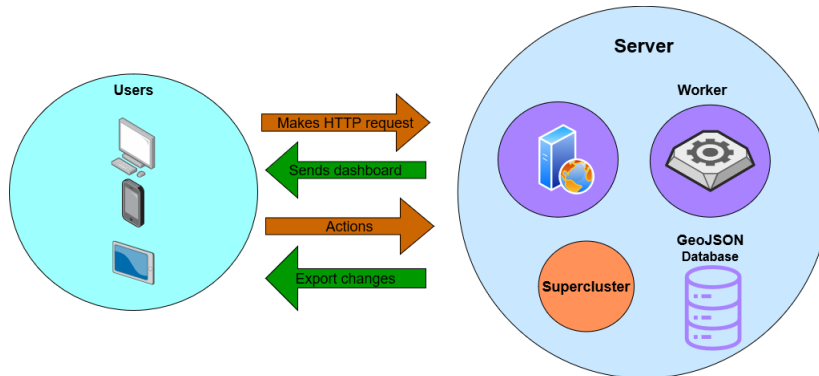


Figure 3.5: Dashboard macro-architecture

complete the required fields. Note that, the content's file type, file structure and data position must be given by the developer. Configuration fields should be completed as in the given example below.

- url : The url of the publishing OSINT document.
- type : Preferred file type
- delim : Delimeter to split objects.
- ip\_pos : Position of the IOC, IP address in our case.
- category : Pre-defined categorization of the incident or position of the desired word in content.
- maintainer : Name of the maintainer, organization, team.
- maintainerURL : Official home website of the maintainer.
- description : Pre-defined description or position of the desired word in content.
- comments : Additional description or comments.

## 3.2 Visualization Dashboard

### 3.2.1 Live Dashboard

In today's world, cyber criminality impact is way underrated. In the effort of raising awareness about cyberattacks, we developed a dashboard to help experts and not experts comprehend the amount of attack incidents worldwide.

### 3.2.1.1 Dashboard Macro Architecture

Initially, when designing a multifunctional dashboard, the first goal to be achieved is the simplicity. Keeping the architecture design simple, is achieved by keeping the implementation and communication of the components separate. In this way, the update, debug and extensibility processes become more stable and fast. As shown in Figure 3.5 on server side, we create a server-worker model. The idea behind this implementation is to give clients access to the dashboard through their browser. So, the first step is to set up a Node.js application, which serves and displays OSINT sources fetch from the web. Eventually, this application service can be exposed to any networks and provide dashboard utilities to any user having access to it.

### 3.2.1.2 Functionality

Now, we will discuss how components work with each other, which is responsible for each task and explanation of our choice not going with alternative software. Visualizing large datasets requires usage of mechanisms specifically designed to handle the load. Our system fetches ninety thousand unique IP addresses per day, so the load is much larger before removing duplicates. In extreme scenarios, e.g. DDoS attacks, cyberattacks number observed raises exponentially, highlighting the need of appropriate mechanisms of handling unexpected loads of data. That being said, we decided to use Supercluster, a very fast open-source JavaScript library for geospatial point clustering for browsers and Node.js. Having Supercluster library as the key component of smooth and fast visualization, we proceeded in developing the application on top of Node.js framework. For deeper understanding on how it works, you can visit IP-Reputation [7].

The idea of client-server communication resides on a simple message communication protocol, Socket.io. Socket.io library enables low-latency, bidirectional and event-based communication between a client and a server. The communication methods that are implemented, are based on actions triggered by the user. User's input is basic usage of a map (navigation, zoom, clicks, popups, filtering) and querying IP addresses. Input type must be handled as events, triggering specific mechanisms on the backend but at the same time bidirectional calls are important so that backend can make changes to user's visualization e.g. attack filters.

**Server.** The server is a Node.js application responsible to provide event logging, user authentication and backend support for the client web application. Events are logged locally on the server. Server is responsible for website availability and binding functionality between live dashboard and pre-fetched data. User authentication is required for providing security in case of exposing the dashboard in the network. In other cases, it is optional and can be bypassed. Server component is responsible for backend orchestration mechanisms such as map bounds changing, zooming, clustering, filtering and IP address querying.

- Map bounds change : Navigation and clicking.
- Zooming : Zoom in or out.
- Clustering : Given new map bounds, zoom to recreate pinned locations clusters.
- Filtering : Add or remove attack type filters, remove pins with the required category.
- IPQuery : Search database for specific IP address CIDR notation.
- getChildren : Return pinned locations of a specific cluster.
- Update : Update map with latest fetched data.

Furthermore, deploying and running the application should be an easy task. Automating these processes requires setting various parameters like restart policies, max heap and scheduled start, stop or restarts. Requirements: Running this tool optimally requires, Python3+, Node.js, make, git unzip and pm2. It was developed and tested on Ubuntu 20.04+ and Debian 10 Buster+. To achieve application management, we used a production process manager for Node.js, PM2. Configuring an PM2 ecosystem is pretty easy, as shown in the example below.

An ecosystem configuration example. File ecosystem.config.js

```
module.exports = {
  apps : [{
    name : "cc-driver",
    script : "./server.js",
    max_old_space_size : "4096",
    cron\_restart : "30 14 * * *"
  }]
}
```

Start the application : Run the ecosystem configuration.

```
pm2 start ecosystem.config.js
```

Deployment of this application is also feasible in Docker environment. Dividing the two-layer application, we distributed the load of the application to two different instances. Regarding the parsing system, we used Python3.9 image to handle the collecting procedures. This image, is scheduled by default to wake up and run once a day, in order to fetch OSINT data. After completing its purpose, the image terminates, resulting in freeing memory and space that were consumed. The parsing system extracts data fetched to a shared volume, in order to be used by the visualization system. The visualization system, is deployed on a Node:18 docker image, that serves the content of our server on default port :8080. When the visualization system successfully runs, content is already served and can be viewed by visiting localhost:8080 with a common browser.

Listing 3.1: Docker-composer configuration.

```

services:
  web-parser:
    build:
      context: .
      dockerfile: parser.Dockerfile
    volumes:
      - shared-volume:/usr/src/app
  web-server:
    build:
      context: .
      dockerfile: server.Dockerfile
    ports:
      - "8080:8080"
    volumes:
      - shared-volume:/usr/src/app
volumes:
  shared-volume:

```

In order to set up and deploy those images, we use Docker-composer. Docker composer's configuration can be found and edited in the application and looks like this Figure 3.1. Distributing the load of the application to different images, improve efficiency and fault tolerance, while keeping each environment sandboxed for users who might choose one of the two functionalities.

**Worker.** Worker is a JavaScript script controlled by the Server and whose primary role is event handling. Starting points are loading feeds on dashboard, data filtering and both Supercluster [8] initialization, implementation. After parsing is complete, the worker parses the GeoJSON file, excludes IP addresses that were found on old datasets and creates a new dataset of unique feeds. This procedure is required in order to distinguish and finally visualize IP addresses which were reported for malicious intent in their geospatial characteristics, reducing loading latency and improving the dashboard's view. Worker module is implementing the server's basic functionality for each client that accesses the RE.Dashboard.

### 3.2.2 Globe Map

This study consists of open-source tools and plugins exclusively. Considering the map implementation, we searched for interactive maps compatible with [8] and JavaScript frameworks. Leaflet is a lightweight library that interacts perfectly with the existing components.

### 3.2.2.1 Clustering

Clustering is the last component of the backend that we will discuss. The actual need of clustering arise from the weakness of multiple pins to be shown on the exact same location. For any case of more than two pins on the same location, only one pins is shown, and the rest are hidden behind. A way of visualization is to construct clusters based on some common characteristics. In our case, clusters are constructed based on the distance between the pinned locations on the map. Clustering thousands of geolocation points on the map requires great CPU power and is not supported by all frameworks. For our study, we imported Supercluster, an open-source library which to RE.Dashboard like a glove.

### 3.2.3 User Interface

When creating a dashboard for multi-type users, the primary goal is to keep it simple. Users feel comfortable using tools which concept is easy to deliver. We designed our tool so that users can immediately point out its basic functionality and purpose of creation. The dashboard's basic functionality is separated with four tabs. The first two are two maps visualizing suspicious IP addresses worldwide. The third tab is a Search engine, that the user is able to query any IP address or IP address range. In the last tab, we display the RED statistics that are collected from April 2023'. In the next sections, we will dive in each tab, analyze its content and discuss the actions a user can perform using RED.

#### 3.2.3.1 Maps

The concrete output of this research malicious activities, is a live dashboard, or more specifically a global map. A map instance of all suspicious activity provides informative statistics and knowledge for each continent, country or city around the globe. The IP-Reputation dashboard shows all IPs that parsers fetch each day. The dashboard is divided into two tabs, "All" and "Today". Dashboard's Actions :

**Cluster** Clusters expand by clicking on them or zooming in. In that way, when reaching a specific zoom level, the cluster will expand and reveal a list of entries, as shown in the Figure 3.6. Clicking on a specific IP address open a new popup containing information for this specific entry.

**Pin** A pin is the visualization technique of viewing a suspicious target on our map. The IP2Location [9] module is responsible for translating the IP address to geolocation attributes. Furthermore, IP2Location [9] provides more information such as Country, City, ASN of the specific entry. A pin contains information about the reporter, the category, and a short description of the suspicious IP address.

Pins as shown in Figure 3.8 are locked geographic locations, enriched with custom popups, that provide extra information about the attack incident. Popups

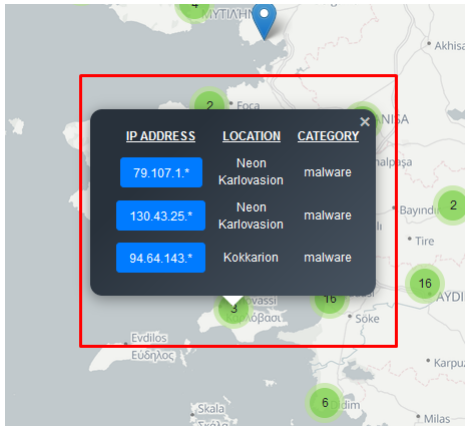


Figure 3.6: Clicking on a cluster.

contain six fields of information. As shown in the Figure 3.7, the suspicious IP address has the following information : Location : Tympakion It is reported by an organization named cite Cinsscore. We categorized this incident as a generic attack. Cinsscore team flagged this packet as Rogue, as it raised a number of alerts when passed through Sentinels. The last field is a button "More information" as shown in Figure 3.9, that contains eight certified organizations that perform IP address reputation scoring. Hitting any of those links provide validation for this IP address's reputation score and in general network behavior.

As shown in the Figure 3.10, the IP address's last byte covered by an asterisk. Under the General Data Protection Regulation (GDPR), the protection of personal data is given high priority. An IP address can be considered personal data if it can be associated with an identifiable individual. The GDPR applies to the processing of such personal data, including the collection, storage, use, and disclosure of IP addresses. The categorization of each malicious activity derives from description, keywords or activity patterns of each attack.

**Attack type filter** Attack type filtering is a module for viewing specific attack types on the dashboard. This mechanism is triggered by (un)checking the top left checkboxes on the map. Each time a preference is changed, the server responds with new filtered data based on attack category. Having 7 different categories of attack types results in many combinations of the original dataset. As a result, every new combination would trigger full recreation of Supercluster [8] object, which is time and memory inefficient. For that reason, we created a data structure to store the up to eight different combinations in order to avoid latency. A user, viewing the dashboard, comes to the realization of two different maps.

1. "All" tab displays all IP addresses and metadata obtained each day from all parsers. Content downloaded is displayed on the globe map without any further filtering.

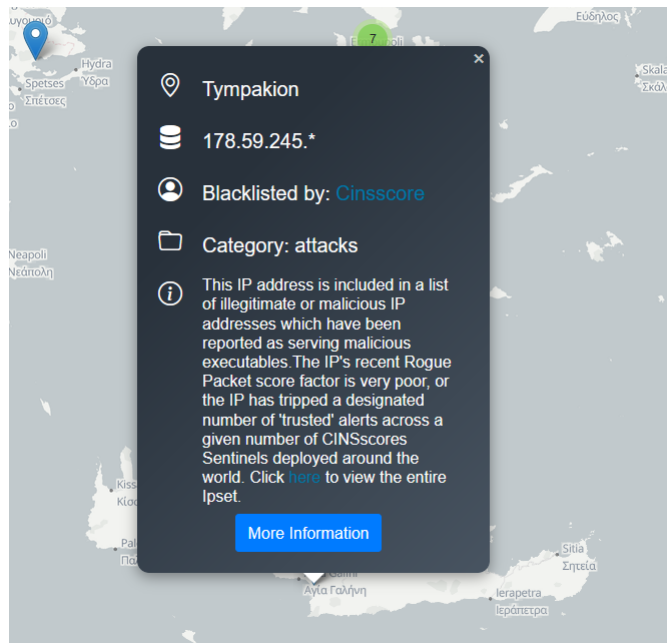


Figure 3.7: Popup example.

2. "Today" tab displays all IP addresses obtained for a specific day, except those found on the previous day's dataset. Using this kind of filtering, we manage to acquire IP addresses that most probably were discovered the same day that were added to the dataset.

This modification of the dashboard's contents was firstly introduced to filter poorly maintained blacklists. Furthermore, it applies maintenance policies for IP-Reputation in regard to the time that an entry (IP address and metadata) should "live" in a dataset. IP-Reputation applies this policy due to the fact that every other blacklist has a different policy of maintaining IP addresses. It is essential, when developing a tool of different sources, that the same policy applies to all the data fetch. In other words, blacklists are not stable at all. Keeping the previous state of each blacklist and compare with the current, helps to remove false positives, error cases and poor maintenance techniques.

### 3.2.3.2 Search

Under the search tab of the tool, users can run queries on specific IP addresses or subnets by using the CIDR notation and receive a list of malicious IP addresses that matches their requirements, as shown in Figure 3.12. Since some queries can result in vast amounts of data, making it difficult to display on a single screen, the data are broken down into pages. Users can also click on the "More Information" button next to each IP address, which would open a popup similar to the ones on the global map, Figure 3.13.



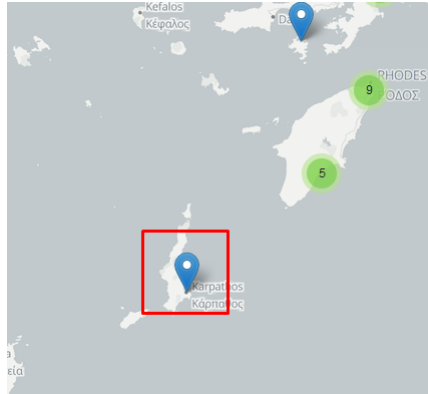


Figure 3.8: Marker on map.

### 3.2.3.3 Statistics

Lastly, we look at the tab named Statistics, viewing Figure 3.14. Statistics is powerful knowledge, while users, especially in our study, can keep track of incidents in cases of enormous datasets. During our work, we managed to build, a lightweight implementation of displaying attack trends over time. Under the statistics tab, we can look at a time series chart, demonstrating the frequency or intensity of attacks over a specific time period. This can reveal patterns, spikes, or trends in cyberattack activities on all countries. The user can view statistics of each country selecting from this bar, as shown in the Figure 3.15.

As shown in Figure 3.14, the time series chart displays a consistent average of 50,000 attacks per day, with a lower bound of 40,000 attacks per day. This suggests a relatively stable baseline level of attacks throughout the observed period.

However, the chart also highlights seven notable spikes where the number of attacks exceeded 80,000 per day. These spikes indicate significant deviations from the average and signify periods of heightened attack activity.

The distribution of attacks appears to be skewed towards these spikes, as they represent major outliers in comparison to the average and lower bound. These spikes may be indicative of specific events or vulnerabilities that attracted a higher number of attacks during those periods.

Analyzing the causes and patterns behind these spikes would be valuable in understanding the factors driving the increased attack activity during those times. It may also be important to investigate whether these spikes correlate with any specific external factors, such as major events or changes in the security landscape, to gain further insights.

Overall, the chart suggests a consistent level of attacks around the average, but with sporadic spikes indicating distinct periods of intensified attack activity. In the next chapter, we will extend the discussion on statistics analyzing plots, derived from a 4-month research.

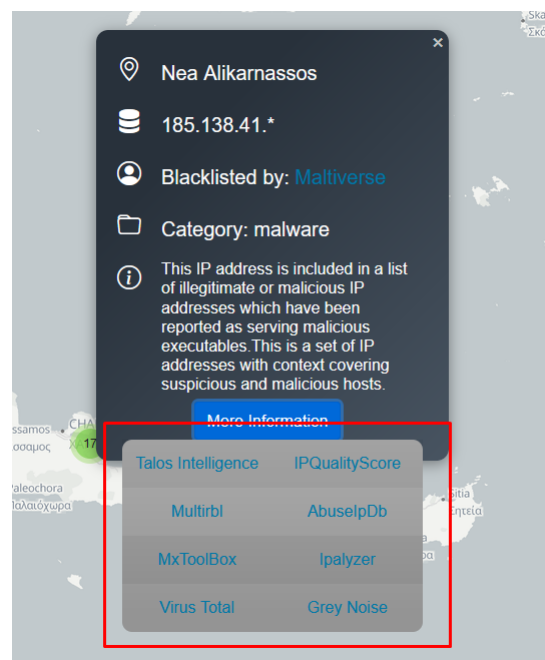


Figure 3.9: Cross validation of IP reputation through alternative reputation services.

#### 3.2.3.4 Threat Exchange

Finally, contributing back to the OSINT community, we expose high threat level IP addresses. The verification and filtering process is done by querying the IP addresses, that found to be in most blacklists, to the Google Safe Browsing API. A user view these IP addresses by running the application and navigating to [http://localhost:8080/malicious\\_ips.txt](http://localhost:8080/malicious_ips.txt). These hosts, hence IP addresses, are considered malicious by a large part of the community. Document format in Figure 3.16

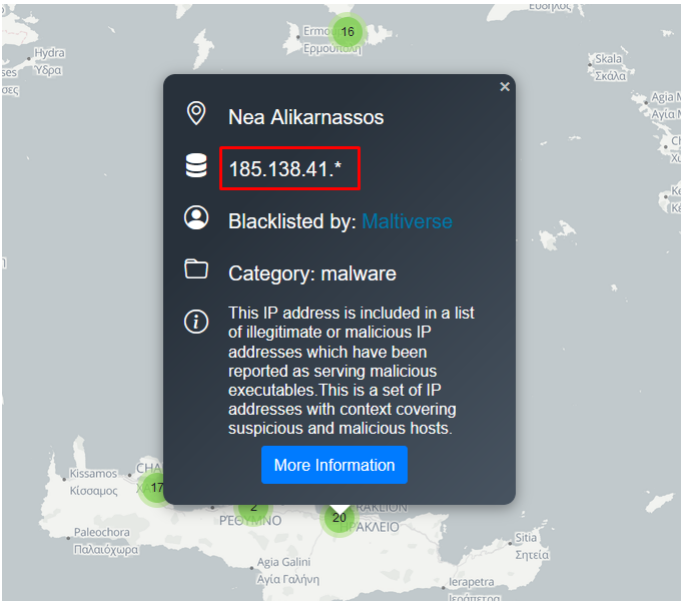


Figure 3.10: Attack additional information, when clicking a marker.

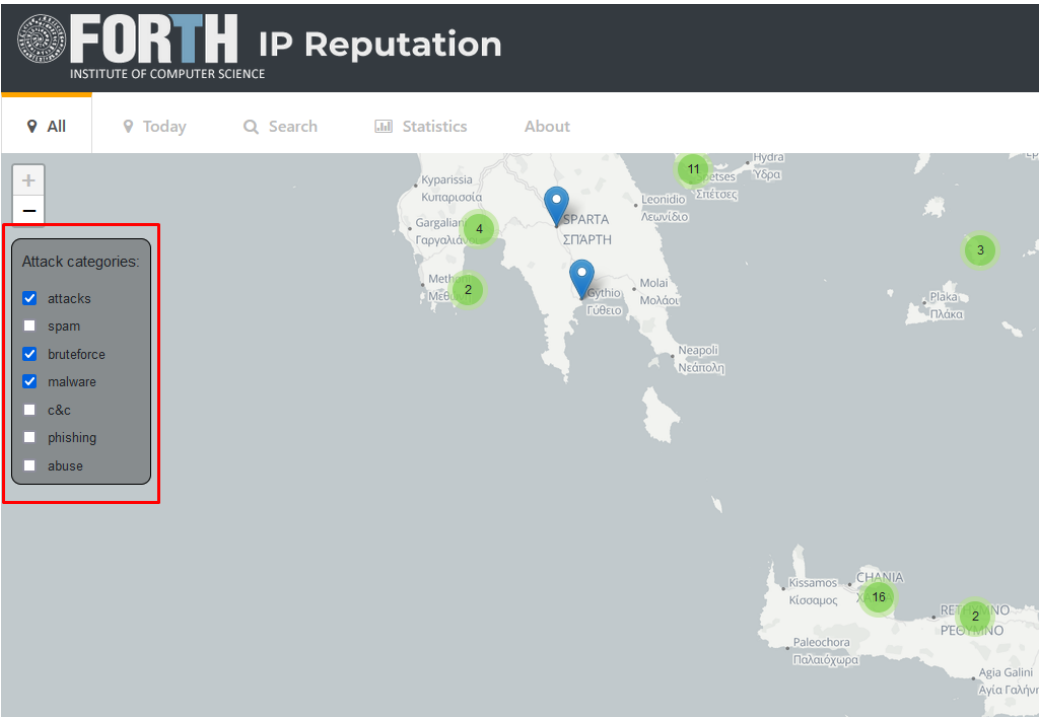


Figure 3.11: Check/Uncheck boxes to show/hide chosen attack categories.

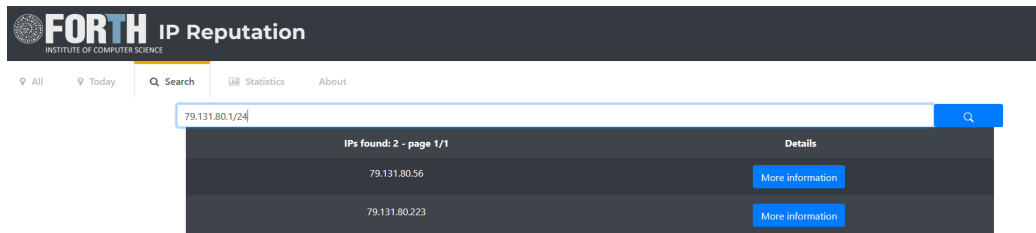


Figure 3.12: Inspecting search module.

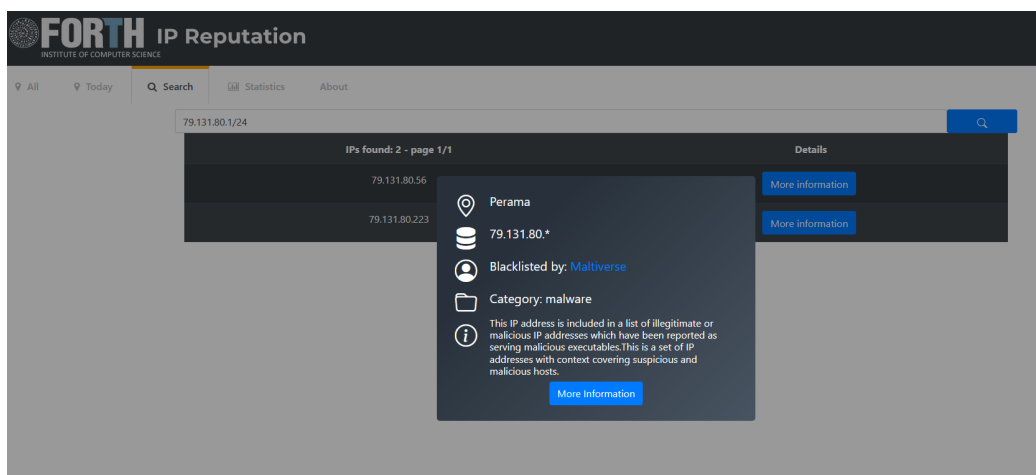


Figure 3.13: Inspecting search results.

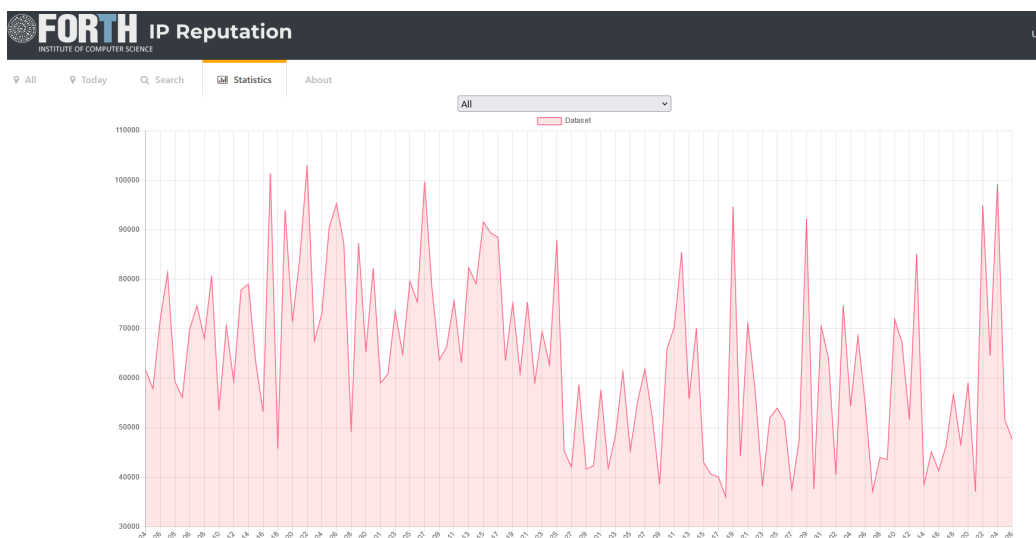


Figure 3.14: Inspect attack distribution worldwide.

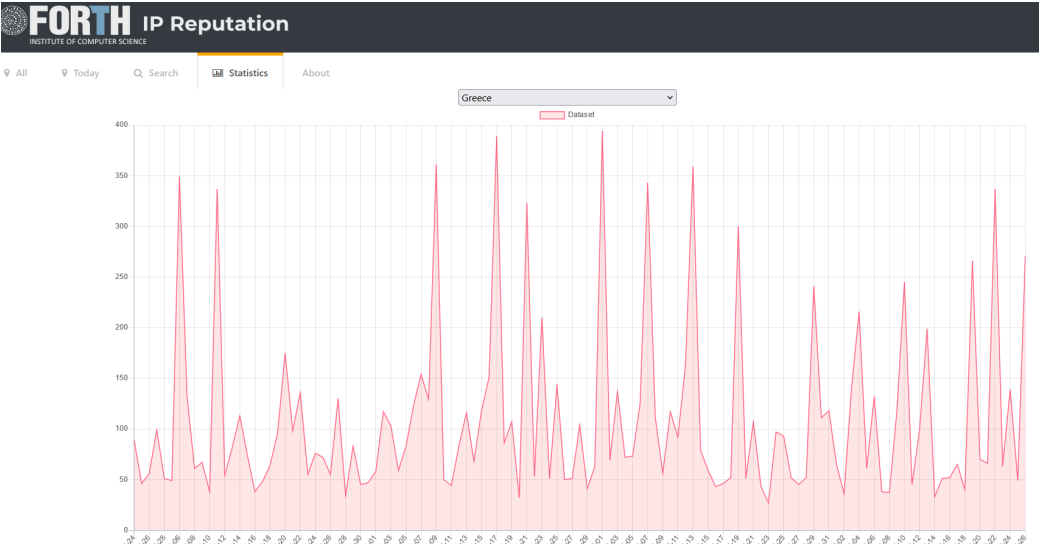


Figure 3.15: Inspect attack distribution of a chose country, Greece.

```
# Generated Mon Jun 26 2023 14:30:33 GMT+0300 (Eastern European Summer Time)
# This is a list of malicious ip addresses.
95.70.166.13X
104.248.62.1X
47.180.212.13X
143.198.39.1X
187.200.175.1X
89.23.96.23X
51.79.49.7X
20.200.169.4X
24.144.96.1X
159.223.100.1X
5.42.64.3X
89.23.107.3X
137.184.84.1X
159.223.147.254
100.25.183.1X
102.68.156.4X
103.155.92.10X
103.155.92.141
103.155.92.2X
103.155.93.1X
103.155.93.24X
103.155.93.4X
103.252.116.5X
```

Figure 3.16: OSINT data exchange module.



## Chapter 4

# Statistics

### 4.1 Overview

This chapter serves as the entry point to a 4-month research collection dedicated to statistical analysis in the realm of cyberattacks. This statistical analysis is conducted on a comprehensive dataset collected from reputable sources. Despite the trustworthiness of the sources, due to GDPR compliance, an IP address is considered personal data, that can be used to identifying an individual. Based on these constraints, our statistical analysis is focused on the multitude of the attacks, their origin limited to country level and time span.

### 4.2 Findings

#### 4.2.1 Attack Distribution per Capita

A follow-up statistic of attack distribution (Chapter 4 Statistics tab), is identifying the percentage of cyberattacks originating from different countries based on the population of those countries. Combining the collected cyberattacks from each country, we calculated the ratio of average attacks per country to the population of this country. As shown in the Figure 4.1, the deeper the blue color on the country, the more attacks have been reported on average. Focusing on the deepest blue colors, we see China and then United States averaging with huge distances with all other countries. India is third on attacks, while Russia and Brazil hold high rank too.

Now, we combined numbers of attack incidents with current country population to calculate the percentage of cyberattacks per capita. In Figure 4.2, we look at completely different results. In this figure, we're spectating the Top20 countries with the biggest average attacks/ country population ratio. Surprisingly, China (representing as HK) and US are not even at top5. The first country averaging more attacks per population are the British Virgin Islands, averaging 90 attacks per day with 30,000 thousand citizens, Figure 4.3.

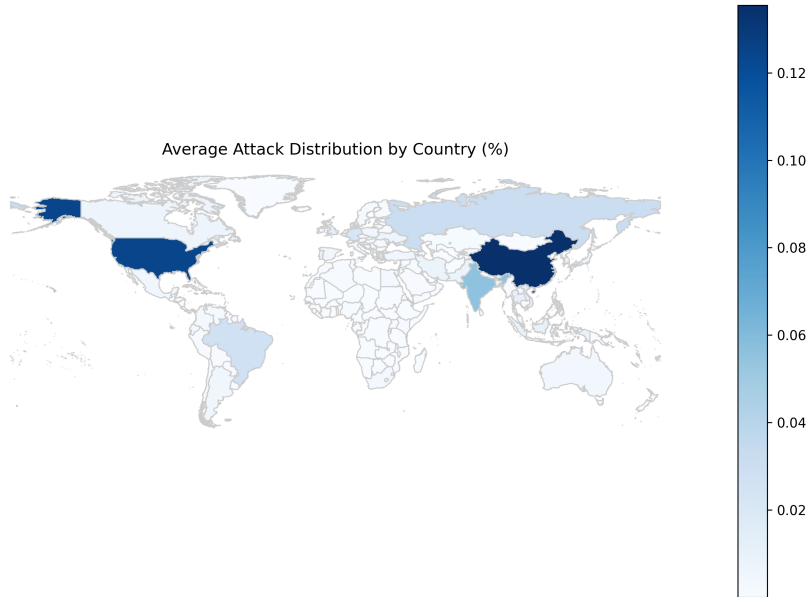


Figure 4.1: Choropleth map of attack distribution.

#### 4.2.2 IP Occurrence

Moving on to our next finding, we conducted research on the occurrences of IP addresses in our dataset. This statistic may have different interpretation, based on the storing policies of each maintainer. Having the same IP in a dataset, without any further information, leads to many results. It may indicate the attack duration, a repetition of the attack, or a blacklist clean up policy that varies throughout the organization. The latter case causes several issues while analyzing cyberattacks on statistics, especially if blacklists does not provide information of the first occurrence of the IP address. Despite these issues, looking at the chart on Figure 4.4, we observe that 10% of the attacks are from IP addresses that have not been blacklisted over a four-month period. Close to 50% of all the attacks, have been seen and reported 2–10 times and 30% are reported 11–100. From these numbers, we can surely say the over a four-month period about the 80% of the attackers are known. Considering the blacklist various policies, some IP addresses may never be excluded from a blacklist if not whitelisted or may be excluded after a fixed period of time handled by the maintainer. Lastly, we observe that approximately, 10% of the attacks are originated from sources reported over 100 times, concluding that blacklists can be utilized as a proactive mechanism eliminating the case of false positive by only acquiring the most-reported IPs.



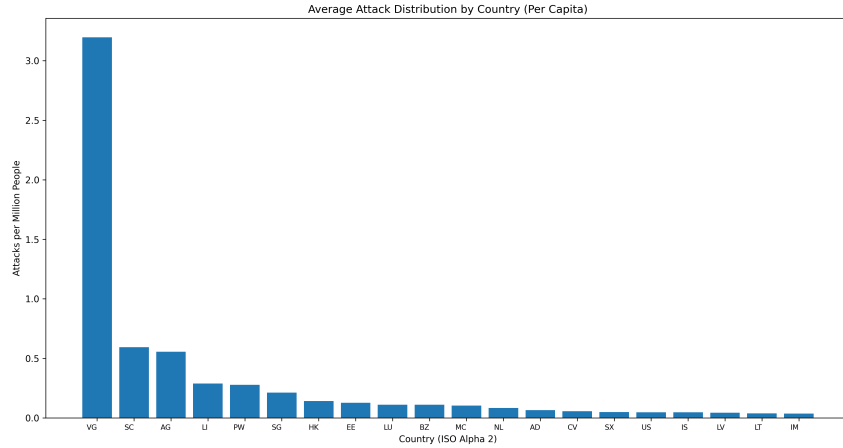


Figure 4.2: Attack distribution per capita.

### 4.2.3 Attack Category Distribution

Analyzing the attack percentages within different categories provides valuable insights into the prevalent threats, as seen in Figure 4.5.

Malware attacks account for the largest portion at 56.0%, indicating the widespread use of malicious software to compromise systems and steal sensitive data. Command and control (C&C) attacks, although relatively less prevalent, still pose a threat at 0.6%, primarily involving the establishment of unauthorized control over compromised devices. Phishing attacks, comprising 1.3% of the total, while phishing domains and emails are taken down faster. Attacks in a broader sense make up 21.3% of the total. Abuse category, holds 19.9% while spam persists at 0.8%. Spam is often categorized as abuse of Internet usage, so that explains the low rates.

### 4.2.4 Attack Origin

In the modern Internet, performing an attack does not always require the use of your own IP address. Attackers managed to abuse free web hosting services to serve malicious content while keeping their identity hidden. So, we search which IP addresses, collected from a four-month period, are included in large corporate free web hosting services. Specifically, we looked up the three different organizations' IP ranges, Google, GitHub, GitLab. Google's public IP addresses are known and in our case mostly used for file sharing such as Google forms, documents, slides, spreadsheets, etc. GitHub and GitLab are known DevSecOps platforms with public sharing option like GitHub pages or directly exposed static webpages. Searching through these organizations, we found out that 834 unique IP addresses belong to Google, 91 belong to GitHub and 5,121 to GitLab services. The number

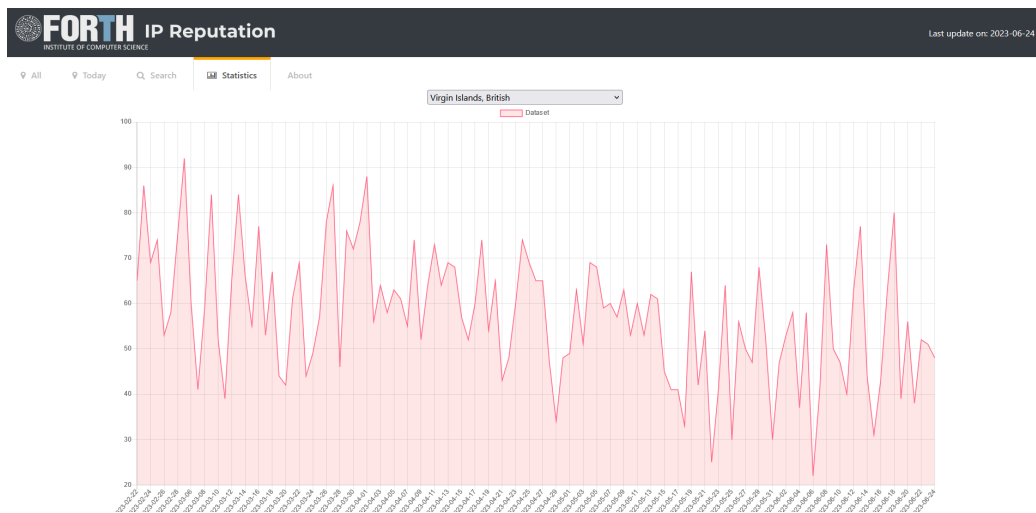


Figure 4.3: Attack distribution of the Virgin Islands.

of reports for each organization is 13,246 reports for Google, 8,068 for GitHub and finally 111,254 reports for GitLab. This result shows the impact of blacklisting an IP address. These IP addresses may be assigned to another user, resulting that the future user will be blocked if the blacklist is poorly maintained.

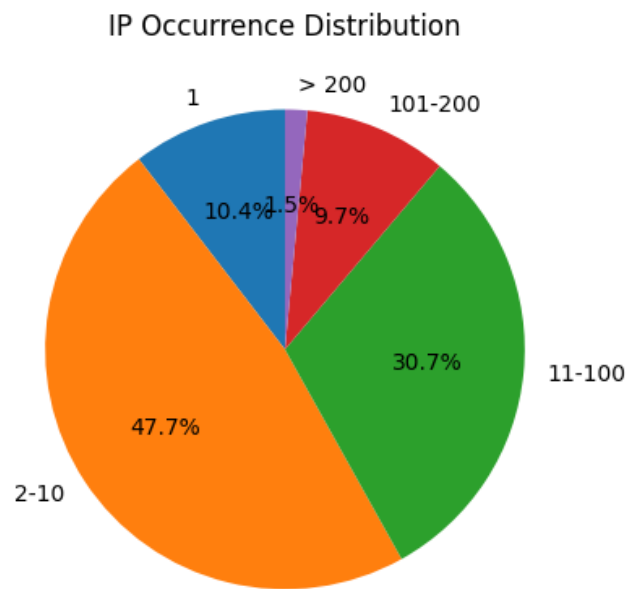


Figure 4.4: IP address occurrence.

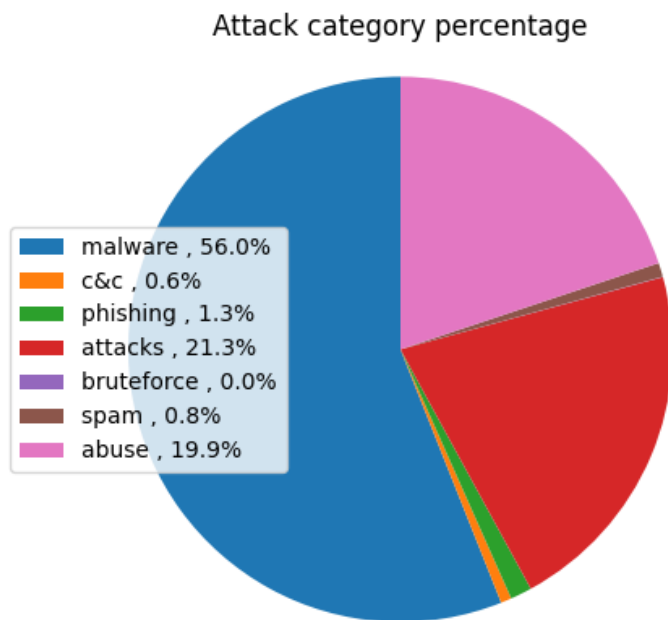


Figure 4.5: Attack category distribution.

## Chapter 5

# Related Work & Limitations

While this study was based on OSINT data, we will exclusively focus on open-source projects and works, that share ideas, techniques, features and discuss improvement on OSINT based tools. Many of the projects we will look, have already contributed to our study by introducing problem solutions, providing OSINT data and posing issues that eventually improved RE.D.

**Maltrail :** Maltrail is a malicious traffic detection system, utilizing publicly available (black)lists containing malicious and/or generally suspicious trails, along with static trails compiled from various AV reports and custom user defined lists, where trail can be anything from domain name, URL, IP address or HTTP User-Agent header value. We include Maltrail as a source to our work. This project fetches thousands of IOCs daily. Maltrail's maintainers include some "static" sources, which results to unmaintained blocklists, that contain old information.

**FireHOL blocklists :** The objective is to create a blacklist that can be safe enough to be used on all systems, with a firewall, to block access entirely, from and to its listed IPs. The key prerequisite for this cause, is to have no false positives. All IPs listed should be bad and should be blocked, without exceptions. We included FireHOL blocklists in our work as the first valuable source. FireHOL is updated once per day. FireHOL maintainers tend not to clean up blacklists from old IP addresses, resulting in large ipsets that may contain false positives.

**Malware World :** A system based on +500 blacklists and 5 external intelligences to detect potentially malicious hosts on the internet. It is worth noting that Malware World might be the largest collection of datasets. Unfortunately, as the number of ipsets raises, the map's efficiency drops, overloading the user.

### 5.1 Limitations

Two major limitations that affect the OSINT data have been observed during this study.

**Format of OSINT data** Every maintainer chooses to publish their data in a different format. So, collecting various available sources may be time-consuming, while adapting to each format costs and requires specific structure of the fetching mechanism.

**Maintaining policies** Each blacklist maintainer may follow different policies on adding and removing IP addresses. During this study, we have seen removal procedures per day, week and month. One way to adapt to each maintainer's policy, is to keep only the new IP addresses each day. That is the path that we followed, and enabled us to acquire and analyze ipsets from sources with following different maintaining policies. As for similar projects

## Chapter 6

# Conclusion

In conclusion, the development and implementation of a two-layer system consisting of a parsing module and a visualization module have proven to be effective in handling and analyzing OSINT data sets. This system has provided valuable insights and visualizations that facilitate the understanding and interpretation of statistical information. The parsing module played a crucial role in extracting data from diverse sources such as API endpoints, documents, rules and ipsets. The system efficiently transformed raw data into a structured format suitable for further analysis, enriched with additional information besides the IP address, like the geographic coordinates and a description of the attack categorization. With the parsed data, the visualization module demonstrated IP addresses as pinned locations on a world map, making it appealing to non-familiar users. By employing data visualization techniques, such as charts and interactive board, the system provided users with an insightful overview of the analyzed data, viewing attack patterns and trends. Moreover, the two-layer system was designed to be extensible to utilize future public available data and demonstrate with simple but yet meaningful representation. In summary, the system's is by itself a contribution to the OSINT community, while it is a collection of OSINT data processed and presented human friendly. Also, the threat exchange module can be used as an OSINT source itself, providing users a verified blocklist. As for similar projects





## Chapter 7

# Future Work

While this study established a connection between the OSINT community and not familiar users, we believe that there is place for future development. This dashboard, utilized only thirty-four different sources of blacklisted IP addresses, while there are so many organizations that are willing or will offer their own OSINT data. Adding parsers, will have great impact on cross-validating the IP address' reputation and remove the possibilities of false positives. Furthermore, we can enrich the Tab Statistics with more plots, dynamically updated while fetching new data. Finally, we believe that eliminating false positives is the key to a successful visualization dashboard. That being said, validation services can be improved so that a user can check an IP's reputation as a score of multiple reputation services.



# Bibliography

- [1] C. Tsalousis, “Firehol blacklist-ipsets,” 2023. [Online]. Available: <https://github.com/firehol/blocklist-ipsets>
- [2] “Digitalside.” [Online]. Available: <https://osint.digitalside.it/Threat-Intel/lists/latestips.txt>
- [3] “Cinsscore.com.” [Online]. Available: <http://Cinsscore.com>
- [4] “Alienvault reputation system.” [Online]. Available: <https://www.alienvault.com/>
- [5] “Swiss security blog - abuse.ch.” [Online]. Available: <https://www.abuse.ch/>
- [6] “Emerging threats.” [Online]. Available: <https://rules.emergingthreats.net/>
- [7] “Ip-reputation.” [Online]. Available: <https://github.com/spkostas/IP-Reputation-Global-Map>
- [8] V. Agafonkin, “Supercluster,” 2016. [Online]. Available: <https://www.npmjs.com/package/supercluster>
- [9] IP2Location, “Ip2location,” 2001 - 2023. [Online]. Available: <https://www.ip2location.com/>
- [10] S. Ramanathan, J. Mirkovic, and M. Yu, “Blacklists assemble : Aggregating blacklists for accuracy,” 2018.
- [11] V. G. Li, M. Dunn, P. Pearce, D. McCoy, G. M. Voelker, and S. Savage, “Reading the tea leaves: A comparative analysis of threat intelligence,” in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 851–867. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity19/presentation/li>
- [12] L. Guo, “An empirical analysis on threat intelligence: Data characteristics and real-world uses.” 2020. [Online]. Available: <https://escholarship.org/uc/item/5h9983b0>
- [13] FILIGRAN, “Open cyber threat intelligence platform.” [Online]. Available: [about:blank](https://about:blank)

- [14] M. K. M. Stampar, “Malicious traffic detection system.” 2016. [Online]. Available: <https://github.com/stamparm/maltrail>
- [15] V. Agafonkin, “Leaflet,” 2016. [Online]. Available: <https://leafletjs.com/>
- [16] I. Labs, “iocextract,” 2023. [Online]. Available: <https://pypi.org/project/iocextract/>
- [17] K. Reitz, “requests,” 2023. [Online]. Available: <https://pypi.org/project/requests/>
- [18] “Bambenek consulting feeds.” [Online]. Available: <http://osint.bambenekconsulting.com/feeds>
- [19] “Cisco talos - additional resources.” [Online]. Available: <http://www.talosintelligence.com/additional-resources/>
- [20] “Clean mx - realtime db.” [Online]. Available: <http://www.clean-mx.com>
- [21] “Cybercrime.” [Online]. Available: <http://cybercrime-tracker.net/>
- [22] “Daniel gerzo bruteforceblocker.” [Online]. Available: <https://danger.rulez.sk/index.php/bruteforceblocker/>
- [23] “Emerging threats.” [Online]. Available: <https://rules.emergingthreats.net/fwrules/emerging-Block-IPs.txt>
- [24] “My ip - blacklist checks.” [Online]. Available: <https://myip.ms/info/about>
- [25] “Sblam! zabezpiecza formularze przed spamem.” [Online]. Available: <http://sblam.com/>
- [26] “Benkow.” [Online]. Available: <https://benkow.cc/export.php>
- [27] “Abuseipdb.” [Online]. Available: <https://api.abuseipdb.com/api/v2/blacklist>
- [28] “Blackhole.” [Online]. Available: <https://ip.blackhole.monster/blackhole-today>
- [29] “pop3gropers.” [Online]. Available: <https://home.nuug.no/~peter/pop3gropers.txt>
- [30] “Dataplane.” [Online]. Available: <https://dataplane.org/>
- [31] “ut capitole.” [Online]. Available: <https://www.ut-capitole.fr>
- [32] “Feodotracker.” [Online]. Available: <https://feodotracker.abuse.ch/downloads/ipblocklist.csv>
- [33] “Honeydb.” [Online]. Available: <https://honeydb.io/>

- [34] “mirai.” [Online]. Available: <https://mirai.security.gives/index.php>
- [35] “Mitchellkrogza phishing database.” [Online]. Available: <https://raw.githubusercontent.com/mitchellkrogza/Phishing-Database/master/phishing-links-NEW-today.txt>
- [36] “Neo23x0.” [Online]. Available: <https://raw.githubusercontent.com/Neo23x0/signature-base/39787aaefa6b70b0be6e7dc425b65a716170ca/iocs/otx-c2-iocs.txt>
- [37] “openphish.” [Online]. Available: <https://openphish.com/feed.txt>
- [38] “phishtank.” [Online]. Available: <http://data.phishtank.com/data/online-valid.json>
- [39] “pulsedive.” [Online]. Available: <https://pulsedive.com>
- [40] “Rescure blacklist.” [Online]. Available: [https://rescure.me/rescure\\_blacklist.txt](https://rescure.me/rescure_blacklist.txt)
- [41] R. S. of Arts and Sciences, “Network monitoring system.” [Online]. Available: <https://report.cs.rutgers.edu>
- [42] “IsC sans.” [Online]. Available: <https://isc.sans.edu/>
- [43] “Sslbl.” [Online]. Available: <https://sslbl.abuse.ch>
- [44] “Threatfox.” [Online]. Available: <https://threatfox.abuse.ch/>
- [45] “Threatview.” [Online]. Available: <https://threatview.io>
- [46] “Urlhaus.” [Online]. Available: <https://urlhaus.abuse.ch>
- [47] “Viriback.” [Online]. Available: <https://tracker.viriback.com>