POLITECNICO DI TORINO

Master degree course in Ingegneria Informatica

Master Degree Thesis

Visualizing Cybersecurity - a comparative study toward a security visualization methodology

Supervisor
prof. Cataldo Basile
prof. Andrea Atzeni

Candidate
Simone Cavallo

Academic Year 2022-2023

Summary

In the contemporary digital landscape, safeguarding against cybersecurity threats has emerged as a paramount concern for both organizations and individuals. The ability to comprehend, analyze, and respond effectively to security issues plays a pivotal role in protecting sensitive data and preserving the integrity of critical systems. This thesis delves into the realm of security data visualization, aiming to augment the understanding and response capabilities concerning cybersecurity threats.

The research begins with a systematic review of the existing literature, encompassing diverse databases and employing relevant keywords. Through this comprehensive approach, a vast array of scholarly articles is curated, providing a robust foundation for the subsequent analysis and exploration of the current state of the art in security visualization. The analysis of the compiled articles adopts a qualitative approach, entailing reading and concise summary of the key points in each article. Through this process, common themes and trends prevalent in the literature are identified, offering valuable insights into the practical strategies employed in the visualization of security data. This comprehensive analysis, which aims to address the research questions such as identification of the current state of the art and the practical strategies adopted in the field of security visualization, together with the identification of related gaps between theory and practice, serves as the foundation to address these questions and present subsequent results in subsequent chapters. The comprehensive review of existing literature has unveiled the extensive utilization of visualizations in the realm of cybersecurity. These visualizations have been employed to facilitate a range of tasks, including threat detection and network analysis. The review highlights the continuous research support and ongoing innovation within the field, particularly with regard to the development of novel methodologies and the incorporation of user-centered design approaches. Moreover, through the analysis of the literature, the study has successfully identified and defined the optimal objectives for security visualization. These objectives encompass achieving a deep comprehension of security data, effectively detecting and responding to security incidents, proficiently communicating security information, supporting decision-making processes, and augmenting security awareness among users and system operators.

The thesis delves into the examination of tools utilized in security data visualization, categorizing their pertinent aspects and evaluating their efficacy in enhancing comprehension and responsiveness to cybersecurity issues. Where feasible, illustrative use cases are presented to showcase the practical application of these tools. Furthermore, the discussion encompasses a comparative analysis of proprietary tools, elucidating the strengths and weaknesses inherent in various approaches. The studies of these instruments reveal that visualizations have the potential to enhance cybersecurity by providing valuable insights into complex data and facilitating decision-making processes across various security domains. However, when compared to proprietary tools, the preference still leans towards the latter in terms of adoption. This preference can be attributed to the prototype-like nature of the majority of solutions encountered in the research,

as well as the specialized support and services offered by these proprietary solutions. Additionally, it was evident that current market solutions do not fully leverage the innovative visualization methodologies derived from research, indicating the potential for further improvements resulting from such research efforts.

Furthermore, the thesis exposes a significant research gap concerning the utilization of security visualization tools by non-expert users, with a specific emphasis on general practitioners. The examination of relevant research literature has identified multiple research contributions centered around the examination of aesthetics in the creation of visually captivating and impactful warning signs. These investigations have focused on evaluating the effectiveness of different visualization effects in alerting the general public about the dangers they may encounter while navigating online platforms. The findings derived from these studies underscore the considerable influence that aesthetics can wield in shaping users' perceptions and level of engagement with online security warnings and cautions. Furthermore, these findings emphasize the applicability of user-centered design principles in enhancing internet security measures.

Additionally, the thesis presents categorizations and statistical analyses that contribute to a deeper analysis and comprehension of security data. Various categorization approaches are examined, shedding light on their impact within the context of security visualization. By means of the analyses conducted, the thesis discusses the process of evaluating gaps in security visualizations and proposes solutions to address these gaps between theory and practice. In particular the study identify the lack of standardization and evaluation of visualizations in the context of research and proposes modifications to an established framework endorsed by SANS, previously presented, proposing a possible implementation obtained through a refinement of a the cited framework, pointing out the various steps that as a result of the review showed critical issues and need for improvement. More specifically, the changes concern extending the analysis of the Exploration phase to encompass existing solutions, involving evaluating both complete solutions that are already deployable and prototypes that offer room for further extensions or modifications and enhancing the Feedback and Fine-tune phase, incorporating privacy and security considerations, ensuring that the visualization tools adhere to necessary standards and guidelines, promoting their wider adoption. Other minor additions to the Visualization Goals, Data Preparation, and Visualization phases were made, further improving these phases, making them more robust and well-defined. By addressing the identified gaps and incorporating necessary considerations, the framework becomes more comprehensive and applicable in diverse security visualization contexts.

In addition to presenting the aforementioned methodology, the study propose a set of potential strategies for validating the proposed framework such as Expert consultation, for seeking feedback and input from subject matter experts in the field, User testing, namely conducting user testing sessions with individuals who would be potential users and Comparative analysis, for comparing the methodology used in this study to other existing methodologies or frameworks in the field. By exploiting these validation processes, the unique requirements and constraints of each organisation can be effectively addressed.

This study outlines also potential practical strategies for leveraging security visualization tools to enhance threat protection. These strategies are derived from the insights gained through the evaluation and analysis of the existing options, evaluated as valid and available at the present time. Within these, we touch upon fields such as vulnerability analysis by exploiting tools that enable the measurement pf the coverage of the attack surface in real-time, moving on to threat analysis by employing solutions that gain detailed insights, enabling efficient resource organisation and prioritisation. By capitalizing on the strengths and successes of these solutions, organizations can establish a robust framework for the development and implementation of their own effective strategies.

In summary, the methodology employed in this research investigates the potential of security visualization and identifies its gaps and limitations. The obtained results and remarks serve as valuable insights to enhance the design and development of applications dedicated to enhancing computer system and network security.

The study has identified gaps that require further research and solution development. One key gap is the need for improved information exchange in security among users at the same level and different types of end users. Exploring the potential of existing tools and proposing

a methodology or solution to address this gap would be valuable. Additionally, there is a need to investigate the use of visualizations in cybersecurity awareness training to enhance learners' understanding and awareness of digital threats. Conducting a study to assess the effectiveness of visualizations in conveying complex cybersecurity concepts in an accessible and engaging manner would be a valuable contribution to the field.

Contents

Chapter 1

Introduction

Cybersecurity is a critical issue facing organisations and individuals today, particularly as the volume and complexity of security data continues to increase with the proliferation of technology and the data on the Internet.

Big data analytics in cybersecurity involves collecting and analyzing large amounts of digital information to predict and prevent potential cyber threats and attacks. A strong cyber defense posture enables organizations to anticipate actions that may lead to attacks, particularly as the amount of data generated by connected devices and the Internet of Things continues to grow. As new technologies and systems emerge, so do new vulnerabilities, making security a constantly evolving target in the face of increasing data volume and complexity.

Managing and analysing these "big data" is a major challenge for cybersecurity professionals, and traditional methods of data analysis are often inadequate and insufficient.

One solution to this problem is the use of visualizations, which can help analysts and decision makers understand and manage complex security data more effectively.

The hypothesis that visualizations enhance users' comprehension of information and subsequently improve their performance in making judgments and decisions is strongly supported by research findings [1]. This effect is primarily attributed to cognitive mechanisms. Cognitive mechanism refers to the cognitive processes involved in understanding and processing information. In the context of visualizations, these processes include perception, attention, memory, and reasoning. One of the most important cognitive mechanism in this context is the Preattentive Processing [2]. This is an automatic and rapid processing of visual information that occurs before conscious attention is directed toward the entire visual field. The process involves the initial perception and analysis of basics visual features such as colors, orientation, line ends, contrast, tilt, curvature, shape and size. These simple features are extracted from the field of view in the preattentive system quickly, effortlessly and in parallel without any attention being focused on the visual field, allowing the brain to quickly identify patterns and relationships in visual information.

Information visualisation has the potential to greatly support cybersecurity efforts by allowing analysts to recognise patterns and anomalies in large datasets, and by allowing humans to remain an integral part of the analysis process. Information visualisation involves the transformation of data into interactive graphical displays, taking advantage of the visual perception abilities of humans, capable of visualising patterns and anomalies faster. It can be used for a variety of purposes, including exploration, discovery, decision making, and communication of complex ideas.

Security data visualisation can be used in many areas in information security and can analyse a very different heterogeneous group of data, such as meta-data, network traffic data, enterprise log data, and more. The use of security data visualisation in areas such as security metrics, security monitoring, anomaly detection, forensics, and malware analysis can greatly enhance the capabilities of security professionals and contribute significantly to the overall effectiveness of security measures.

Cybersecurity visualization is important due to different dimensions that may depict security analyses [3]. One first example to show its importance is about Visual threat investigation through

the use of graph visualization. Graph visualization is a technique used to represent complex data structures and relationships between objects in a visual form. The primary goal of graph visualization is to facilitate data exploration and analysis by providing an intuitive and interactive way to visualize and navigate large and complex datasets. The technique involves mapping data elements to graphical objects, such as nodes and edges, and applying layout algorithms to arrange these objects in a meaningful and informative way. The utilization of graph visualization has been shown to assist analysts in the identification of patterns and anomalies that may indicate potential threats. By combining human pattern recognition abilities with the data processing capabilities of machines, it becomes possible to uncover anomalies that may have otherwise gone unnoticed. This methodology has demonstrated its ability to identify data breaches, detect malware entry points, predict external attacks, and reveal vulnerabilities in an organization's perimeter.

The presented example showcases a simplified representation of data reflecting user logins to an online portal. Each individual component represents an online account, and is linked to one or more IP addresses that have accessed it (figure 1.1).

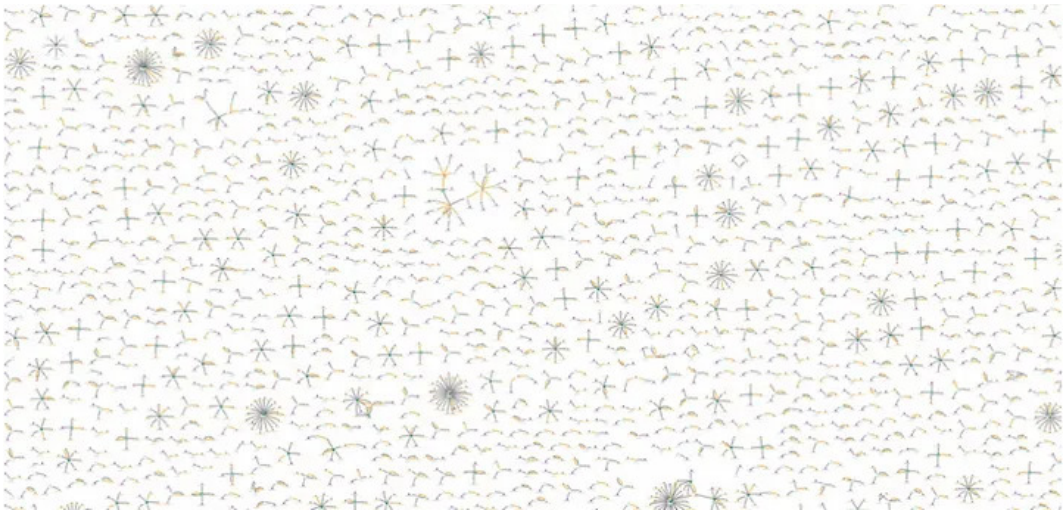


Figure 1.1: Users logins

The visualization highlights that the majority of accounts have been accessed by a small number of IP addresses, with a few exceptions that show a significantly higher number of accessing IP addresses (figure 1.2).

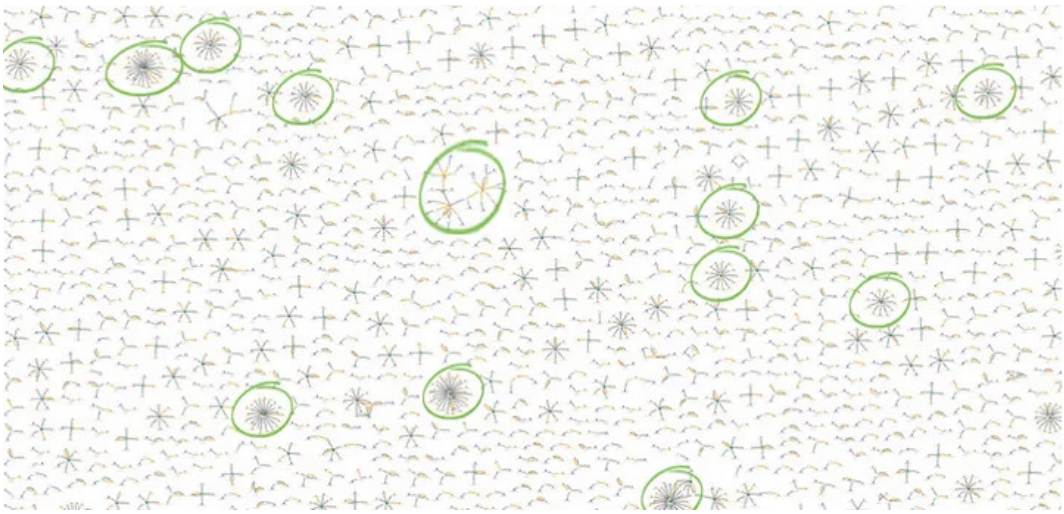


Figure 1.2: Users logins with focus on larger shapes

A deeper analysis of such exceptional cases allows the analyst to identify potential security

threats, such as in the case of a UK-registered user who has accessed the system from over 20 different locations (figure 1.3). This approach has the potential to significantly enhance the security of organizations and provide critical insights into potential threats.

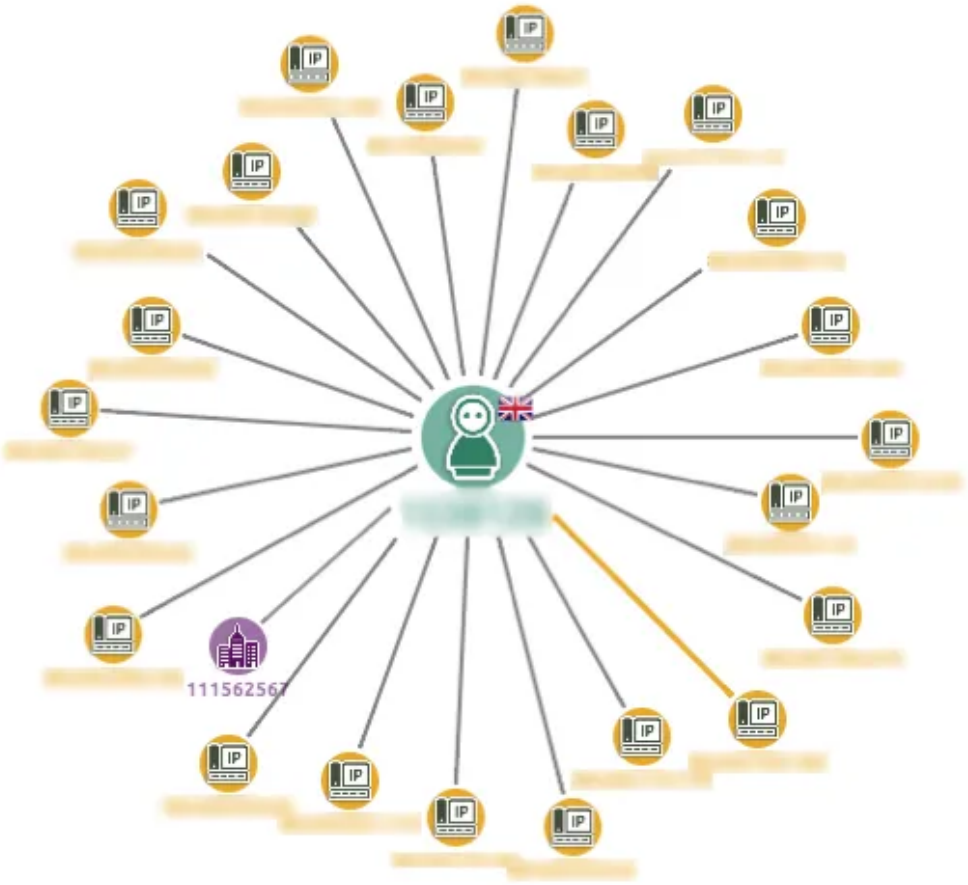


Figure 1.3: User logins for analysis

Another example of utilization showed by the Cambridge Intelligence's article [3] is about the Visual threat detection phase, always using the graph visualization. The visualization enables analysts to perform quick and precise assessments of events and cases flagged by automated tools. The visualization of a fictitious company's IT network showcases the various offices in different locations and the subnets between them. In this instance, a threat signature has been matched, indicating that an occurrence has taken place that appears similar to a previous malicious event. An analyst is now responsible for reviewing this alert and determining the necessary action quickly. By delving into the basic network overview, an intuitive, fast and comprehensive view of the situation can be obtained. In this scenario, a computer in Cambridge and a phone in Paris are sharing an unencrypted communications, and these devices can be isolated to determine which connected devices may have been impacted. By using this type of visualization, analysts can quickly and accurately review events and cases flagged by automated tools (figure 1.4).

This second example shown here, that represents a more common situation in a business context, and the others presented in the original article prove how the same visualization technique can fit different types of analysis, in different phases and how the analysts can benefit from the different visualization techniques for their purposes.

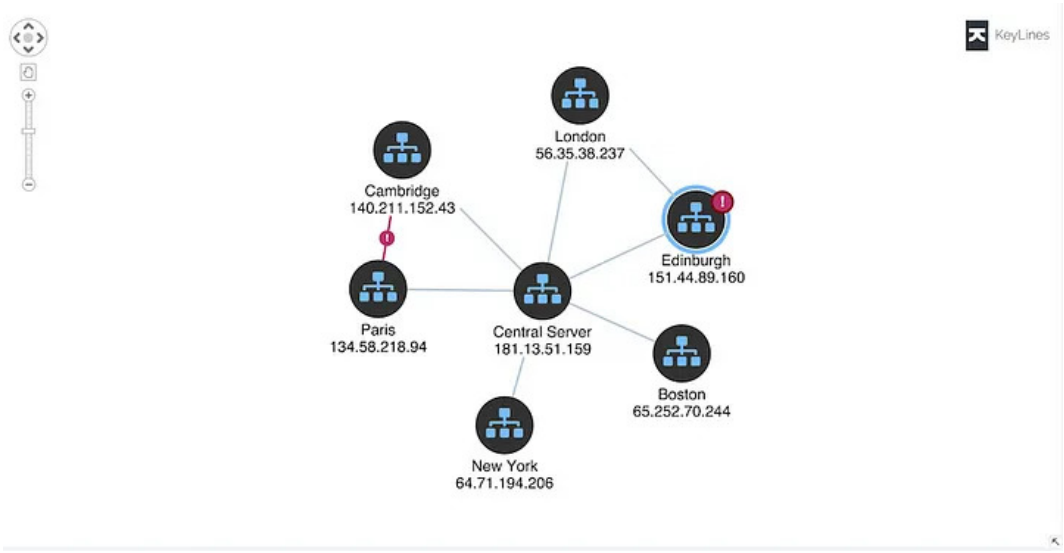


Figure 1.4: Network representation

The goal of this thesis is to investigate the use of visualisations in the field of cybersecurity. Specifically, the research aims to define the following purposes:

- Define a theoretical framework of the state of the art in security visualisation.
- Define and indicate practical strategies to be followed.
- Understand any gaps between the proposed theory and the practises used.

The theoretical framework should provide a comprehensive overview of the current state of research and practice in this area, including the key concepts, theoretical models, and empirical findings that underpin effective security visualization. The result produced by this work can guide the design and development of effective security visualization tools and systems by providing a set of principles and guidelines based on the current state of research and practice. Another type of use for which the framework can be exploited is the evaluation of the effectiveness of existing security visualization tools and systems, work that has been conducted for example in the following chapters of this thesis, exploiting the work developed by Balakrishnan in the chapter 3.1.

Any evaluations that have been carried out, both on theoretical research and on the various tools, have made it possible to describe hypothetical practical strategies based on the different needs and/or on the actual availability and characteristics of the developed products taken into consideration. Taking into consideration the different analyses, various types of practical indications can be indicated, such as for example the simple suggestion to use certain methods or tools, when possible, or in other cases an indication, supported by correct information, if wanted or needed to develop their own solution, starting from the foundations of those who have already developed and conceived.

These bases already present will precisely be the subject of discussion as regards the various possible gaps present, such as for example those between the objectives of the security visualization, and the results actually achieved, or a different type which can be the gap between the tools developed, very often as we will see more at an experimental level and the tools actually used.

To address these purposes, this thesis will review the existing literature on cybersecurity visualisations. The results of this research will provide insights into the current state of the field and offer recommendations for improving the effectiveness of visualizations in cybersecurity.

Overall, the use of visualizations has the potential to greatly improve cybersecurity by providing analysts and decision makers with a clearer understanding of the data and trends related to cyber threats. This research aims to contribute to the body of knowledge on this important topic and help organizations better protect themselves against cyber attacks.

Chapter 2

Analysis Methodology

The goal of this chapter is to describe the methodology that will be used to analyze the existing literature on the use of visualizations in cybersecurity. To achieve this goal, the following research questions will be addressed.

- What is the current state of the art in security visualisation?
- What are the practical strategies that are being used to visualize security data?
- What are the gaps between the proposed theory and the practises used in security visualization?

To answer these questions, a systematic review of the literature was conducted. The review include a search of multiple databases, including ACM Digital Library, IEEE Xplore, Springer Link and Google Scholar, using relevant keywords such as "cybersecurity visualizations", "security data visualization" and "information visualization in cybersecurity". The review include articles published in English without restrictions on the date of publication.

Once the articles have been identified and collected, they were analyzed using a qualitative approach. The analysis has involved reading and summarizing the main points of each article, and identifying common themes and trends in the literature. The results of the analysis was used to answer the research questions and to provide insights into the current state of the field and will be presented in the subsequent chapters of the thesis.

The analysis also includes survey-type articles conducted on the state of security visualization over the years. Although this is not a survey article, any theories, proposals and categorizations present in these articles, which was also analyzed, was considered, expanded and eventually refuted in the following chapters.

In addition to a systematic review of the literature, this research also include a practical component. To complement the theoretical analysis, a practical test was conducted to investigate the effectiveness of available visualization tools in a real-world cybersecurity setting. The test involved selecting an available sample of visualisation tools identified through the literature review and online research to test their capabilities and performance in analysing security data. In particular the selection criteria for the tools used in this study primarily revolved around their presence in the examined scientific articles, considering their relative availability, as well as their adoption within industry contexts, and additionally, the choice took into account their popularity and subsequent utilization in real-world settings, thereby ensuring their relevance and widespread usage. The data used for the tests were a combination of simulated and real security data.

The results of the practical test were compared and contrasted with the findings from the literature review, to provide a comprehensive understanding of the use of visualizations in cybersecurity. This allowed for the identification of both the strengths, the real-world effectiveness and limitations of the selected visualization tools, as well as any gaps between the theory proposed in the literature and the practical use of these tools.

To enhance the credibility of the findings, a thematic analysis was carried out to identify prevailing themes, trends and statistics in literature. The approach entailed categorising articles based on their shared themes before conducting an extensive examination to uncover deeper insights about the contemporary status of the field. This technique should provide us with a comprehensive and balanced picture of cybersecurity visualisation's current state.

In addition to examining data types and processing techniques, this study also evaluated the influence of user experience on security visualizations' effectiveness. This involved analyzing the type of data that is being visualized, such as log files, network traffic, and attack patterns.

This analysis includes an assessment of how design elements like color, shape, and size impact users' ability to understand and make full use of visualization tools. Finally, to address the limitations of the study, a critical evaluation of the methodology used in this research was conducted. This evaluation considered factors such as the representativeness of the sample of articles used in the literature review, the generalizability of the results, and the limitations of the practical test. The evaluation provided insights into the strengths and weaknesses of the methodology used in this research and suggest areas for improvement in future studies.

Moreover, the research methodology was expanded to incorporate empirical findings derived from a real-world scenario and an expert consultation session involving cybersecurity specialists from Politecnico di Torino. This experience provided practical insights into the implementation and efficacy of commonly employed techniques for enhancing cybersecurity. These insights played an important role in extending the analysis methodology and identifying potential disparities between the theoretical constructs proposed in the existing literature and the pragmatic application of the conceptual framework. By validating the analysis methodology, a comprehensive comprehension of the utility of visualizations in the realm of cybersecurity was achieved. The feedback received from the experts regarding the efficacy of the chosen visualization tools in real-world contexts were instrumental in fine-tuning the analysis methodology and shedding light on any inherent limitations associated with the selected tools.

Overall, the methodology used in this research combines both a systematic review of the literature and a practical test of available visualization tools. This mixed-methods approach will provide a comprehensive understanding of the use of visualizations in cybersecurity and allow for the identification of both the current state of the art and the real-world effectiveness and limitations of these tools.

Chapter 3

Theoretical Research

In this chapter, we will explore the theoretical foundations of security visualization, including the different types of security visualization techniques, the theories and models that underpin security visualization, and the applications of security visualization in the field of information security. Understanding these theoretical foundations enables us to create more effective security visualizations that help security professionals identify and respond to security threats more efficiently.

Security visualization is not a new concept, and its roots can be traced back to the early days of information visualization in the 1960s. However, it was not until the 1990s that security visualization gained widespread attention in the field of information security. This was due, in part, to the growing awareness of the need for improved security measures in response to the increasing number and complexity of cyber threats.

One of the pioneers in the field of security visualization is certainly Raffael Marty [4], author of one of the most comprehensive guide to the principles and practices of security visualization. The book provides a detailed overview of the different types of security visualization techniques and although today it is a fairly dated source, it is certainly to be considered as the basis for all the research produced in the years following its publication given the extensive treatment of topics such as cognitive psychology, visual perception, and information visualization. In his book Marty discusses several challenges that arise when using security visualization to identify and respond to security threats like Information Overload, Data Quality, Visualization Design and more.

3.1 The SANS Framework

Over the years, research works have tried to expand the concepts and go beyond the limitations proposed by the foundations of data visualization. In addition to a more practical part of the work, a necessary effort was made in defining frameworks and standards for the Security Visualization process. Although there are no defined standards in the field of Security Visualization, a reference in this area is given by the work of Balakrishnan [5] which aims to provide general guidelines on the visualization of data concerning security with a repeatable process (figure 3.1). In the proposed model, the process is divided into 5 key steps:

Visualization Goals In this initial phase, the focus is on understanding the objectives and determining what is actually desired to be achieved through the visualization. The suggestion given by the author is that the visualization must be goal-driven or use case-driven and never data-driven. In this way, according to the author, having already a documented idea of the objectives, a better choice will be made when choosing the data and tools for visualization.

Data Preparation This second phase can in turn be divided into two sub-phases. As a first action you have to search and collect the data that will be the object of the visualization. Later, before deciding on the actual method and use for displaying them, the data needs to

be cleaned and converted if necessary into a usable format for the next steps, a process that could take a considerable amount of time.

Exploration In this phase, it is necessary to perform an analytical analysis to determine the optimal approach for visualizing the data. The author suggests the use of a framework of analytical activities developed in the same year [6] whose purpose is to allow the analyst, through the various phases of the framework, to ask himself the right questions and obtain the best results based on his purpose, analyzing all possibilities.

Visualization This specific phase can be approached from two different perspectives. The first is that of aesthetics, or how to use colors, hue, thickness and many other aspects related to visualization through methodologies that can be described in literature, including a chapter in Marty's book [4]. Another point of view on which the perspective can concentrate instead is that of the visualization methodology. Here too there are many sources in the literature that analyze and explain the many methods of displaying information. However, as suggested by the author and as you will see in the following sections of this thesis, the possibilities in case there's the need to develop your own visualization methodology increase exponentially thanks to the use of languages and extensions developed to create visualizations.

Feedback and fine-tune This is the final step in which continuous feedback and the possible availability of new data allow continuous improvement. Unfortunately, not many indications are provided regarding this phase since the options with which these two actions can take place depend a lot on the context in which the framework is used.

As previously mentioned this framework is not to be considered standard for the entire Security Visualization community but for the subsequent work carried out in this thesis it was taken as an initial reference thanks to which analyzes have been carried out on the various tools, when possible, and after that the framework itself was analyzed and possibly detailed or modify it, subject to critical evaluation.

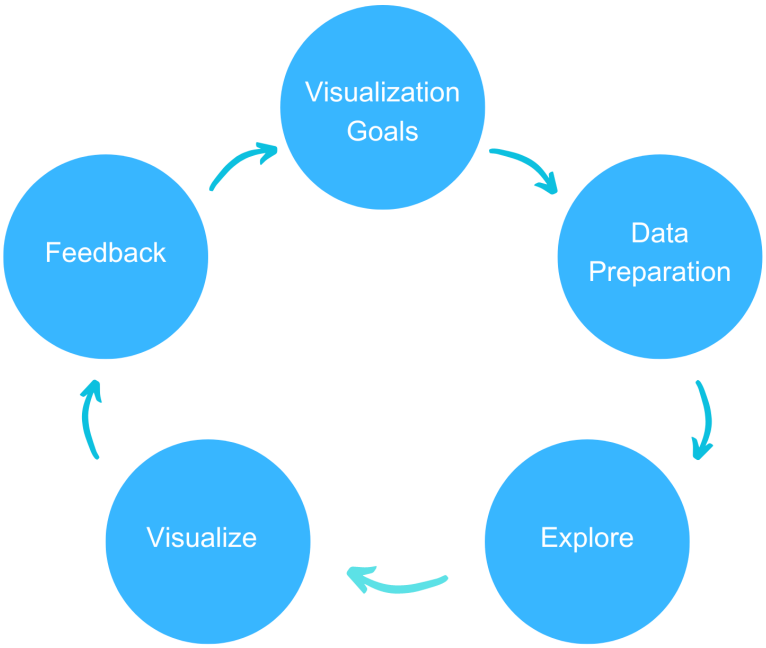


Figure 3.1: Security Data Visualization Process [5]

3.2 VizSec

As expressed in the previous chapter of the methodological analysis, the selection of papers did not have particular filters regarding the origin of the research. However, one of the elements common to many of the works found and analyzed was the membership of the IEEE Symposium on Visualization for Cyber Security or VizSec [7]. In the world of security visualization, VizSec represents the reference forum for workers and researchers from the academic, industrial and governmental world.

The VizSec IEEE Symposium has grown since its inception in 2004. The early years of the conference emphasized the development of new ways of visualizing security data but, over time, it became clear that effective security design does not depend solely on the technical aspect of design at even how well it meets the needs of the people who use it. As a result, the conference has focused more on the human factors of safety simulation, including how to create simple, easy-to-use, and effective simulations for decision-making. While the conference has always focused on the safety picture, the range of topics covered has broadened considerably over the years. In addition to traditional topics such as network security and malware analysis, the conference now covers areas such as threat intelligence, incident response and cybersecurity education and training. As the field of security design has matured, it is clear that interdisciplinary collaboration is essential to achieving effective design. The VizSec conference addressed this need by encouraging interdisciplinary collaboration, bringing together researchers and practitioners from fields such as computer science, psychology and design. With the growing importance of cybersecurity in today's world, the VizSec Symposium has become an increasingly important event for security professionals. The conference provides a forum for sharing best practices, discussing emerging threats and exploring new ways to mitigate cyber risk. Overall, the VizSec IEEE Conference has evolved to reflect the changing needs and priorities of the security community.

3.3 Literature Review

Upon reviewing the extensive research conducted in the field of security visualization, it becomes evident that significant efforts have been dedicated to three primary areas: user-centered design studies focused on developing effective data visualization solutions for security analysts [8, 9, 10], comprehensive surveys investigating data visualization in the context of cybersecurity [11, 12, 13], and innovative approaches or exploratory investigations into data analysis techniques and visualization methods tailored specifically to the domain of cybersecurity [14, 15, 16].

In the context of the first category examined, user-centered design embodies a set of design methodologies that entail the active involvement of identifiable user representatives throughout the product development process. These methodologies aim to comprehensively grasp the users' needs, preferences, and requirements, ensuring the resultant product is finely attuned to their specific contexts and demands. Moreover these design methodologies emphasize the primacy of usability, effectiveness, and the overall user experience, thereby amplifying the intrinsic value and pertinence of the developed solution. In particular, McKenna's [8] work references two key components within the user-design methodology. The first intriguing aspect pertains to the techniques for understanding user needs like the cognitive task analysis (CTA), which is used to understand the cognitive processes and mental strategies employed by individuals when performing complex tasks. It aims to uncover the underlying knowledge, decision-making processes, and problem-solving strategies utilized during task execution. CTA ca be very useful in various domains, because enables researchers and practitioners to gain a deeper understanding of how individuals acquire, process, and apply information to complete tasks effectively. The second noteworthy aspect is the discussion and adoption of the design process model in the context of creating a visualization. Specifically, reference is made to a work produced by the same authors [17], which supports iterative and user-centered visualization design and provides guidelines for achieving a set of design objectives. The study cited provides a comprehensive reference as a practical guide for the design process, making clear connections between the various design processes and decisions regarding visualization design. Furthermore, the paper presents a range of methodologies, both well-established and novel, to be employed throughout the entire framework of the design activity.

In the course of reviewing the articles, it was observed that several recent solutions, discussed in the subsequent chapters, have employed a user-centered design approach in the design and development of the tools. A further aspect unrelated to the development process in this domain, pertains to the paramount importance of devising aesthetics elements and interaction paradigms that not only incorporate robust security measures but also encompass engaging and user-friendly elements. The literature review conducted in this study has revealed several research contributions focusing on the exploration of aesthetics in creating visually captivating and effective warning signs [18, 19]. Investigations have been carried out to assess the utilization of various visualization effects to alert the public about the perils while navigating online platforms. The findings from these studies demonstrate the significant role that aesthetics can play in shaping users' perceptions and engagement with online security warnings and cautions and highlight that user-centered design principles can be applied to improve internet security. It is essential to acknowledge that while these aspects bear substantial importance within the wider realm of online security, they have been deliberately excluded from the purview of this work owing to their specific emphasis on a distinct subject area.

The second prominent type of research conducted is surveys focusing on security visualization, which play a crucial role in providing a comprehensive understanding of the existing landscape. These surveys aim to explore and analyze the various aspects of security visualization, including tools, techniques, methodologies, and challenges. Surveys in security visualization often involve a systematic review of the literature, where researchers gather and analyze relevant articles, papers, and publications, examining different dimensions of security visualization, such as data sources, visualization techniques, user interaction, and evaluation methods. These surveys also help identify trends, gaps, and emerging areas of research within the field. Through surveys, researchers gain insights into the current state-of-the-art in security visualization, allowing them to identify common practices, identify challenges, and propose directions for future research.

The last category addresses the development of unique methodologies and techniques in security visualization. This involves the ability to customize and expand existing visualization approaches, as well as create new ones, to cater to specific needs within the cybersecurity domain. Recent research efforts have not only concentrated on network analysis, which was initially a primary area of interest in security visualization, but have also expanded to encompass diverse domains such as forensic analysis and malware analysis, among others. Researchers have dedicated their efforts to exploring innovative visualization techniques that effectively represent complex data structures, relationships, and behaviors pertinent to cybersecurity. The development of specialized visualizations tailored to specific cybersecurity domains exemplifies the progressive nature of security visualization as a dynamic and evolving research field.

Despite the evolution of research in this field, many of the visualization techniques used, both in research and real-world contexts, are based on the current data visualization theory, which has been extensively described by multiple sources in the literature. One notable source in this regard is Marty, as mentioned earlier, who delves into the subject matter extensively in his book, exploring various aspects such as the multitude of choices within visualization methodologies and the decision-making process involved. Regarding the grouping of visualization techniques used today, one prominent reference cited in the literature, as seen in the work presented by Balakrishnan, is the representation developed by KPI Library known as "A Periodic Table of Visualization Methods" (figure 3.2), which classifies visualization methods into six categories, providing a comprehensive overview of the field. As far as the visualization choice process is concerned instead, despite being an outdated source, the methodology expressed by Marty in one of the chapters of his book (figure 3.3) is to be considered today both a valid solution and a basis for the development of a decision-making process which it can be aimed at a particular case or at a more general purpose.

In the case of visualizations used in cybersecurity, the focus goes beyond producing static images. Instead, it aligns with what Shneiderman [21] refers to as the "Visual Information-Seeking Mantra". The author proposes that advanced graphical user interfaces should adhere to guidelines that enable the rapid presentation of information and facilitate user-controlled exploration, pointing out the importance of interactive and dynamic visualizations that empower users to quickly navigate and explore the data. The methodology that a visualization should adopt, according to the defined mantra, is "Overview first, zoom and filter, then details-on-demand". Specifically, the

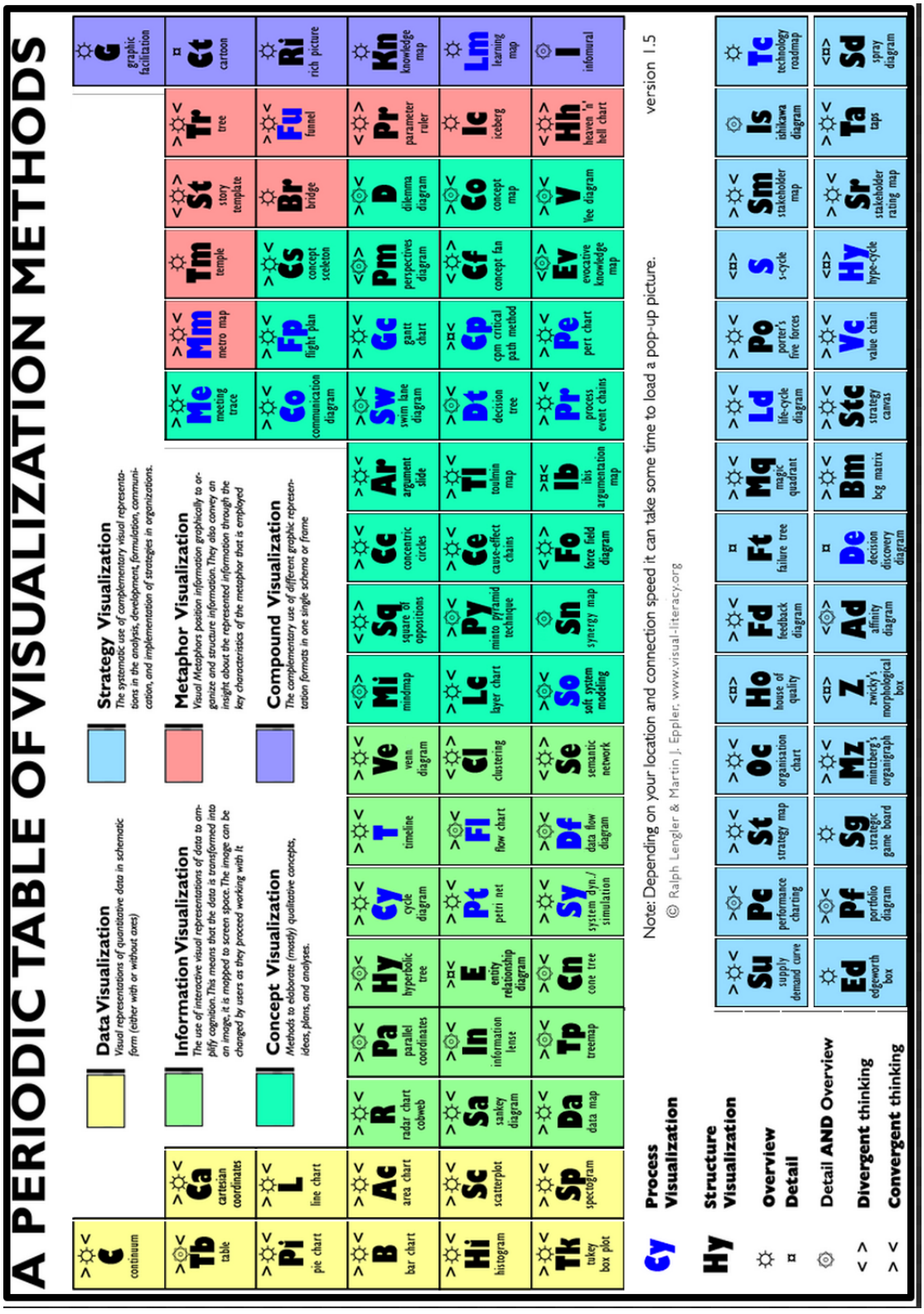


Figure 3.2: A Periodic Table of Visualization Methods(KPI Library, 2014) [20]

methodology expands into seven abstract tasks, which can be briefly summarized as follows:

Overview Present an initial high-level view of the data to provide users with an understanding of its overall structure and patterns.

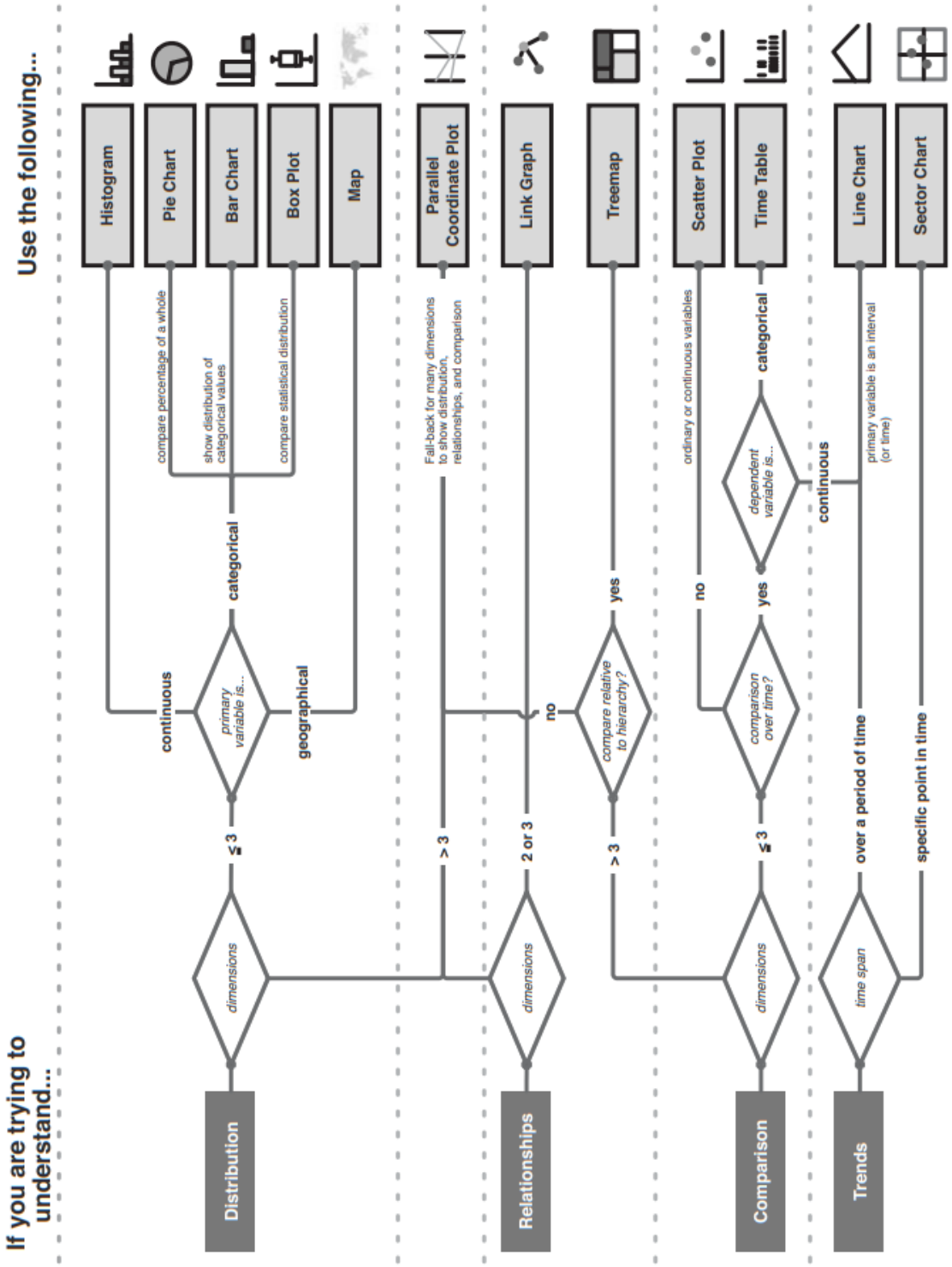


Figure 3.3: Choosing Graph Types, "Marty, R. (2009). Applied security visualization" [4]

Zoom Allow users to zoom in on specific areas of interest to examine details at a more detailed level.

Filter Enable users to apply filters to the data based on specific criteria or attributes, refining the view to focus on relevant subsets.

Details-on-demand Provide users with the ability to access detailed information about specific

data elements or points of interest as needed.

Relate Support the exploration of relationships and connections between different data elements, facilitating the discovery of meaningful insights.

History Allow users to track and revisit their interactions with the visualization, enabling a sense of history and supporting backtracking if needed.

Extract Provide mechanisms for users to extract or export relevant portions of the data or visual representation for further analysis or communication purposes.

Despite the dated nature of this methodology, we can observe the presence of these phases in most of the tools developed in the field of Security Visualization, confirming through expert feedback that the inclusion of these functionalities and the construction of final products implementing this type of design have a positive impact on improving the overall user experience.

All these components described above are made possible thanks to the interactive element, which is always present in modern visual system solutions. The incorporation of interactivity empowers users to actively engage with the visualizations, enabling them to manipulate, explore, and interact with the data in a more dynamic and immersive manner. By offering functionalities such as zooming, filtering, and highlighting, users can navigate and delve into the visual representations, thereby enhancing their understanding, analysis, and decision-making processes. This interactivity aspect not only augments the usability and effectiveness of the visual system but also contributes to an enriched and user-centric experience. During the literature review, the two main types of interaction found can be classified based on their ultimate goal. The first, which can be described as more expressive, aims to modify, for example, the visualization algorithm or its underlying structure. The second type is more exploratory in nature, focusing on the information presented through actions such as zooming and filtering, which are closely tied to the user's actions and provide a means to explore and manipulate the displayed information.

Through the literature review conducted, it has become clear that the use of visualizations in cybersecurity has several key objectives. These objectives are aimed at improving the effectiveness of cybersecurity measures by providing analysts and decision-makers with a clearer understanding of the data and trends related to cyber threats. The ideal objectives of Security Visualization can be delineated as follows:

Understanding security data The primary goal of Security Visualization is to enable analysts to understand security data quickly and intuitively. Data visualization helps identify patterns, correlations, and anomalies that may otherwise go unnoticed. This allows analysts to gain an overview of threats, vulnerabilities, and suspicious activities, facilitating informed decision-making and effective incident response.

Detecting and responding to security incidents Security Visualization aims to provide tools and techniques for early detection of security incidents and timely response. Through data visualization, analysts can rapidly identify indicators of compromise and critical security events, enabling immediate response to mitigate the impact of incidents and restore the security of the environment.

Effectively communicating security information A key objective of Security Visualization is to communicate security information clearly and effectively to various stakeholders, including security experts, business executives, and end-users. Data visualizations can transform complex security data into understandable visual representations, facilitating easier and engaging communication of key security-related information.

Supporting decision-making Security Visualization provides valuable support for informed decision-making related to security. Data visualizations allow analysts to quickly assess options, compare different strategies, and evaluate potential impacts. This enables data-driven decision-making regarding security measures to be implemented, threat mitigation paths, and intervention priorities.

Enhancing security awareness Security Visualization aims to improve security awareness among users and system operators. Through clear and engaging visualizations, users can better understand risks, security policies, and recommended actions to protect their data and digital assets. This contributes to promoting a culture of security and adopting conscious behaviors to prevent breaches and attacks.

Overall, the ideal objectives of Security Visualization are to provide in-depth understanding of security data, detect and respond to security incidents, effectively communicate security information, support decision-making, and enhance security awareness among users and system operators. By understanding these objectives, organizations and individuals can better leverage security visualization to improve their cybersecurity posture and protect against potential threats.

Chapter 4

Tools and Categorization of Relevant Aspects

The first section of this chapter focuses on the various tools, their technical insights and their relative tests or usage examples, used in the field of security data visualizations. The chapter provide an overview of the different tools, developed both in research and non-research contexts, available for security data visualization, which will lead to the discussion of theirs strengths and weaknesses.

The second section of the chapter instead aims to categorize and identify the various aspects of security data visualizations that are relevant to this thesis. In the chapter the various aspects will be presented divided into categories such as types of data, format of data, types of visualizations, and other different categorization.

4.1 Tools

GraphViz

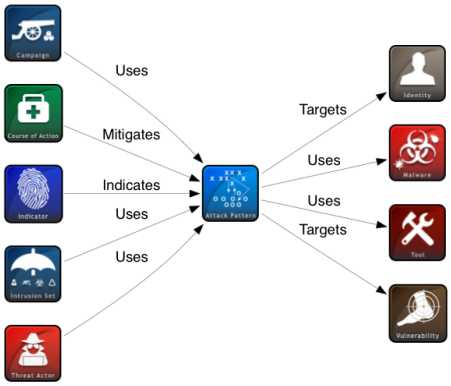
Graphviz [22] is open source graph visualization software that provides graph visualization for tools and web sites in domains such as software engineering, networking, databases, knowledge representation, and bio-informatics. Graph visualization is a technique for representing complex data structures using visual representations of nodes and edges. In this approach, nodes represent entities or objects, while edges represent the relationships or connections between them.

In the current applications context, a multitude of applications utilize Graphviz to create graph layouts that assist users in comprehending domain information and performing visual tasks more effectively. This is due to its layout programs which take descriptions of graphs in a simple text language, called DOT language, and make diagrams in useful formats, such as images and SVG for web pages, PDF or Postscript for inclusion in other documents, or display in an interactive graph browser. Graphviz has many useful features for diagrams, such as options for colors, fonts, tabular node layouts, line styles, hyperlinks, and custom shapes.

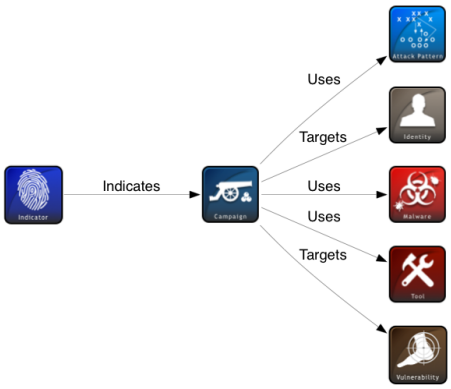
In particular the DOT language, that stands for “Graph Description Language”, is a simple text-based language used for describing graphs and other network structures and it was originally developed as part of the Graphviz software package. In DOT, graphs are described using a set of statements that define nodes, edges, and other graph properties. For example, a node might be defined using a statement like “node1 [label=“Node 1”]”, while an edge might be defined using a statement like “node1 ->node2”.

An interesting example of the use of GraphViz software in the field of cybersecurity is shown below (figure 4.1) in the representation of the STIX [23] language. The STIX initiative is a collaborative and community-driven effort aimed at defining and developing a structured language

for representing cyber threat information. The STIX Language is designed to capture the complete range of potential cybersecurity threats, and emphasizes expressiveness, flexibility, extensibility, and automatability. The language is also designed to be as human-readable as possible, in order to facilitate its widespread adoption and effective use in cybersecurity contexts.



(a) figure Attack Pattern representation



(b) Campaign representation

Figure 4.1: Examples of STIX model with GraphViz representation

Given its characteristics, it would be appropriate to categorize GraphViz as a generic tool which, thanks to its versatility, is able to make a contribution in cybersecurity contexts. The most positive feature probably lies in being the technical basis of several tools developed for security visualization. At the same time, however, this generality makes it limited to the use that is made of it and to the information that is to be represented, analyzed and transmitted. Other technical limitations that could be encountered using the tool are the lack of support for GUI editors, which in some environments could disadvantage its use and the maximum number of nodes and edges at 200 and 400 respectively in the DOT language version without extensions. which in cases of large-scale representations would in some cases result in a non-total representation. Complete documentation and various installation packages are available online [24].

Rumint

This tool takes its name from a term used in the intelligence community that stands for “Rumor Intelligence”. RUMINT [25] is an open source network and security visualization tool that uses a VCR-like interface to enable analyst to visually play back traffic from any point within the set of packets. The data used in input are in the pcap format, that stands for “packet capture”, a standard file format which stores network data in binary format, used in the field of network analysis.

After loading the data through the simple interface, the tool allows you to reproduce the transmission of the packets through seven different types of visualization, which are Text Rainfall, Byte Frequency, Parallel Plot, Binary Rainfall, Scatter Plot, Packet Detail and Combined, which can be activated individually and, allowing multiple simultaneous views for the analyst. Each visualization in turn allows, once opened, to choose between different options regarding the possible data to be used in the plot. For example, in the case of the scatter plot, it is possible to choose up to 19 different types of data, among which you can find the most common fields such as Source IP, Destination IP, TCP or UDP Source, TCP or UDP Destination, or more specific fields such as IP Header checksum, the TTL, or the Ethertype, if present within the string representing the packet.

In the screenshots shown, which show the various views of the tool, a pcap format file was used, containing the data of a spoofed TCP synflood attack [26]. To allow a better use of the images, in the screenshots the analyzes have been stopped at a few packets, in the order of tens and hundreds, to obtain images that can be shown clearly and allow a correct understanding in

the case of virtual reading mode and in case of printing. However, the deductions reported in the following lines, although they are already confirmed by these visualizations with a reduced quantity of packets, were made on the complete analysis of the sample that the tool allowed to load.

From some of the views produced by the tool (figures 4.2, 4.3, 4.7), we can see how the data flow contains a large amount of packets with similar content, which can in turn be inspected individually (figure 4.5). For example, through the Text Rainfall visualization, in this case setted to represent the data via ASCII characters, it can be seen that the packet header is the same for all. Through the other views (figures 4.4, 4.8) where it was chosen to use the starting and destination IP addresses as parameters of the coordinates instead, it can be seen that the destination of all the packets is unique, and then in the last case (figure 4.6), i.e. in the combined view, where instead chosen to represent the source and destination ports of the TCP service, it can be seen that here too the target is a single and specific port.

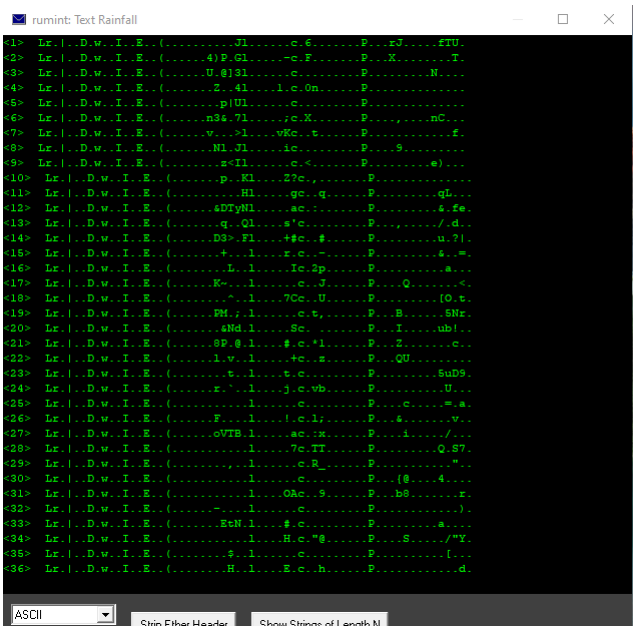


Figure 4.2: Text Rainfall

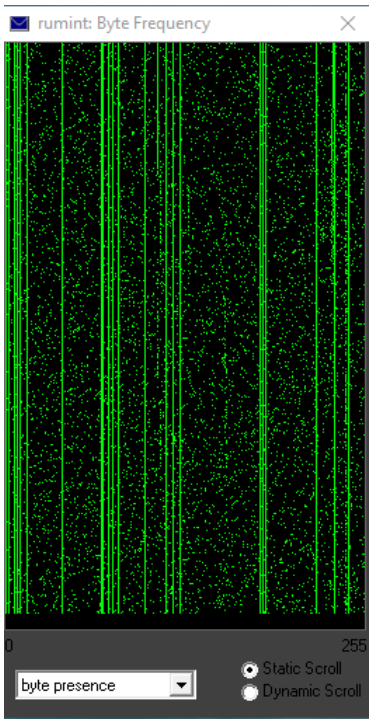


Figure 4.3: Byte Frequency

RUMINT is certainly one of the first examples of tools developed with a focus on security visualization, given that its first versions date back as far as 2005, years in which research focused on the use of more generic software. One of its main strengths lies in the possibility of simultaneously having all the different types of visualization that it offers independently in order to allow a varied and customizable use, together with the multitudes of data that can be chosen to represent. Another feature that facilitates its use is the absence of necessary configurations. The application only needs to be downloaded and once started it is ready to load the file, using the appropriate drop-down menu, and open the various views.

Figure 4.4: Parallel Coordinate Plot

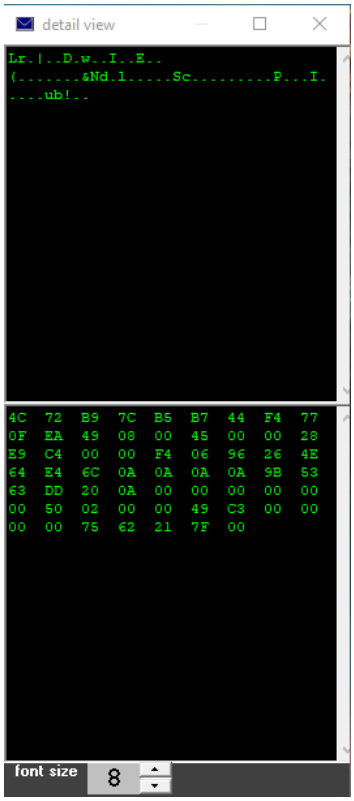


Figure 4.5: Packet Details

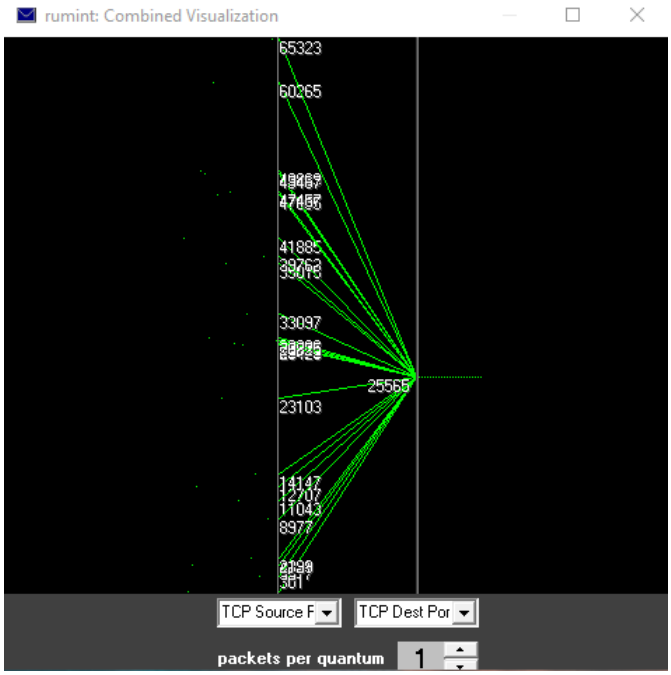


Figure 4.6: Combined Representation

However, his not young age is one of his main limitations. The latest update is dated September 2007 and as can be seen from the screens shown, the compatibility of the windows is not suitable for modern operating systems. Although the graphs do not undergo variations and are not influenced, the various optional settings that can be activated via buttons, the drop-down menus and some details are not correctly shown by the windows and therefore make their use more complicated

and in some cases not possible. From a more technical side, however, the greatest limitation is given by the capacity of the buffer that stores the packets limited to 30,000 packets in memory, which in some contexts may not be sufficient. Finally, the installation file is supplied only with the .exe extension, therefore available only for Windows operating systems.

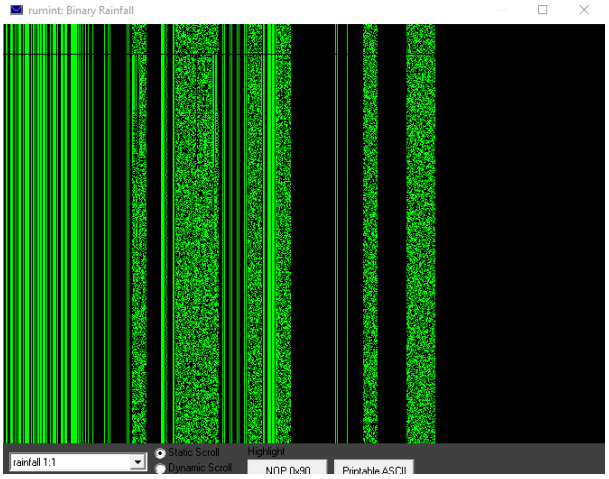


Figure 4.7: Binary Rainfall

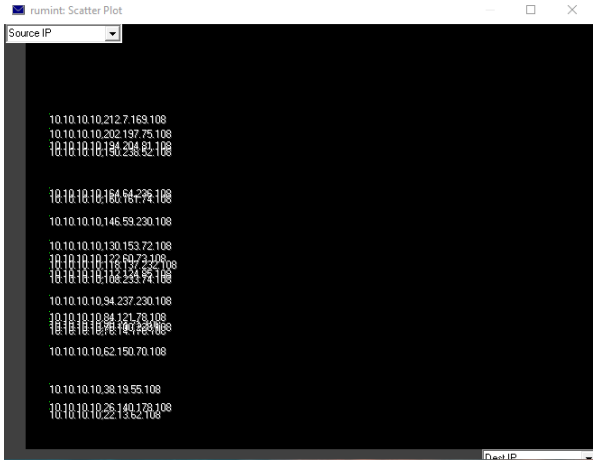


Figure 4.8: Scatter Plot

Tulip

Tulip [27] is a cross-platform, open-source framework designed primarily for the visualization and analysis of large graphs. It serves as an information visualization framework dedicated to exploring and visualizing relational data. Its primary objective is to offer developers a comprehensive library that supports the creation of interactive information visualization applications tailored to their specific problem domains. Tulip enables the implementation of algorithms, visual encoding techniques, interaction mechanisms, data models, and domain-specific visualizations, with a development pipeline which enhances the framework's efficiency for both research prototyping and the development of real world applications.

The tool not only allows importing CSV files containing various data for visualization but also supports importing files with different extensions representing other graph hierarchies such as .dot files mentioned in the previous tool, .bib files for BibTex, JSON files and more. In addition to the possibility of representing the data provided via file, the tool allows you to perform a variety of algorithms on the data such as clustering algorithms, topological test algorithms, triangulation algorithms and various algorithms for modifying the characteristics of the representation. Among the various possibilities for creating visualizations, the user can either select from a choice of more common graphs such as scatter plots, histograms and parallel coordinates, or choose from more specific types of graphs such as node link diagrams or geographic views.

The two examples shown in the images focus on this last type of graph. In the examples shown there are no data concerning the context of cybersecurity but generic data are reported in order to show the possibilities offered by the tool. In both visualizations the data contain, in addition to their information for analysis, data regarding their geographical position which was selected by the tool and produced the visualization. In the first case (figure 4.9), cities were represented, and in addition to their locations, temperature information was visualized using a color scheme. In a scenario more relevant to our context, for example, we could associate server data with their respective positions and add their status as additional information using a similar color scheme. This way, we would create an overview of the overall state of a network. In the second case (figure 4.10), we have a combination of a geographic representation and a node-link diagram. If we were to provide an example in this case as well, we could consider visualizing a network by including the various connections within it, without sacrificing the information about geographic location and status (or other relevant information based on the analysis), which can be achieved by using a color scale.

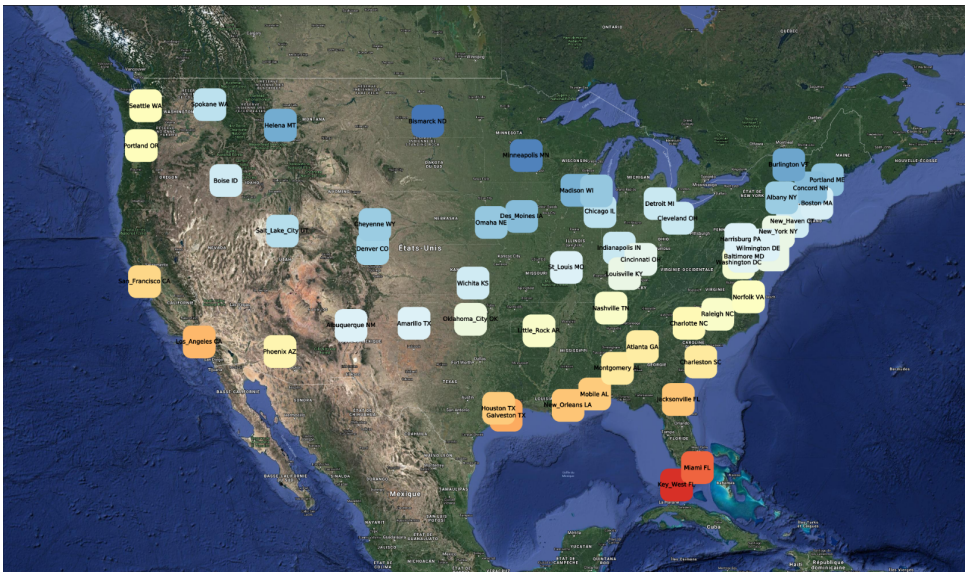


Figure 4.9: Tulip first example

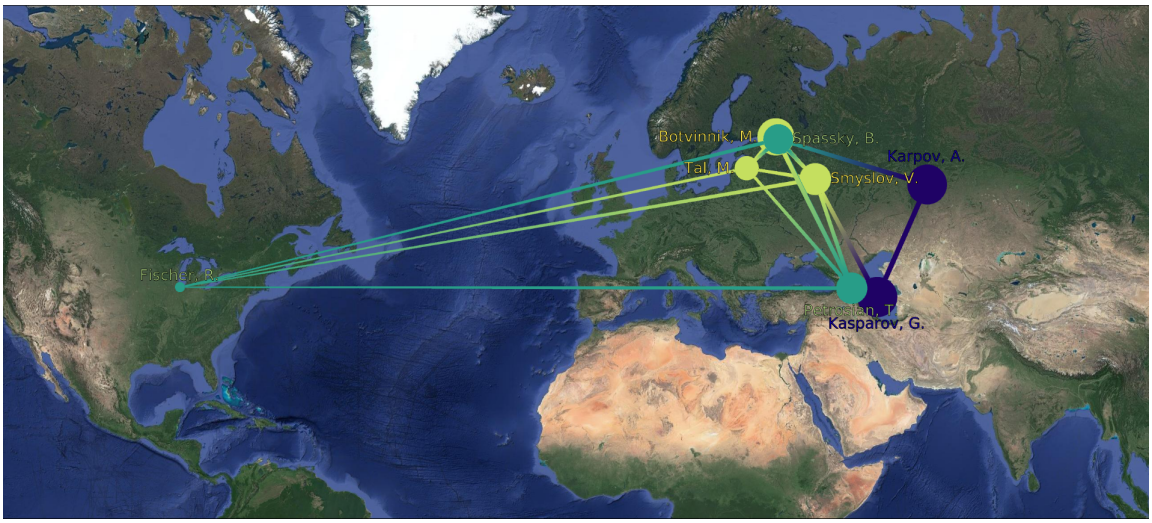


Figure 4.10: Tulip second example

The tool is available both as a file that can be installed on Windows [28] and as an open-source code [29] that can be installed on all operating systems and is also related to various guides available online. Also in this case we are faced with an application developed with a viewpoint focused on the representation of more generic data, and not for a cybersecurity specific focus. One of the fundamental elements that characterizes the tool is certainly the possibility of applying algorithms on the data both before and after the calculation of the visualization. The tool is also designed to be able to represent large graphs and is therefore optimized for the representation of large amounts of data. However, from the point of view of carrying out the analyses, a first limitation is the lack of interactivity of the graphs produced. In fact, once the image has been produced, it can be downloaded or modified for various uses but does not provide any interactivity, thus producing a simply image file.

Gephi

Gephi [30] is a comprehensive tool created to visualize, analyze, filter, and manipulate network graphs, taking into account both qualitative and quantitative variables present in the data. With Gephi, users have the ability to employ predefined layout algorithms to automatically arrange

nodes or manually position them as desired. Gephi utilizes advanced layout algorithms to determine the shape and arrangement of graphs. These state-of-the-art algorithms, designed for optimal efficiency and quality, are integrated within the software. The Layout palette within Gephi empowers users to modify layout settings in real-time, significantly enhancing user feedback and overall experience during the graph visualization process.

Furthermore, the software offers the capability to compute network metrics, especially for social network analysis (SNA) and scale-free networks, empowering users to leverage these analyses results to tailor the visual representation of their network graphs and facilitates the creation of high-quality vector graphics for network visualizations, ensuring their suitability for publication purposes. Additionally, the tool enables the exportation of filtered and analyzed networks in data formats suitable for interactive online display, such as PDF, PNG and SVG, employing JavaScript visualization libraries. The tool can read the majority of graph file formats but also supports CSV, relational databases import and spreadsheets, although the preferred format input file are surely GEXF and GraphML, two variants of XML used for storing and exchanging network graph data.

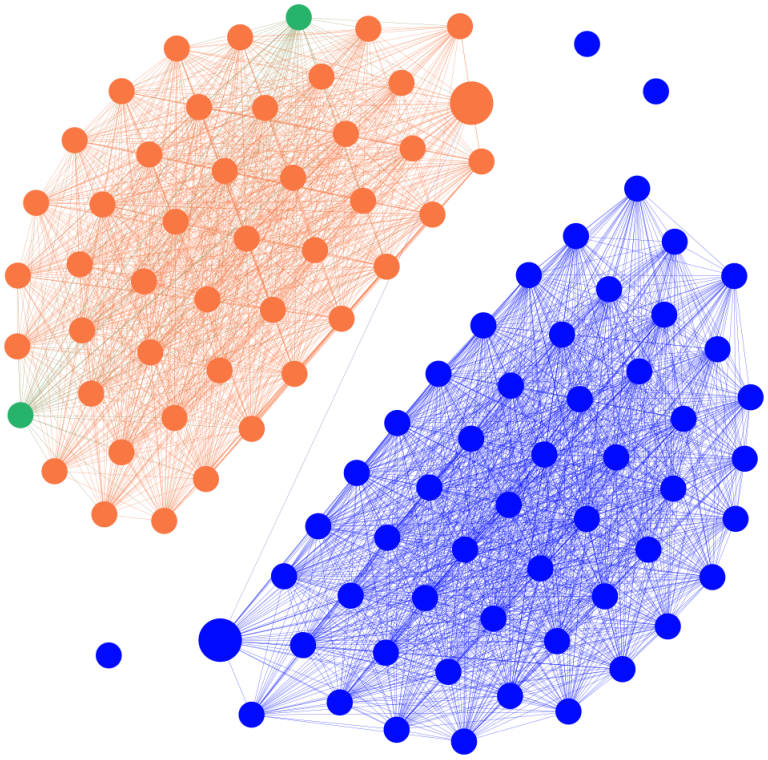


Figure 4.11: Gephi Fruchterman Reingold layout example

Also in this case, given the generality of the tool, the example shown in the figure (figure 4.11) uses as input a fictitious dataset representing two subnets and the respective nodes and messages exchanged. In this particular case the algorithm chosen for the representation of the layout is the Fruchterman Reingold [31]. This algorithm is a force-directed layout algorithm used to visually arrange nodes in a graph. It treats nodes as steel rings and connects them with springs representing edges. Attractive forces act like springs, while repulsive forces resemble electrical forces. The algorithm aims to minimize the system's energy by adjusting node positions and altering the forces between them. For further information, please refer to the Force Directed algorithm. After selecting the algorithm and waiting for the stabilization of the graph in the application, slight aesthetic changes were made such as the size of the nodes and the color of the networks, to allow a better view of the exported image.

As depicted in the image, the tool is particularly suitable for representing relational data, even with high density of connections, and one of its main features is the speed at which it calculates

these large networks. In addition to its basic functionality, the availability of a wide range of freely downloadable plugins extends the already extensive capabilities of the tool.The tool is freely available for download on its official website for all major operating systems. However, its broad interoperability does not make it a specific tool for the cybersecurity context. Furthermore, in this specific case, exporting the graph as an image format loses the interactivity that could characterize a functional view for an analyst.

CodePulse

Code Pulse [32] is a tool developed by Secure Decisions and OWASP that provides insight into the real-time code coverage of testing activities. Code Pulse automatically detects coverage information while the tests are being conducted and even makes it possible to understand the overlaps and boundaries of the code coverage achieved by different testing activities. The application allows you to analyze both Java and .NET projects through its passive monitoring conducted in parallel with manual or automatic tests. The main screen is divided into two different views, where the first on the left shows the structure and the related dependencies of the application, while the second on the right shows the coverage obtained from the tests through a TreeMap view of the classes.

TreeMap visualization is a widely used technique for representing hierarchical data in a compact manner. The technique involves organizing data into nested rectangles, with each rectangle representing a node in the hierarchy where the size and the color of each rectangle can be used to represent different attributes of the data, such as the size or importance of each node. The TreeMap visualization is particularly useful for visualizing large and complex datasets, as it allows users to quickly identify patterns and relationships in the data. The hierarchical structure of the visualization also makes it easy to explore the data at different levels of detail, from the overall structure of the hierarchy down to individual nodes and attributes. In Code Pulse's case, its displayed only selected content from the application inventory.

A demo application [33] developed by Microsoft known as Contoso University was used for the test. ContosoU, as it is named in the screenshots, is a sample application that demonstrates how to use Entity Framework Core in an ASP.NET Core MVC web application. For the first part of the analysis, the project zip archive was uploaded to the application and the structure skeleton was rendered without providing further information. The next step was to connect the project application running to Code Pulse, in this case via the Code Pulse .NET Tracer. In this case the test was conducted only in manual mode, browsing through the various pages of the web application. In the first image (figure 4.12) can be seen the initial phase, where only a few pages were visited and only a few rectangles representing the coverage were covered. In the second (figure 4.13), more pages were visited, more courses were added and removed calling various APIs, and the relative test cases were updated in the TreeMap view. As shown the mouse cursor can be used for display information about the classes and in the image it's used for inspect the class that has not been triggered by tests.

One of the main strengths of Code Pulse is real-time analysis, since most penetration testing tools typically evaluate and present results at the conclusion of a testing run, rather than while the tester is immersed in the testing activity, which also results in reduced analysis times and consequently on the related costs. The rapidity of understanding the analysis is certainly also given by the approach used to show the coverage. In this case the choice, not only to use a visualization method, but to select a non-classical methodology such as the TreeMap has made it possible to obtain an impact on the communication and understanding of the data which in most of the tools are represented in a textual or indicated with simple symbols. However, despite its many advantages, the TreeMap visualization does have some limitations. For example, the technique can be difficult to interpret for users who are not familiar with hierarchical data structures, and it may be less effective for visualizing non-hierarchical data. One big downside of the application lies in its initial setup configuration. If on the one hand, once correctly activated, the tool works in a mode that we can define as semi-automatic, this comes at the expense of the connection phase between the application and the Code Pulse tracer which, depending on the type of application, can be more or less complex depending on the type of application you are working on.

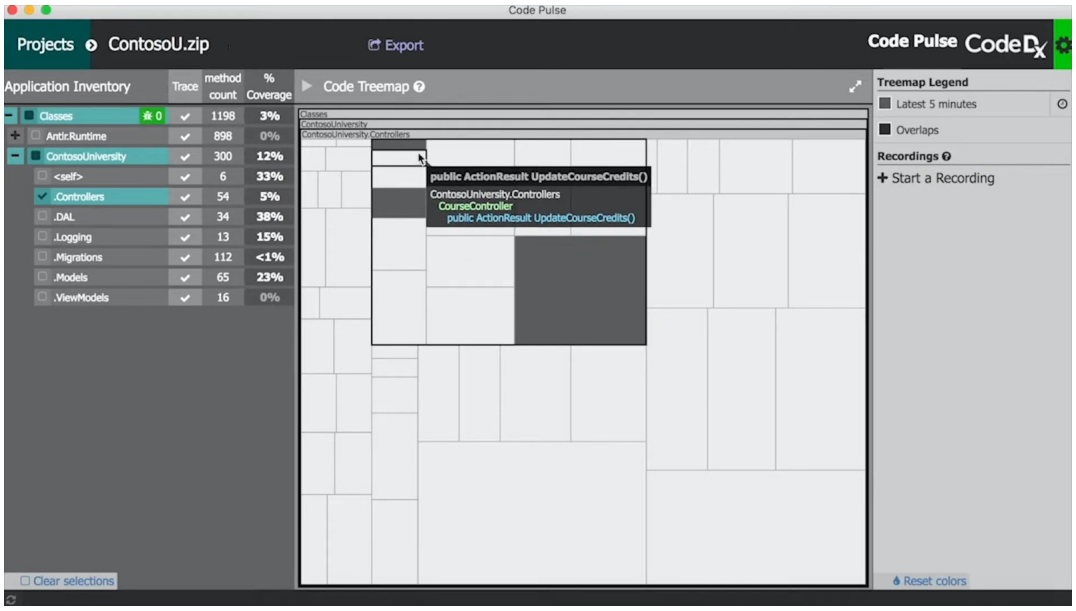


Figure 4.12: Code Pulse

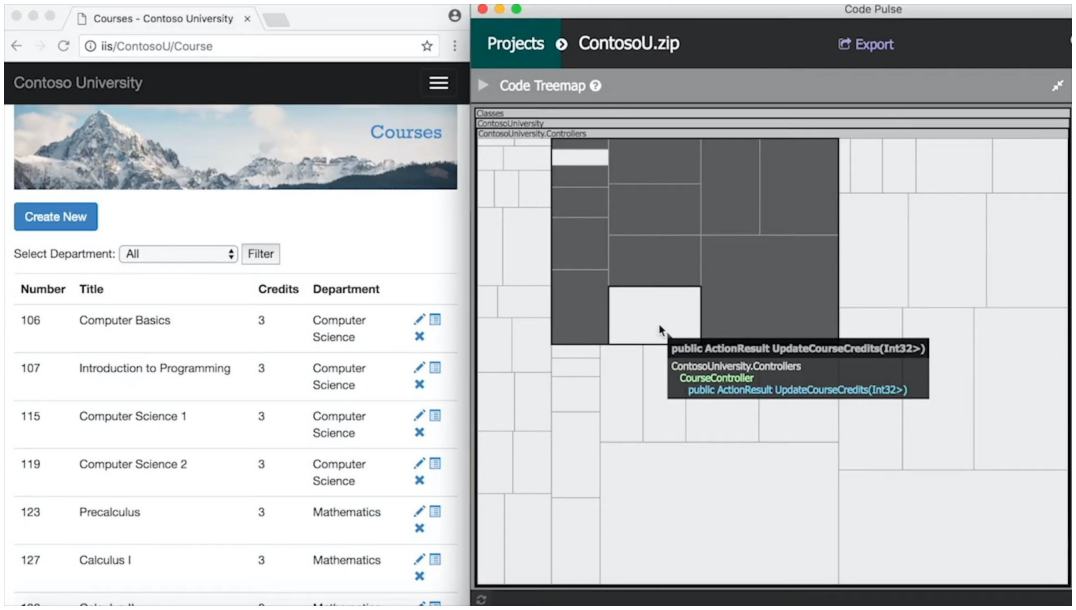


Figure 4.13: Code Pulse and the Application running

Thanks to his research paper [34] we can make a first comparison between a developed solution and the framework proposed by Balakrishnan and mentioned in the chapter 3.1. For the first phase, i.e. defining the *Visualization Goals*, the developers immediately clarify how there were two primary concerns in their objectives: how best to identify the coverage data; and how best to communicate it to the users. For the next phase of *Data Preparation* no specific information is provided but an evaluation that was conducted by the developers concerns the disparity of two different points of view which are respectively the analysis of the application's source code and the analysis of the sitemap and its corresponding URL-set, which do not always translate directly to each other. In this phase, therefore, it was decided to concentrate on the analysis of the source code, postponing the other type of analysis to a future version.

Subsequently, in the paper not enough information is provided on the selection process of choosing the TreeMap as a visualization methodology, so in this case we can only assume that the *Exploration* phase occurred due to the targeted and uncommon choice, but there is no evidence.

As far as the *Visualization* phase is concerned, the only possibility of modifications concerns only the color that the various rectangles could have, given that the other characteristics such as shapes and dimensions are dictated by the logic with which the TreeMap is developed. In this case it was decided that the different analyzes had different colors, with the possibility of personalization left to the user and consequently also allowing a more direct comparison between the methodologies during use, leaving only white as the standard colour, for the cases in which the tests are not covered by the coverage, and gray, for the case in which the tests are covered by the single coverage in the case of a single analysis or by all the coverages in the case of multiple analyses. No information is provided for the final stage of *Feedback and fine-tune*. However, analyzing the final part of the conclusions and the current state of the tool, we can state that the tool has evolved allowing the analysis not only of Java projects but also of .NET projects as described above, resulting in improvement and update of the application. Thanks to this feedback, we can see how much this last phase is the most uncertain given the possible and varied developments that the various applications may undergo due to their different typology.

Pixel Carpet

Pixel Carpet is a result from a collaboration between data visualization researchers and computer security engineers. The idea behind this collaboration is to take the best knowledge from both fields to develop through a user oriented approach a solution to represent multi-dimensional data sets, specifically log files. The types of logs for which the application was designed are SSH logs, given that they are a crucial part in the post-attack analysis, and Apache access logs, to allow identifying and understanding possible traces of security breaches. The main purpose of pixel-oriented visualization techniques is to represent as many data objects as possible on the screen at the same time by mapping each data value to a pixel of the screen and arranging the pixels in a suitable way [35].

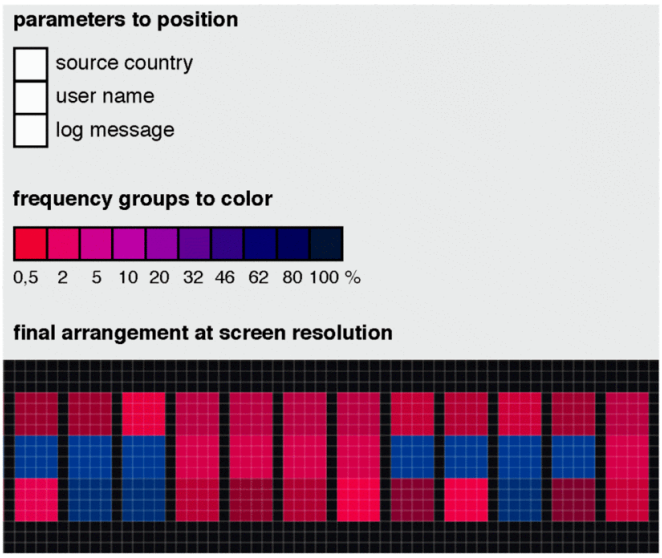


Figure 4.14: Pixel Carpet multipixels construction [36]

Among the various possibilities, the developers have chosen to use a pixel map in order to graphically represent parameters such as IP, source country and request or message, grouped together in a single column (figure 4.14). In this way there are many small columns, each representing a row of the file, which produce a denser visualization and allow the patterns to be recognized. Besides the subdivision in pixels, the most important parameter is the colouring. Within the application there are various shades from which it is possible to choose but the most important concept is that the range of colors and not fixed colors has been chosen to represent the frequency that that single entry has within the log file. So, for example, the pixel corresponding to the country will be of a color that represents a higher percentage, the more that entry is present in the inserted file, as well as for the IP address and the message or request.

Although the type of data chosen is widespread on a large scale, the requested input file is in .json format and not .csv or .log and therefore a conversion process is required before being able to use the visualization. Furthermore, the required JSON formatting is specific (4.1) and a different formatting causes the script not to work.

Code Listing 4.1: JSON format example

```
{
    "Host": "200.120.190.33",
    "Log Name": "-",
    "Time": "23:18:40",
    "Time Zone": "GMT+0100",
    "Method": "GET",
    "URL": "/wp-content/uploads/the-legend-of-zelda-skyward-sword-boxart.jpg",
    "ResponseCode": 200,
    "Bytes Sent": 217935,
    "Referer":
        "http://ba-k.com/threads/3605485-MEGA-POST-JUEGOS-DE-WII-POR-TORRENT",
    "UserAgent": "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML,
        like Gecko) Chrome/32.0.1700.107 Safari/537.36",
    "Country": "Chile"
}
```

Unfortunately, given the nature of the research project, the tool should be considered a prototype. The only ways available to try its features are through the demo available online [37] and by downloading the source code [38] and then launching it on your machine. The differences between the two versions, demo and offline, are minimal and are mainly a different arrangement within the menu, and for the demo version a smaller choice of options for the color range. However, having the source code available allows you to modify the input files that the tool must display. In addition to the code for the tool, the developers provide two log files, one with geographic data and one without, containing an attack attempt, to provide an overview of the use of the application. The other types of logs with which it was developed and tested have not been made available for privacy reasons.

Using the log file containing the suspicious behavior, the display below is reproduced (figure 4.15). From a first glance, the patterns present among the various requests present in the file are evident. Trying to analyze the longer pattern below, we can see how the series of requests highlighted in a similar way and close in time can be traced back to a series of requests, or to an attempt of Local File Inclusion attack. LFI is an attack technique in which attackers trick a web application into either running or exposing files on a web server. The following image (figure 4.16) shows the various options that can be activated via the Controls menu. For example, in the proposed image it was chosen to mark the entries with the same URL as the selected one, all highlighted by the green box, and not to apply any filter. Furthermore, by opening the Key menu it is possible to have a better view of the graphical representation of the selected entry and to compare it alongside the frequencies within the file through the gradation of the colors represented.

Given the nature of a prototype, unfortunately there are many limitations. The tool is not optimized and there are no specifications regarding the maximum supported size of the files to be analysed, even if with a simple test conducted with a generic 30MB file it was found that the current version does not allow the start of the visualization. Then, in the case of readable files but in any case with a large number of entries, the visualization could not be easy to understand due to the large amount of data to be represented. All these assessments are a consequence of the fact that the main purpose of the research was not to develop a product but to demonstrate the validity of the visualization approach which can turn raw unlabeled data into an image that makes easier to survey and inspect.

Figure 4.15: Pixel Carpet usage example

Figure 4.16: Pixel Carpet Menu example

From a development analysis point of view, given the mode of cooperation between teams of different skills, in the *Visualization Goals* definition phase we can see a combination of different elements such as interviews, single observations and an ideation workshop. All these different interactions have led to the definition of well-defined objectives including, for example, Display raw data, and Encourage explorative analysis. Subsequently, for the *Data Preparation*, the files already in their possession are mentioned due to a previous analysis but no technical details are provided on the cleaning and conversion part of these data. In their *Exploration* phase the developers have not limited themselves to a theoretical research [35], provided by publications on pixel representation, but have integrated the ideas obtained through the group workshops reaching the conclusion of opting for a solution that would allow them to have the best "data to space ratios" by using the smallest building block on a computer screen. Within their paper, a large space is dedicated to the corresponding *Visualization* phase and the choices that have been made. Summarizing them in this brief summary would not be sufficient, therefore, for further information, please refer to their publication. However, the constant presence in the choices, such as the mapping of colors and related patterns or filtering and highlighting effects, of the objectives set in the previous phases in a context of cooperative work should be mentioned. Finally, also for the final part of the process, identifiable and synonymous with the *feedback and fine-tune* phase, a great effort is made in the cooperation area. Several groups of different organizations, again with different skills and knowledge, were gathered to get feedback about the prototype, producing a positive result as the prototype and its visualization was generally evaluated as valid and suited for the inspections of log files.

Visual Decision-Support for Live Digital Forensics

The tool presented in this section is a prototype developed in the context of forensic analysis. Specifically the prototype collects relevant core information for live digital forensics and provides visual representations for connections between occurring events, developments over time, and detailed information on specific events [39]. The motivation behind the development lies in the need to provide forensic experts with new methods for real-time analysis to support their decision-making process. In the case of forensic analysis, the choice of data to be analyzed is crucial as they can be volatile or non-volatile, raw or pre-processed data or data that can have an impact on the running system.

The choice adopted by the developers was to consider as relevant data such as, system Logs, Process information, Network traffic, File-system activities and copy of modified files. The prototype state of the application does not allow for the uploading of personal data, the only option therefore is to test the application with the data provided within the available code [40]. From the research paper we can deduce that the data was extracted from an Apple mobile device running iOS 14.5 and that the data extraction was conducted at an earlier stage. Specifically, the data provided within the code is artificial data created with the aim of simulating a particular use case [41], based on a real investigation report, to provide a potential scenario for the prototype and to show a possible use of the tool during forensic investigation activities.

By downloading and starting both the server component and the front-end part of the available code, the screen shown is rendered (figure 4.17). We can immediately see how the tool comes in the form of a single-page application with the dashboard divided into various views. On the left side we can find the search parameters and interactive filters that can possibly be applied to the visualizations, thus modifying the displayed diagram both by changing the start and end times for the analysis and by including or not the filters. Note that when viewing the filters they act both as buttons, which can be activated or not, but also as a legend. At the top is shown a Bar chart (figure 4.18) for an overview of all activities with a specific time range brushed, allowing users to quickly perceive when the device has conspicuously high or very low activity, allowing specific selection within the timeline. In the central part of the page instead we find the node link diagram which provides us with insight into the activities of the device during the time window selected in the Overview, if it is selected, or the entire time interval that has been provided within the search parameters. The peculiarity of this view lies in showing different types of nodes at the same time, and at the same time showing the respective links, i.e. the activities that that node has performed, providing a visual indicator of how active or inactive they are, depending on the

analysis. Finally, the right side presents three graphs, two of which are scatter plots and one a bar chart, which provide information about network and file system activities and information about file versions. In case a node is selected from the node diagram, this part of the dashboard changes to provide information about the selected node, as shown (figure 4.19).

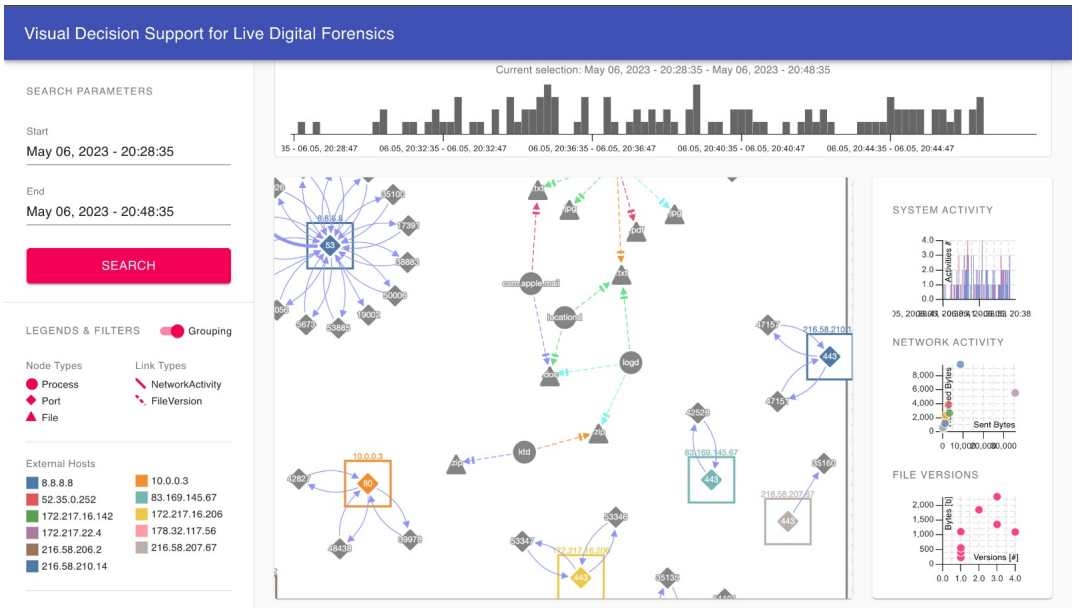


Figure 4.17: The rendered page of the application

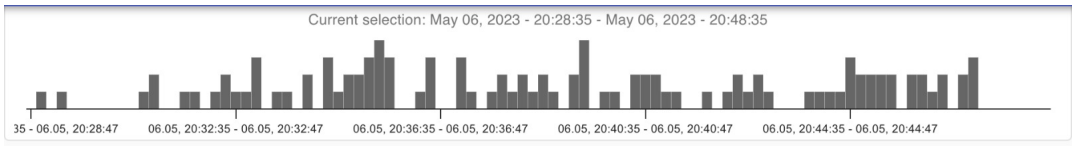


Figure 4.18: Bar chart

In terms of development phases it is clear that the main goal of the development was to provide a tool that was a support for accelerate and improve decision-making for domain experts. From the point of view of data selection we cannot say much since the data used to produce the prototype were artificially produced, even if using a real use case as a reference. As regards the cleaning and conversion phase, the program pipeline correlates the raw data provided. The main reason for this is given by the need to obtain representable connections between processes, file modifications and network communications, not allowed by the simple reading of raw data. No particular information is provided on the process of choosing the display method, however we can prove that one of the main characteristics sought in this case is the interactivity of the display to be shown. The characteristics of the visualizations have been presented previously. However, an important observation is that this is the first case addressed in this work which presents a combination of multiple visualization methodologies, all aimed at a single purpose but each providing a different type of information. All this within a single dashboard without the need to navigate to different pages, like the most modern analysis solutions also provided in other sectors. For the final part of the process, i.e. *feedback and fine-tuning*, we know that the prototype was integrated in a workshop dedicated to forensic professionals and that the results would be used to extend the work produced. To date, however, there are no updates found in new versions of the tool.

The prototype developed and shown in this section is another example of how visualization in the field of cybersecurity has a broader response, allowing it to also range in fields such as forensic analysis. However, as suggested by the authors themselves, the tool should be considered as a tool that supports the initial selection of more specific forensic tools. Considering possible developments for the prototype, useful for inserting it in a real context, it would be interesting to

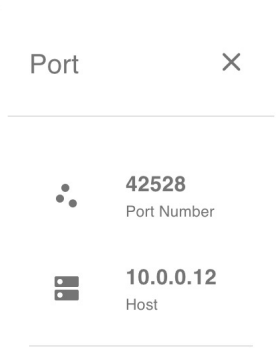


Figure 4.19: Node details

add the possibility of inserting the various data necessary for the analysis through the application itself, so as to be able to change datasets during the same analysis. From a graphic design point of view, however, a possible limitation is given by the choice of the node-link diagram that tends to display only a "hairball" when the number of nodes is high or when the nodes are highly interconnected, therefore not suitable for certain analyses.

NetCapVis

NetCapVis [42] is a prototype developed with the aim of providing analysis, through a visualization interface, of PCAP data. The main idea behind it is to provide an alternative to Wireshark [43] for the analysis of this type of data, by supporting the user in finding points of interest more easily and in less time. However, in its current prototype state, the developers consider the tool an interactive visual filter prior to delving into details with Wireshark. The prototype was developed following a client-server infrastructure, thus providing the user with a web service. One of the main features of NetCapVis is the use of progressive data processing. Progressive data processing is a technique used in data analysis and visualization that allows the user to interactively refine the results as more data is loaded and processed. For example, instead of waiting for all data to be processed before presenting the results, progressive data processing provides an iterative and incremental approach where partial results are displayed and refined as more data becomes available, allowing a faster data exploration and analysis.

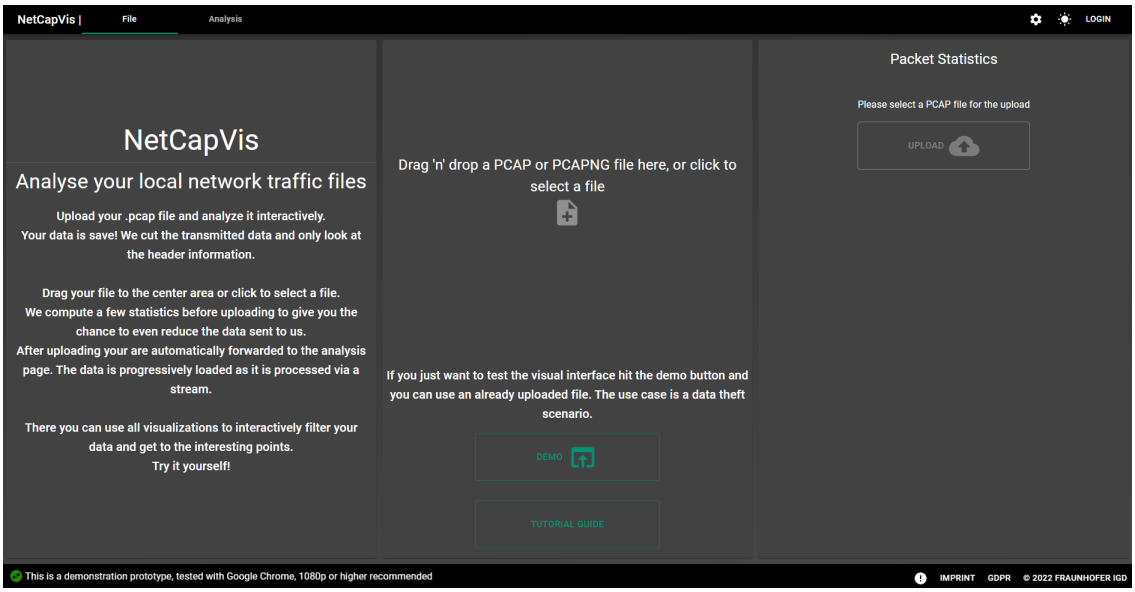
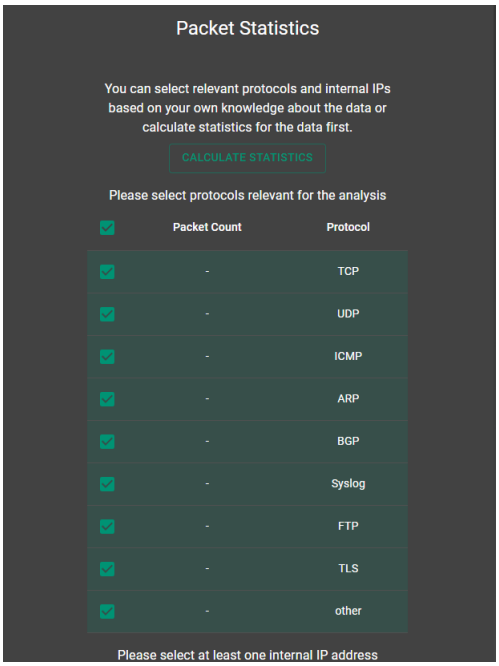


Figure 4.20: File page of the application

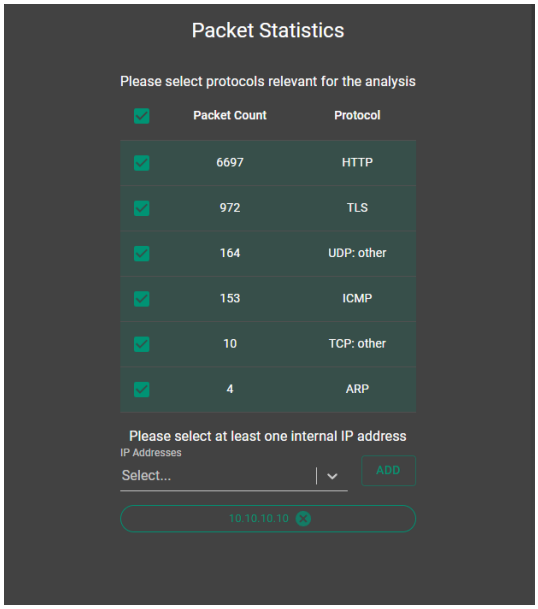
Figure 4.21: Analysis page of the application with demo dataset

To showcase its features, an online version of the prototype is available [44]. On the file page (figure 4.20), users can choose to proceed with the demo dataset or upload their own PCAP file (figure 4.22), which is limited to a size of 1MB. Starting from the example data provided in the demo, the analysis page is obtained (figure 4.21). This page contains several visualizations, divided into different areas of the page. In the upper half of the page there is the timeline visualization which allows to inspect the volume, in bytes, of the packets over time. This visualization is split in two separated parts, the timeline context which provide an overview over the whole time span contained in the PCAP data, and the timeline focus, that shows the zoomed version of the brushed area in the timeline context view. The first one can be used for filtering based on time intervals, and the second one shows additional details and information about the raw packet. On the left of the page we can find a bar chart representing the protocols stats, where we can choose to switch between the packet's number count and the volume of the packets, divided by the protocol type. This can be used for differentiate between the most know protocols and for filter packets for specific protocols. Right above this representation there a single stacked bar chart, which represents how many connections are incoming and outgoing. Its usefulness lies not only in transmitting the information and the proportion between the two data, but as with all interactive visualizations on the page, it allows you to filter based on what is selected. In the lower central part a graph view of the connections is shown, to allow grasp interesting patterns or connections in the network at a glance. This views also permits to switch between two representations, IP addresses or Ports. The last two representations are also bar charts. They are located in the lower right corner and represents the source and destination visualization. They can be also switched from representing IPs and ports and they can help to identify attacks that focus on opening large numbers of connections to the same IP or identify sources that scan for ports of a specific connection. The last two components that compose the page structure are not actual views but are a simple list of active filters, filters that can also be imported using the Wireshark syntax, using the appropriate button, and lastly a button that shows the raw data contained in the package.

As highlighted by the results of the tests conducted by the developers themselves, the prototype should be considered targeted towards experts in this field rather than simple users. As previously mentioned, it should be seen as a support or pre-filter for more specific analyses conducted through more concrete tools. Graphically, the chosen representations may be limited to certain types of analysis, disadvantaging others. For example, by importing a PCAP file [26] containing a TCP reflection attack, it can be noted that not all the produced visualizations are suitable (figure 4.23). Regarding PCAP files, it is not always possible to limit the size of the analysis to 1MB of file, which is why it would be necessary to increase the size of the importable files.



(a) File uploaded



(b) File uploaded and statistics calculated

Figure 4.22: Upload options

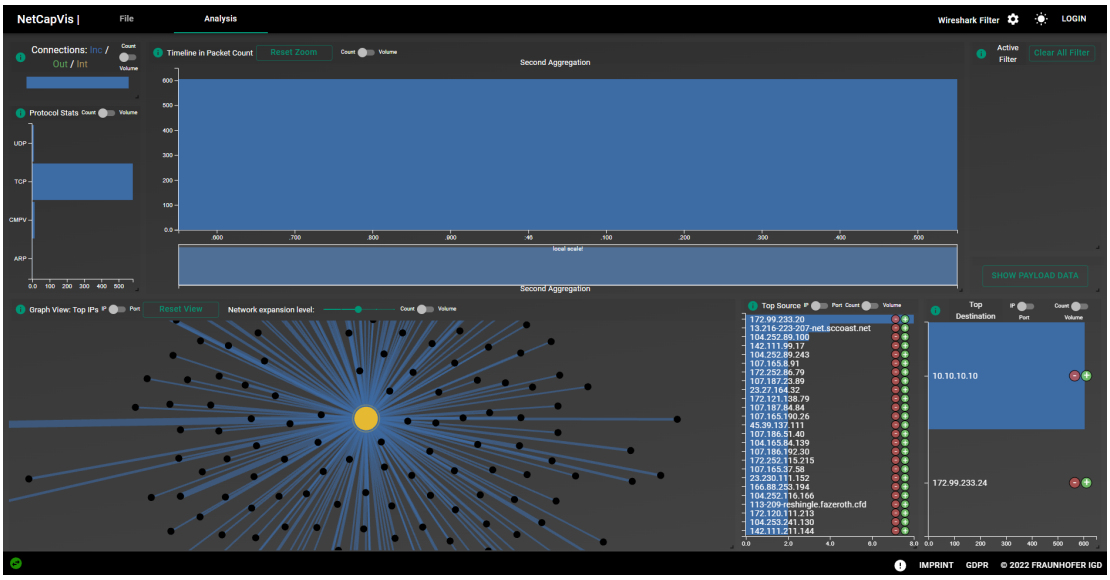


Figure 4.23: Analysis page of the application with imported PCAP file

Regarding the development of the prototype, the various phases can be described as follows. Unlike other developments that we have seen previously, the first information provided about it can be attributed to the *Data Preparation* phase. Details about the sources of the data used are not provided, but an important examination is carried out on the various processes that the data undergoes before being visualized. One of these processes is the pre-upload filtering (figure), which allows for a reduction of packets that are not relevant to the analysis even before they are sent to the engine that will render them. Immediately after, for the *Visualization Goals* definition part, the main objectives that the prototype must have are defined through a use-case driven approach. Among the main objectives, we can mention, for example, Find suspicious connections in large packet captures or Determine relevant events to filter for in Wireshark, also highlighting here how, after an in-depth analysis, the initial goal of providing an alternative to Wireshark has evolved

into an objective that emphasizes the correlation between the two and the need for the prototype to have a second tool for further analysis. In this case, no information is provided regarding the search and eventual selection corresponding to the *Exploration* and *Visualization* phases. Although for the second phase, there was a brief description in the previous section, for a more in-depth analysis, the related paper is referred to. Finally, regarding the *Feedback and fine-tune* phase, in this case, surveys were conducted, both with non-specialized users without experience in network traffic and with specialized users. From the results obtained from both students and experts, we can note how the prototype has evolved, both in terms of functionality, such as the possibility of seeing the packet list shown before, and in terms of user experience, introducing, for example, the possibility of conducting a tutorial during the use of the tool. Unfortunately, it is not possible to analyze in what ways and at what timing any "tuning" was made, as no information is available regarding the code or version of the prototype.

RopMate

Another field in which security visualization can provide its support is undoubtedly red teaming. In this specific case, a solution called ROPMate [45] is presented, targeting ROP exploit builders. Return-oriented programming (ROP) is one of the main "code reuse" techniques used in cases where security measures are implemented to defend against code injection attacks. In the specific case of ROP the attacker leverages existing code sequences called "gadgets" to manipulate the program's control flow. By chaining these gadgets together, the attacker can bypass security measures and execute malicious actions. The purpose of the research and the developed prototype [46] aims to introduce a support that allows assistance in constructing the rope chain, or in other words the ROP gadgets sequence, which is still predominantly performed manually. Through the GitHub page of the prototype, it is possible to download and subsequently run the server on which the application is rendered. In addition to the prototype code, the developers have provided test files that can be used to perform operations and test the tool. Further details regarding the type of input files and their production will be discussed later, following the presentation of the various visualizations.

After downloading the application code and installing the necessary requirements for its functionalities, it is possible to launch the server and access the tool through a browser. Once again, the tool consists of a single-page application (figure 4.24) that displays various types of visualizations, divided based on their usage, within its interface. In the upper section, we immediately notice the various options for applying filters, modifying the memory threshold, and the button that allows uploading a new file for analysis. On the left-hand side, it is possible to select different registers to further filter the various gadgets. Moving to the main part of the visualization, we find the Tree View, which represents the list of various analyzed gadgets. They are organized hierarchically based on the type of operations and classes. Each gadget is accompanied by a corresponding bar located on its right side, representing the memory space it requires. The bar is colored red if it exceeds the set threshold, and green if it does not. In addition to the information about the required memory, on the corresponding line of each gadget, we can find the dereferenced registers matrix. This matrix indicates which registers are dereferenced from the corresponding gadget. Each register is represented by a column. If the dependency is not satisfied, the column is filled to its full height. If the dependency is satisfied, the column is filled halfway. In the rightmost section (figure 4.26), there is a box that provides various information about the gadget. It includes the gadget's assembly code, memory address, registers it modifies, and registers it dereferences. By clicking on a gadget in the box, it is added to the rop chain. Just below, we find the similarity pane which, when a gadget is selected, displays all the gadgets that perform the same operation. Finally, below the Tree View visualization, we have the displayed rop chain (figure 4.25), where gadgets are represented by movable rectangles to modify the chain. Additional information is shown both inside and below each gadget rectangle. An example of Python code, generated from the chain shown in the figure, calculated using the "dump" button located above the chain, is provided below 4.2.

In the discussion of the various development phases, limited information is provided about each individual phase. Therefore, we will now summarize only two phases that refer to the model mentioned in previous chapters. Regarding the *Data Preparation* phase, it can be stated that

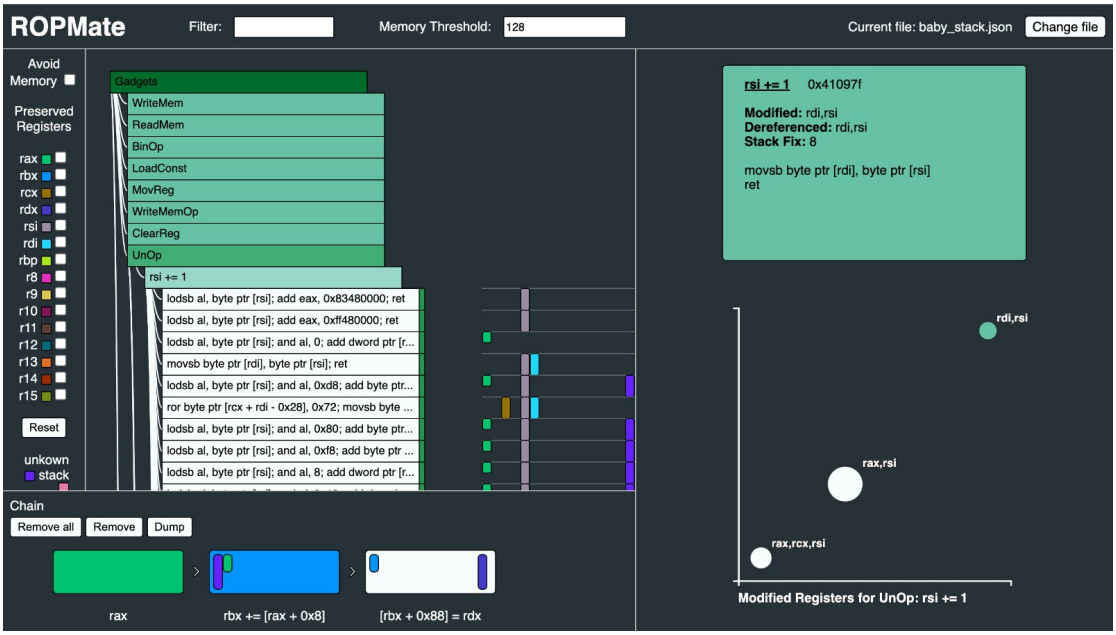


Figure 4.24: ROPMate dashboard

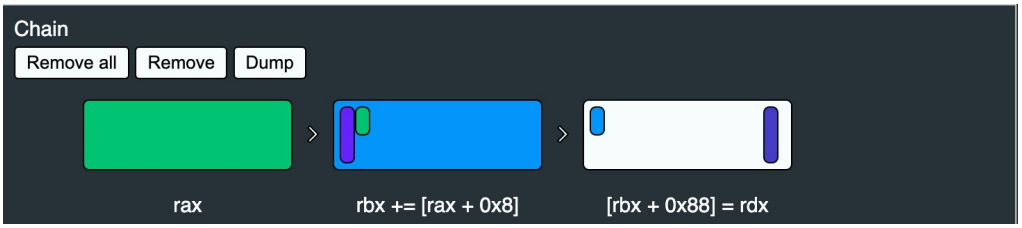


Figure 4.25: ROPMate chain builder

the data provided to the application consists of gadgets obtained through a tool developed by the same developers of the tool we are analyzing. This tool takes a binary file as input and produces its corresponding gadgets, which are listed in a JSON format. The data are therefore processed by the ROP-gadget semantics analyzer and subsequently fed into ROPMate to obtain the visualizations mentioned earlier. Regarding the *Feedback and fine-tune* phase, expert user evaluations were conducted, and the paper provides a visualization of the tool before this phase to demonstrate the developments and improvements that have been made based on the feedback of ethical hackers. The general feedback was positive, with testers considering the tool "very useful and user-friendly". However, the development appears to be stagnant at the moment, given the dates of the latest updates available on the GitHub repository.

Code Listing 4.2: Python code of the ROP chain

```
IMAGE_BASE = 0x0
rebase = lambda x : p64(x + IMAGE_BASE)

rop = ''
rop += rebase(0x4016ea) #pop rax; ret
rop += p64(0x0)
rop += rebase(0x4942df) #mov rbp, qword ptr [rax + 8]; add rbx, rbp; mov
    qword ptr [rsp + 0x10], rbx; ret
rop += rebase(0x44b4c8) #mov qword ptr [rbx + 0x88], rdx; add rsp, 0x50; ret
rop += p64(0x0)
rop += p64(0x0)
rop += p64(0x0)
rop += p64(0x0)
```

```
rop += p64(0x0)
rop += p64(0x0)
rop += p64(0x0)
rop += p64(0x0)
rop += p64(0x0)
rop += p64(0x0)
```

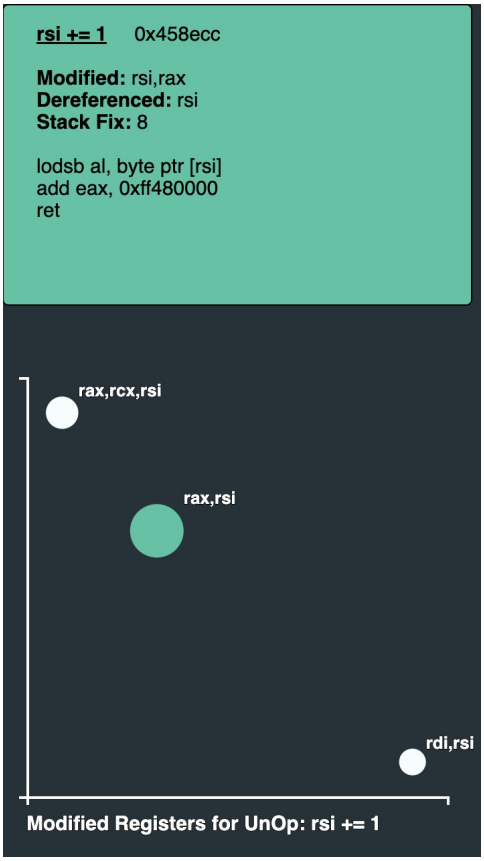


Figure 4.26: ROPMate Analysis pane and Similarity Pane

Its major strength lies in being one of the few, if not the only, alternatives to the traditional tools for constructing rop chains, particularly by utilizing visualization as support for gadget selection. Another strong point is the usability of the tool, which gives the impression of a complete application rather than a prototype. On the other hand, the greatest limitation is the requirement to use the ROPDaemon [47] tool, also developed by the same developers. This limitation not only restricts its compatibility with other existing tools but also compromises usability, as the version available on the ROPDaemon repository is currently unusable due to issues with the provided code. Therefore, no example test is provided in this section, as it would be similar to the one presented in the original ROPMate paper. For a comprehensive example, I would recommend referring to the original paper for a complete demonstration [45].

RedEye

RedEye is an interactive open-source analytic tool, released by CISA [48] (Cybersecurity and Infrastructure Security Agency) and DOE's Pacific Northwest National Laboratory, to assist Red Teams and Blue Teams with visualizing and reporting command and control activities. RedEye is capable of parsing logs from attack frameworks (at the moment only Cobalt Strike framework's logs) to present complex data in a more comprehensible format. This enables operators to assess and display intricate information, evaluate mitigation strategies, and enhance decision-making

processes. The tool allows users to upload campaign data to view relevant information such as beacons and commands.

The historical records of any campaign logs hosted on RedEye can be viewed at the graphical level associated with connected servers and hosts. Analysts can also examine key events in selected campaigns to identify payload activity and track attacker access paths, such as lateral movement activity or the use of credentials to escalate privileges on a device. Using data from analysts and the techniques used during the campaign, RedEye can also create presentations that can be shared with stakeholders and customers. All data collected from campaigns and analyst cases can be exported for customers. Blue teams can also use RedEye to quickly understand raw data from analytics, and identify attack methods and compromised hosts in order to do the right thing.

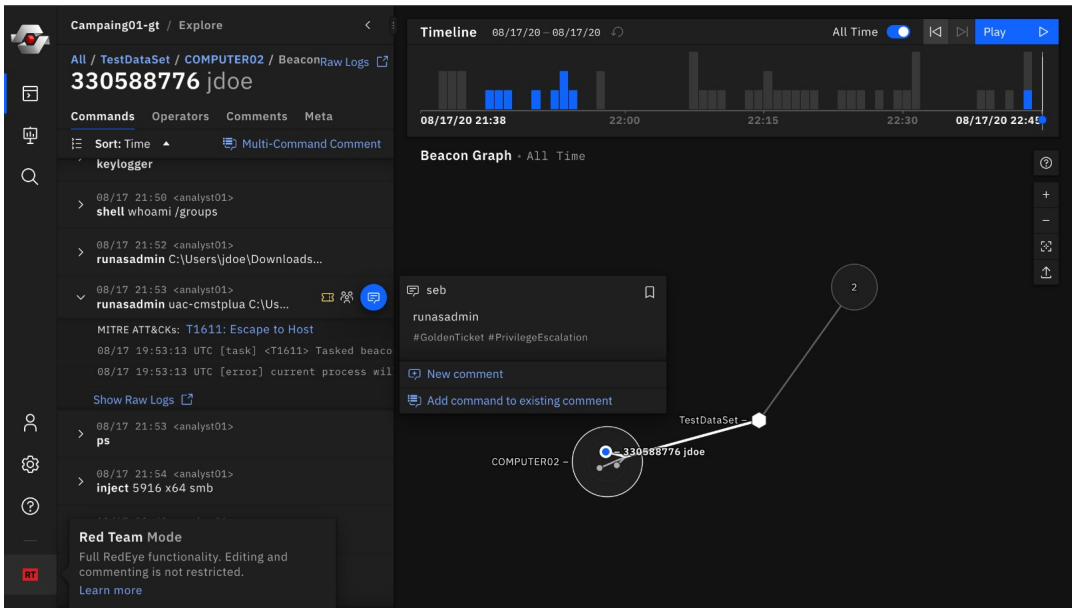


Figure 4.27: RedEye Red Team version

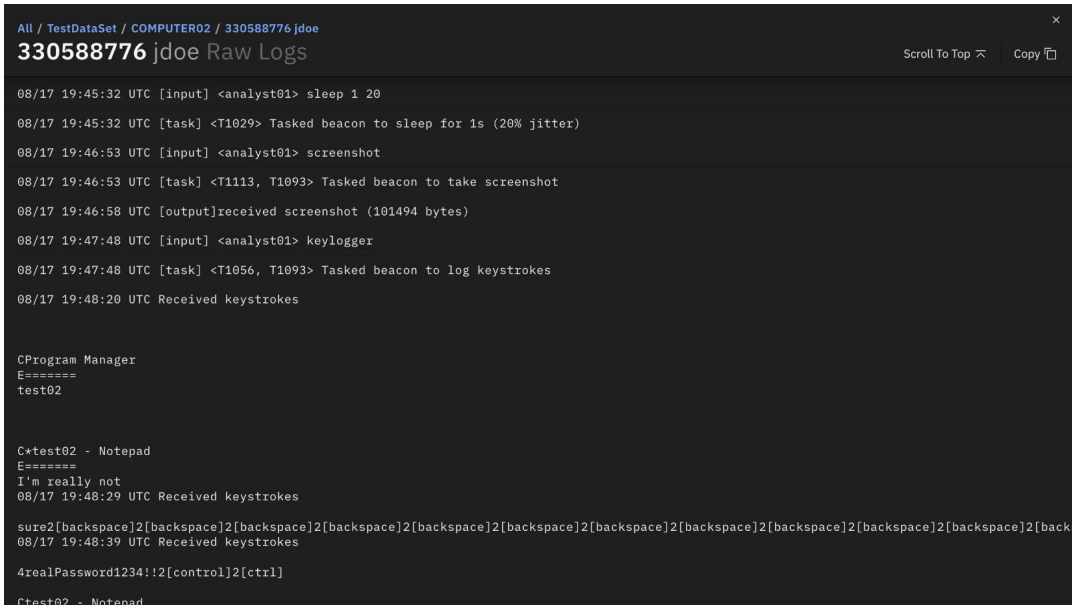


Figure 4.28: The Raw Logs

By downloading the latest release from the GitHub repository [49], it is possible to install RedEye on major operating systems. Depending on the startup mode, the tool can be activated in Red Team or Blue Team mode. The main difference between the two modes lies in the complete

availability of all functions for the Red Team version, while the Blue Team mode offers a simplified version intended for reviewing campaigns exported from Red Team analyses. Once the program is launched, the server hosting the application will be started and accessible via a web browser on the local network. After creating the campaign, there will be a request to import a RedEye database file, which can be in either .sqlite or .redeye format, or Cobal Strike logs via the connection with its server. This file is either generated from a previous analysis using the RedEye program or imported from logs produced by Cobalt Strike. Due to the difficulty of finding online sources of the specific tool in those formats, the developers have provided two database files for demonstration purposes, thanks also to the community which communicated with the developers trough the issue panel in their repository. These files have been imported and represented in the subsequent images, one in the Red Team version of the application and one in the Blue Team version.

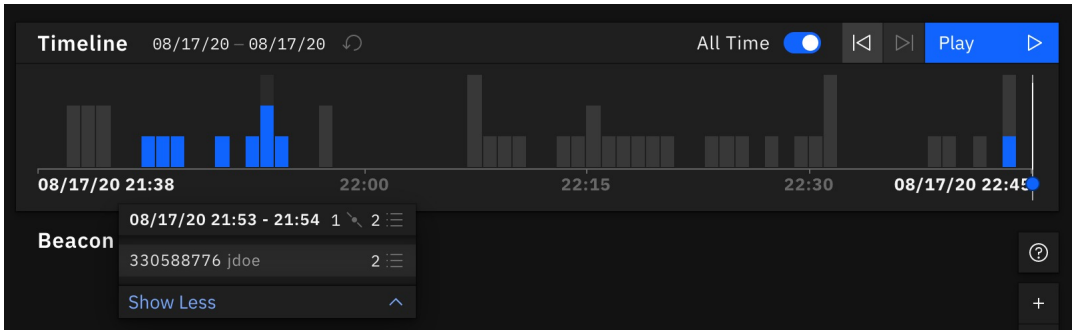


Figure 4.29: The Timeline visualization

From the first provided image (figure 4.27), representing the Red Team version as indicated by the icon in the bottom left corner, we can observe the structure of the application, which also follows a single-page design. As with most dashboards, we can find a sidebar containing the main settings for usage. Immediately next to it is the beacon exploration section, where beacons can be grouped according to different categories. Beacon is Cobalt Strike's payload to model an advanced actor. In the example shown, a beacon has been selected, annotated by a fictional analyst, who categorized it with various tags and added a reference to the threat by linking it to the MITRE website. To display the logs, one can click on the reference link, and a window containing the raw logs will appear (figure 4.28). In the main part of the screen, we have two visualization components. The first one is the Beacon Graph, which represents the connections between various machines depicted as numbered circles. Inside these circles, the detected beacons are represented as dots. At the top, we have the Timeline (figure 4.29), which not only provides a temporal indication of the discovered beacons but also allows users, through dedicated buttons, to recreate the temporal progression of the beacons. This update affects both the graph and the list, providing different observation points based on the selected date. Lastly, in the screen representing the Blue Team version of the application (figure 4.30), we can observe how the functions are limited both by the absence of related tools and by the reminders provided by the application.

Among all the showcased applications, RedEye is certainly the most active one, as its repository is updated weekly, and regular releases are provided. Another advantage of the tool and its developers is the presence of code testing, which is updated with each application update. This is likely due to the practical nature of the tool, rather than being solely research-oriented, as it is designed for real-world contexts even in its early versions. However, compared to the others, RedEye utilizes the visualization components provided by visualization techniques to a lesser extent, leaving room for a mixed interpretation of the classic data visualization and analysis approach.

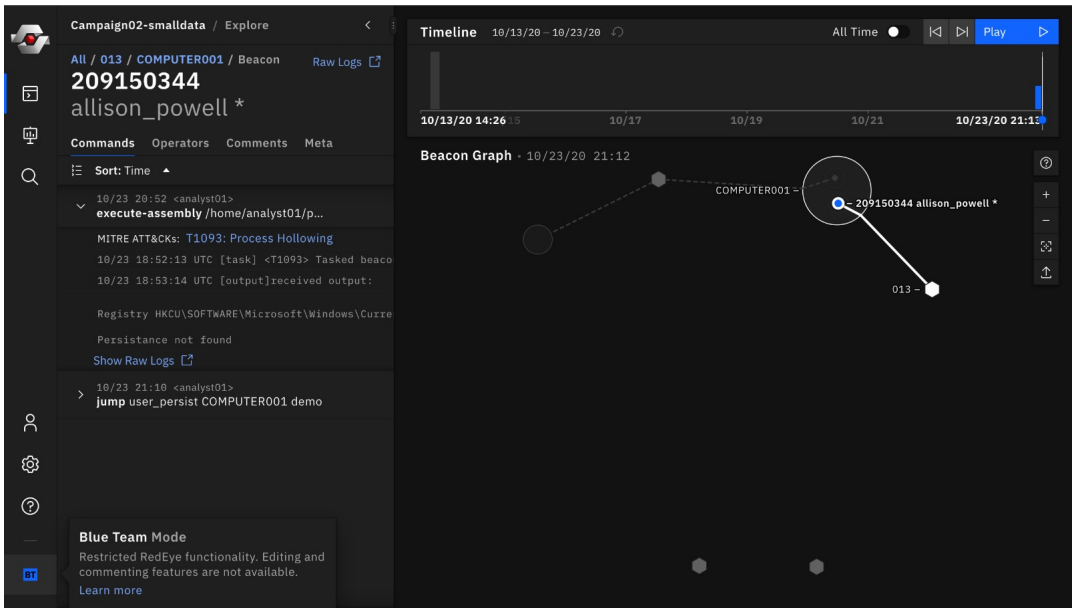


Figure 4.30: RedEye Blue Team version

Vulnus

Vulnus [50] is born with the objective of proposing a solution to dynamically inspect and compare vulnerabilities present within a network and simulate the effects that potential fixes would have on the network. The tool is designed for roles such as security managers, who need to interact with various information regarding the state of system vulnerabilities on the network and make decisions regarding potential fixes and their relative priorities. During the evaluation of priorities, elements such as the spread of vulnerabilities within the network, their impact, and the level of danger they pose should be taken into account. To provide this assessment, vulnerabilities are evaluated using both the CVE [51] score, with associated metrics like CVSS [52], and a score called "Target Environmental", which is automatically calculated by the tool. The Target Environmental score is calculated, or rather extracted, from the Attack Graph that is computed on that network, which is a graphical representation that depicts the potential paths an attacker can take to compromise a computer network or system.

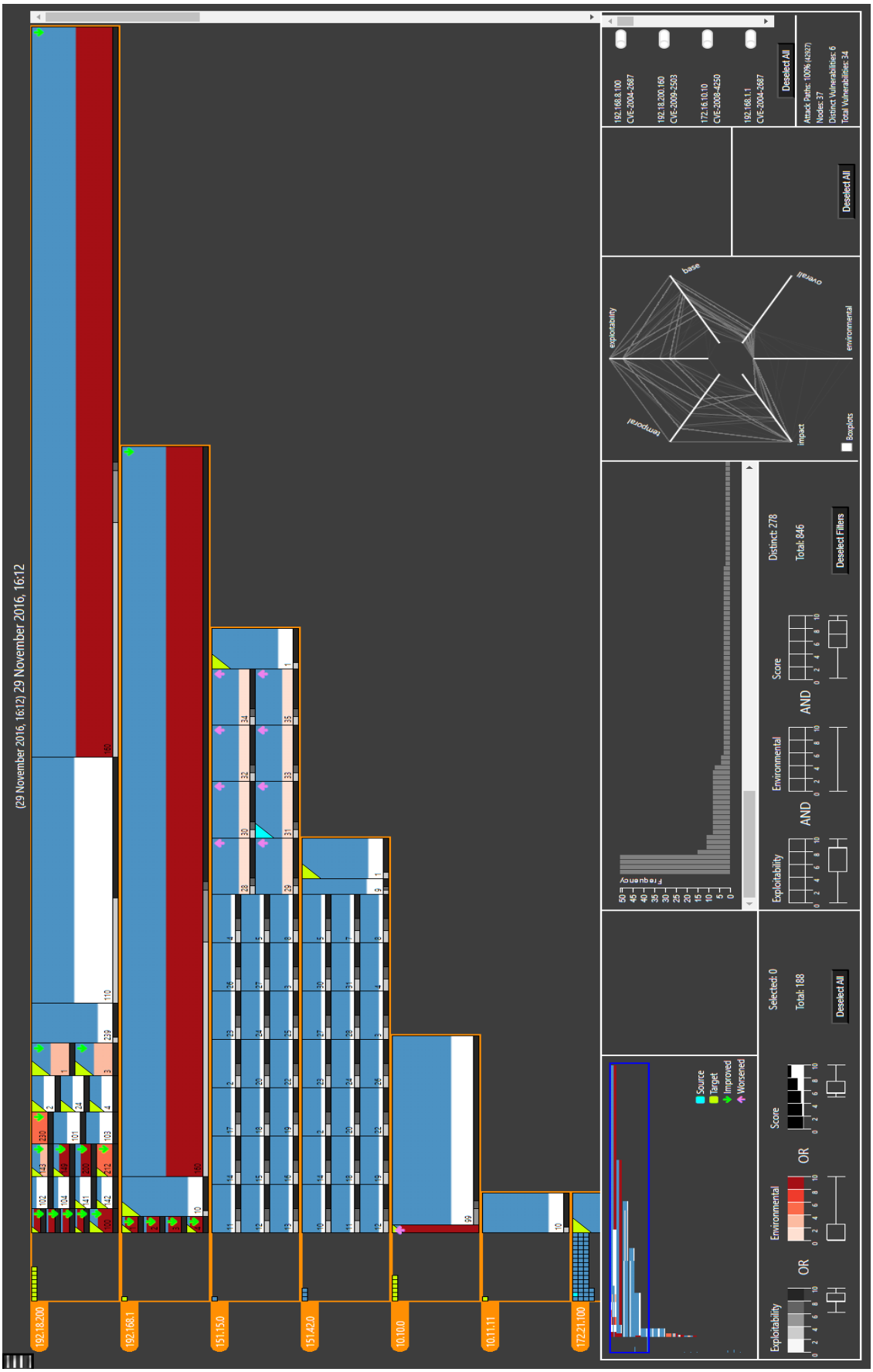


Figure 4.31: Vulnus default sub-network view

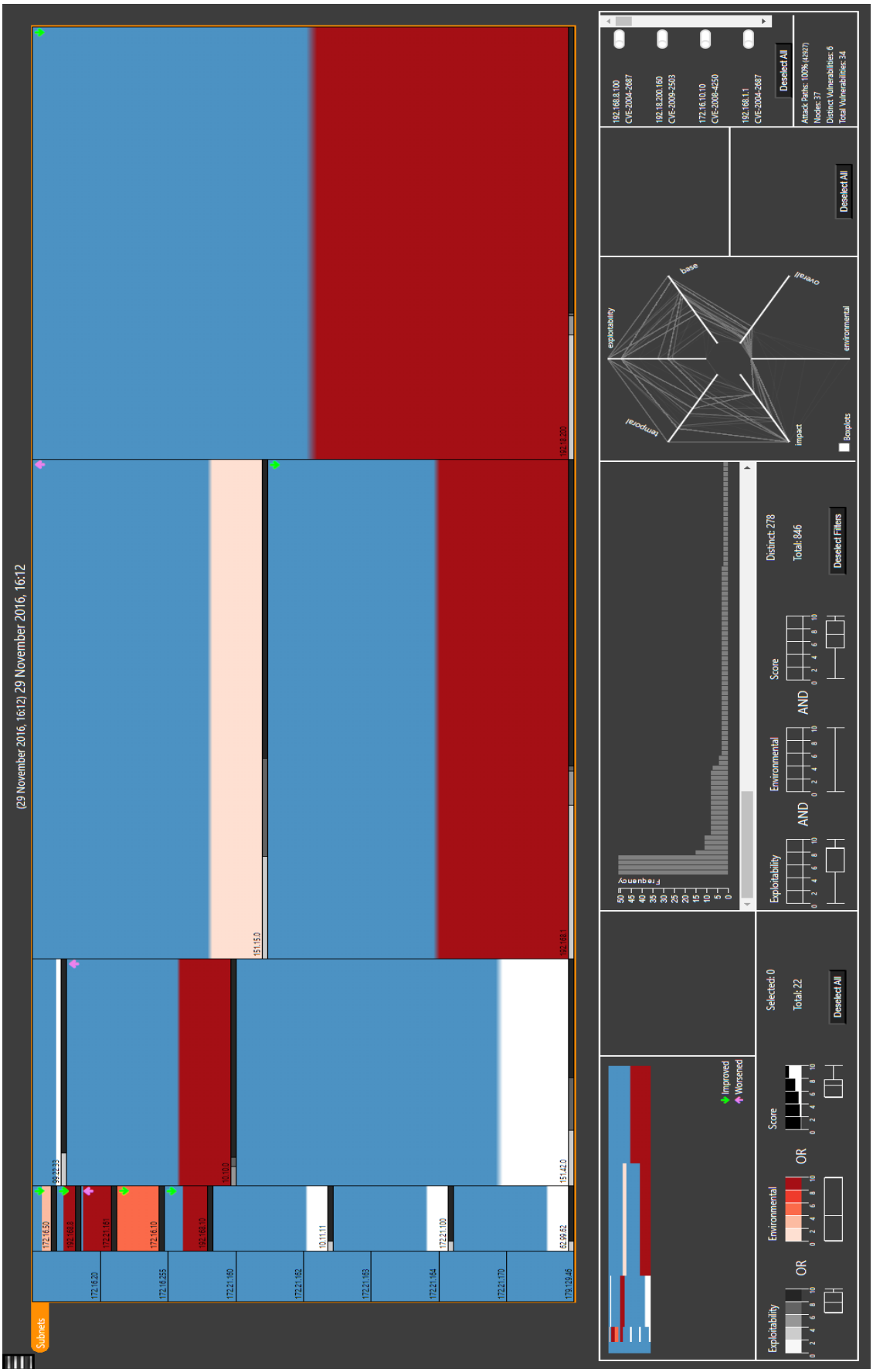


Figure 4.32: Vulnus network view

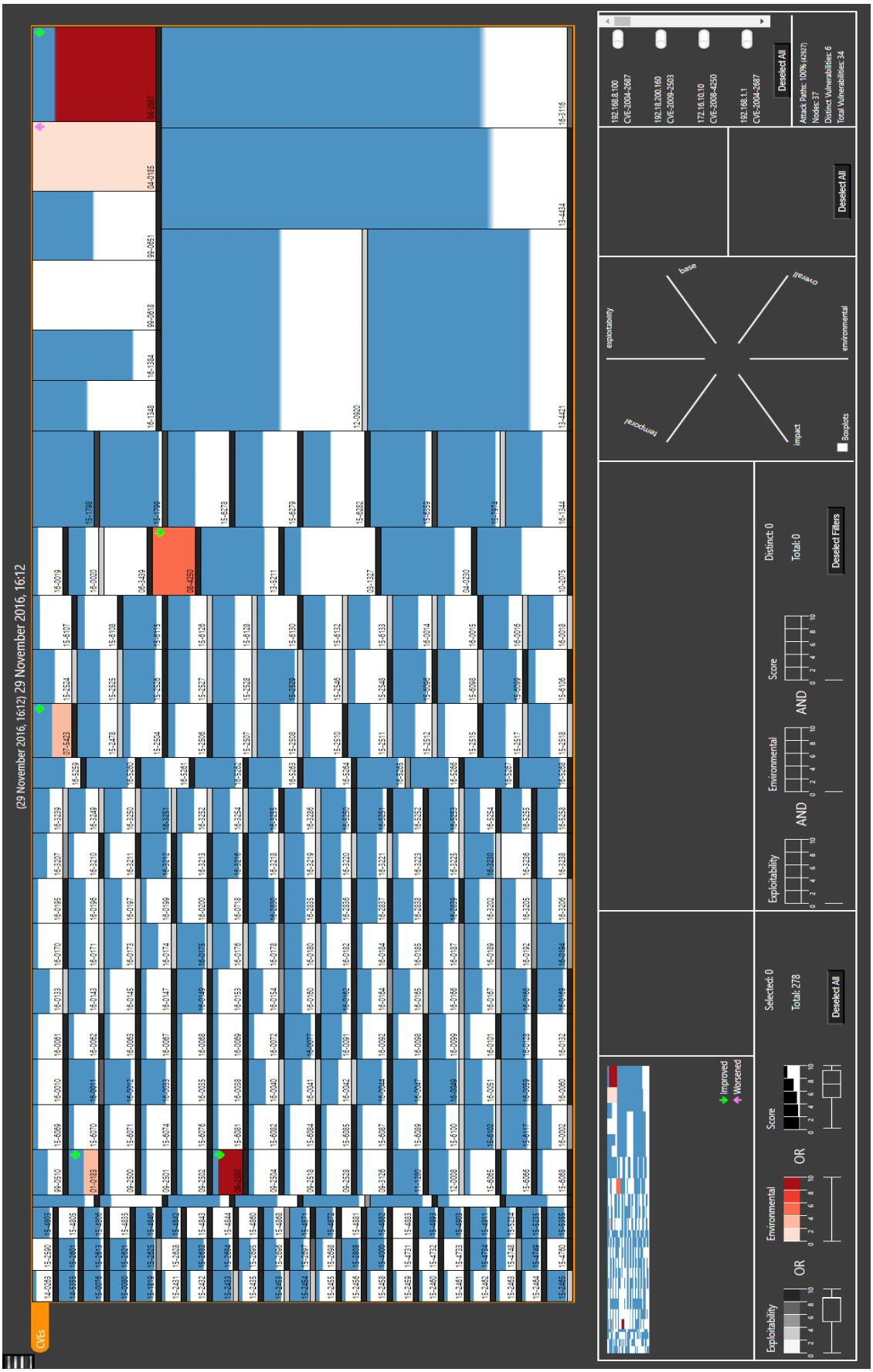


Figure 4.33: Vulnus vulnerability view

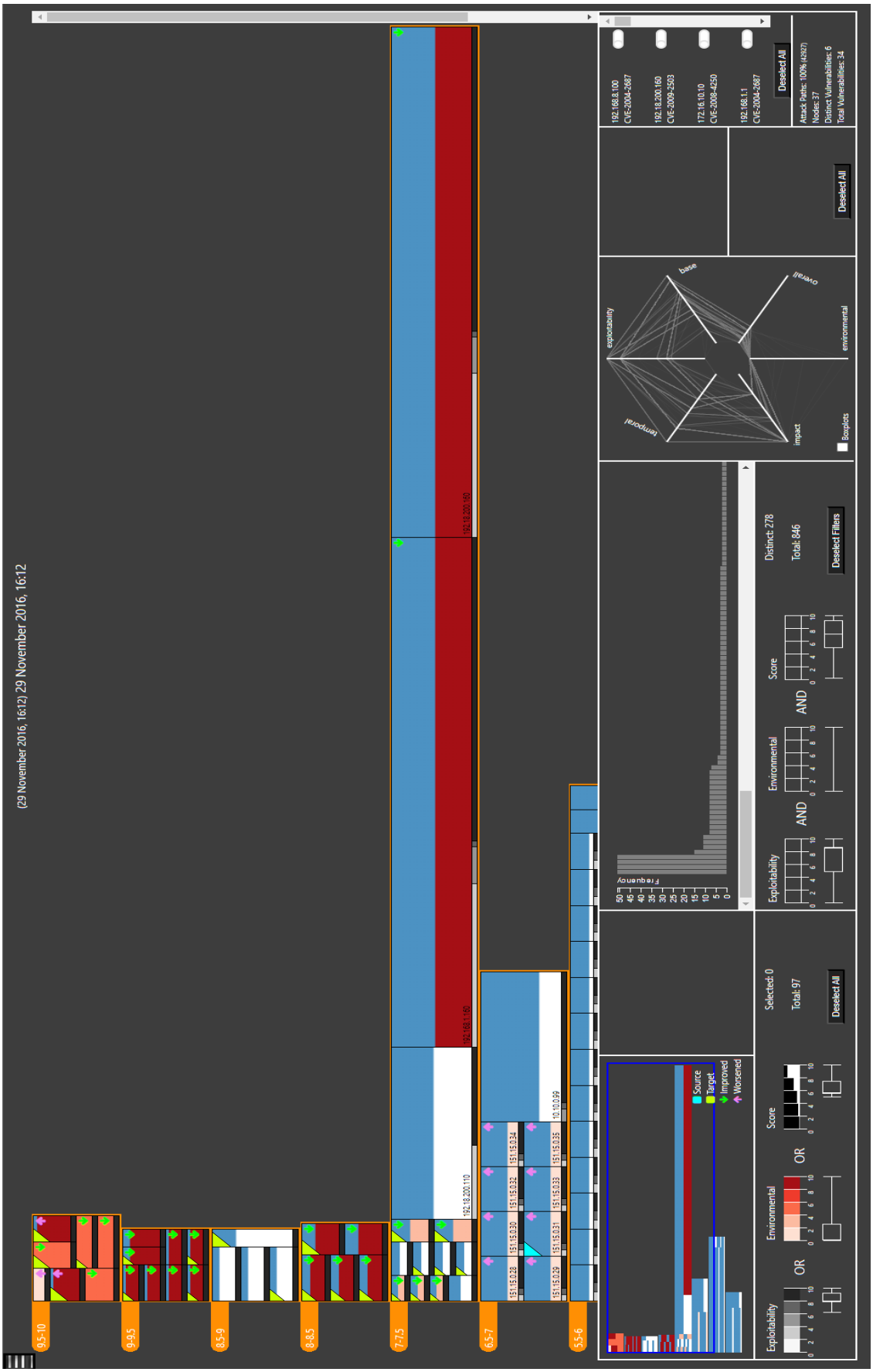


Figure 4.34: Vulnus score view

For the calculation of these metrics, the tool utilizes data from scanners such as OpenVAS and Nessus, which enable the identification of vulnerabilities present in the nodes within the network. In the currently available online prototype version [53], unfortunately, it is not possible to input additional scans or personal files, therefore, the analysis and evaluations can only be performed on the existing dataset upon startup. Upon opening the page, you can observe that the main section is dedicated to the visualization of nodes within the network using a modified version of a TreeMap Bar chart. This panel represents the distribution of nodes based on their subnet, or can be displayed according to vulnerabilities or calculated scores (figures 4.31,4.32,4.33,4.34). The nodes are visualized as rectangles, and within them, various subdivisions of colors and sizes represent different information (figure 4.35), for example, the size of the rectangle indicates the cardinality of the vulnerability. Additionally, in the upper left corner, triangles are placed to indicate whether the node is a source (blue triangle) or a target (yellow triangle). By hovering the mouse over a node, information about its scores is shown. In the bottom part of the page, you will find various tools that can be used for analysis through two main types of operations: selection and inspection. The tool allows you to select nodes in different ways, both from the tool panel and the central panel, leveraging the full interactivity of all the components on the page. Among these tools, there are two elements in particular. The first is a frequency chart of the selected vulnerabilities, while the second is a radar diagram that enables immediate comparison based on score evaluations. Finally, in the bottom-right part of the page, there is a sorted list of vulnerabilities that need to be fixed, calculated through the attack graph. Specifically, the vulnerabilities are displayed in descending order based on their impact on the various paths calculated by the tool. By selecting them, the attack graph is recalculated and, as a result, the visualization is updated, allowing for a "what-if" analysis, as defined by the developers, to provide an idea of the effects that resolving these vulnerabilities may have on the network (figure 4.36).

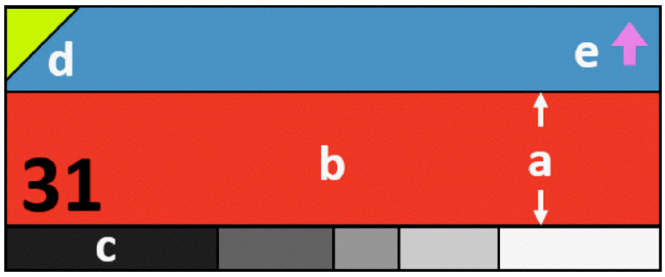


Figure 4.35: Example of node visualization. a) the base score is encoded by a red sliding windows that progressively covers the node blue background and it is in the fourth level (range [6, 8)); b) the target environmental score is encoded by the sliding window red color and it is in the fourth level (range [6, 8]); c) the exploitability bar-gram shows that about 25% of the vulnerabilities have a very high exploitability (black, range [8, 10]); d) the yellow triangle classify the node as a target node; e) the pink upwards arrow indicates the Target Environmental has increased with respect to the last scan. Provided by [50].

And it is precisely this last feature that is one of its major strengths because, in addition to providing an "approximated optimal fixing strategy," by updating the visualization of vulnerabilities and consequently the possible paths at each iteration, it allows the analyst to identify a suboptimal strategy that may differ from the calculated one but is more aligned with the limitations imposed by the organizational context, which may not always allow for instant fixes. At the visualization level, as seen in this example, it is evident that even large-scale networks can be represented clearly without the loss of information. Unfortunately, as previously mentioned, the tool is only available in this non-modifiable prototype version. Despite its functional and adequate features, the inability to input scan files limits the tool's usability, and additionally, no information is provided regarding the current development status, thereby restricting its purpose to a practical demonstration.

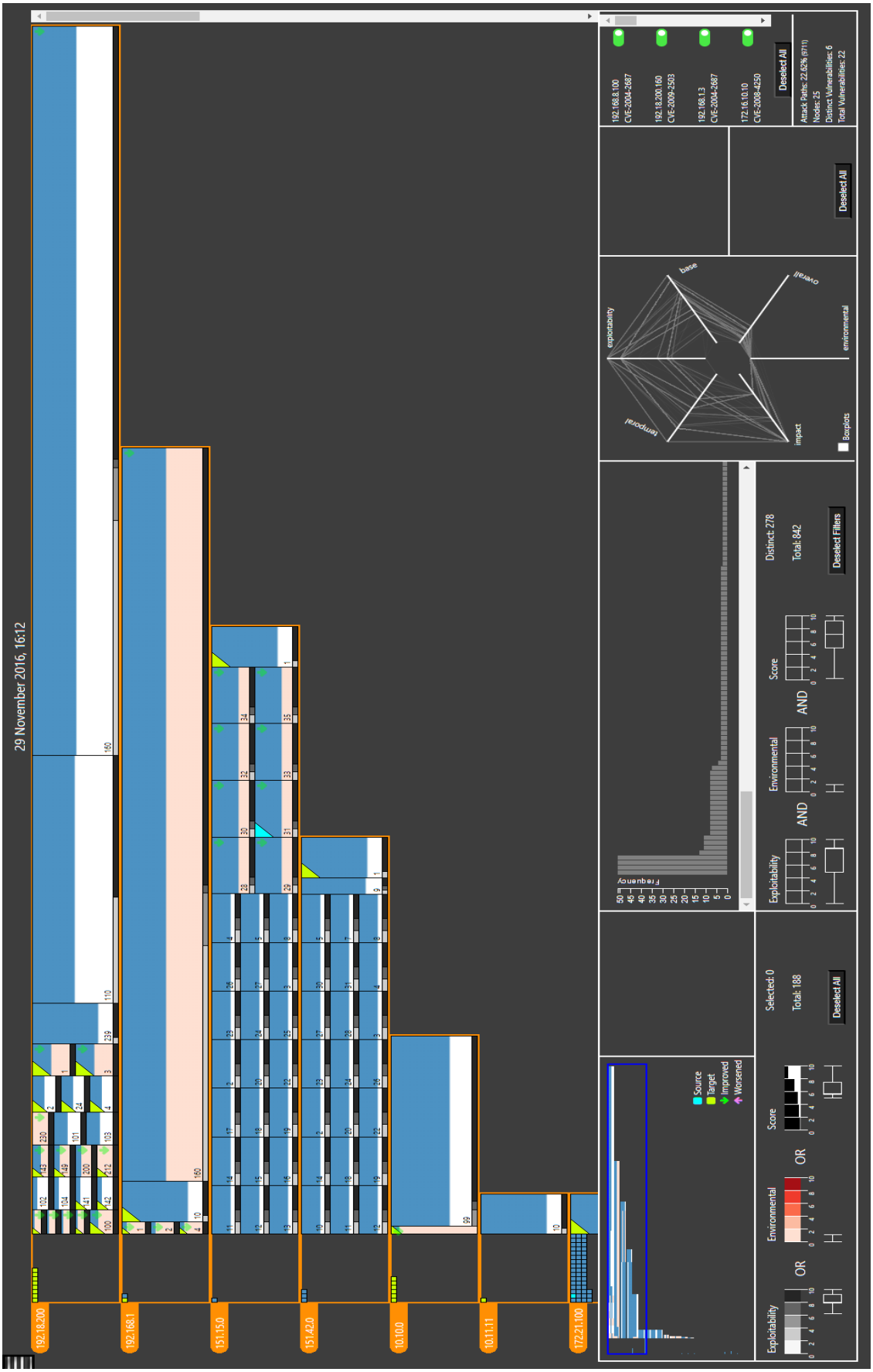


Figure 4.36: Vulnus view after first 4 CVE fixed in “what-if” analysis

In terms of the development phases, we can identify a phase of defining *Visualization goals* conducted in collaboration with specialists such as security managers and security operators. The paper does not provide specific details, but rather examples of the main activities that need to be carried out. During the *data Preparation* phase, as mentioned earlier, it is possible to use data from scanners, even though uploading is not allowed in the current prototype. One of the main features of the tool lies precisely in this phase, which involves processing this data for the calculation of the attack graph and its corresponding scores. The paper provides interesting insights for the *Exploration* phase, as it not only emphasizes collaboration in decision making with the group of specialists, but also highlights the continuous modification of initial visualizations based on their interactions, leading to the prototype we observe. For the final phase of *Feedback and fine-tuning*, once again, the use of surveys is mentioned for evaluating usability and effectiveness with users of varying levels of experience, resulting in a generally positive evaluation from more experienced users.

Crumbs

Crumbs [54] is visual analytics solution aimed at addressing the Italian adaptation of the Cyber Security Framework (IACSF), derived from the proposal of the National Institute of Standards and Technology (NIST), targeting security managers. At its core, the framework proposed by NIST consists of five key functions: Identify, Protect, Detect, Respond, and Recover, which provides an understanding of the possible management of cyber security risks. This functions are structured in categories and each category is structured in subcategories which refer to practical actions that must be carried out. The Italian model extends this structure by adding a priority level and a maturity level to each subcategory. The Priority level can take 4 values (Mandatory, High, Medium, and Low) and primarily indicates the order of implementation. The maturity level provides a reference by which each organization can assess the implementation of its subcategories and establish objectives and priorities for their improvement and it can be divided into 4 incremental sublevels ranging from M1, such as mandatory legal actions, to M4, which represents the most sophisticated risk management practices.

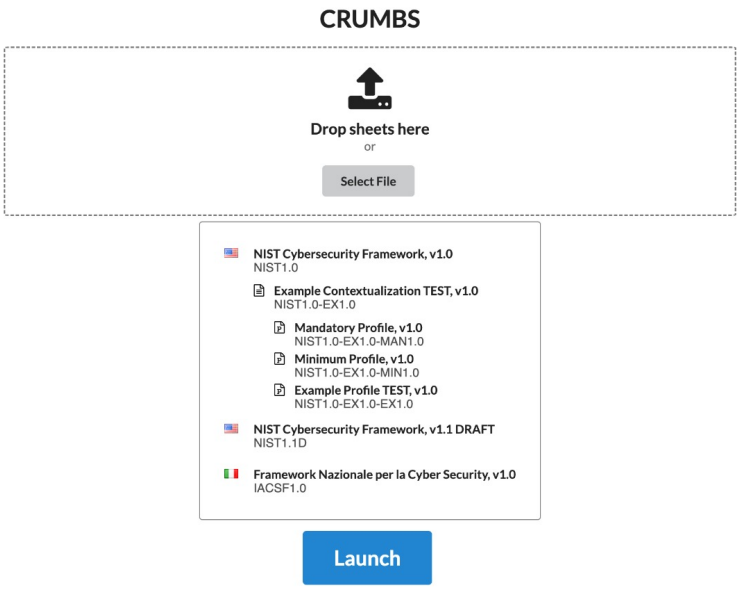


Figure 4.37: Crumbs launching page

The prototype is available both as an online demo version and as a downloadable code that can be run locally, with both options accessible from the same page [55]. In the online demo version sample files are available for usage, while in the local version, it's possible to create a copy of the base Excel framework files and modify them according to different needs. Once the

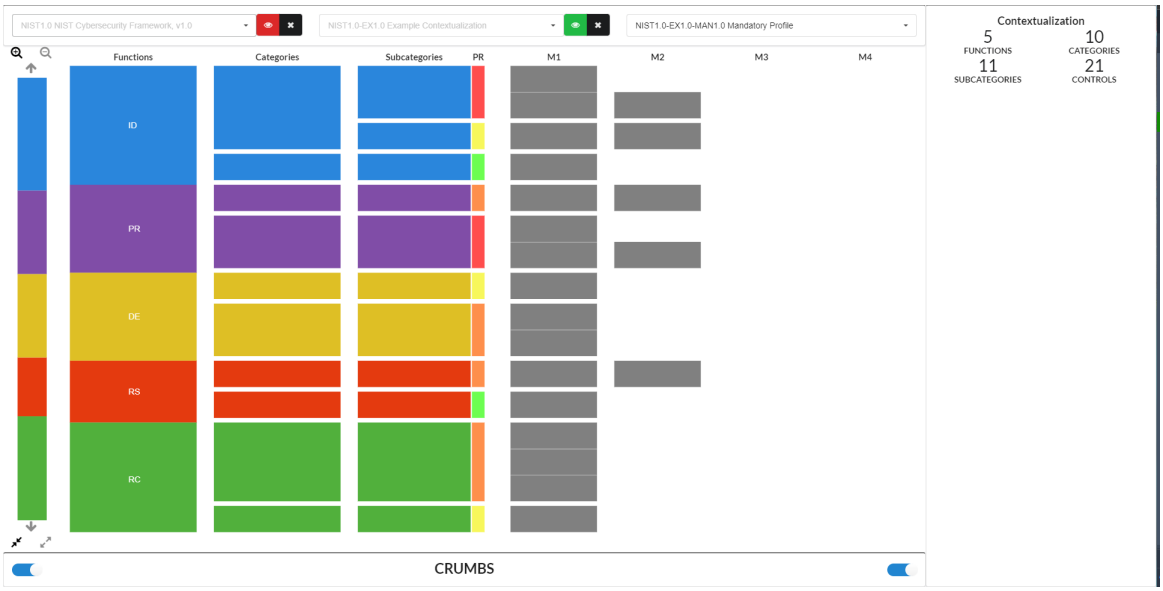


Figure 4.38: Crumbs first view

Figure 4.39: Crumbs with Gap Analysis

server is started, the file can be uploaded to the application page and initiate the visualization (figure 4.37). After selecting the reference framework through the top toolbar, the main view of the application is displayed (figure 4.38). Once the contextualization, described in the Excel file, is selected, the view is updated to include visual components that refer to the priority level and maturity level. The priority level is represented using a color code (red for mandatory, orange for high priority, yellow for medium priority, green for low priority, and white for no priority), while the maturity level, divided into sub-levels, is indicated by a simple color code to denote whether it has been addressed or not (not addressed=red, partially addressed=half green-half red, fully addressed=green) and it's displayed after the selection of the current profile, which represents the current status of cybersecurity of the organization. In case there is a need to navigate between different controls, it is possible to use the toolbar located on the left side of the functions to adjust the zoom level. Once the current profile is selected, it is possible to enter a "target profile" to perform a gap analysis by adding two new visualizations (figure 4.39). The first one is called "Gap Analysis" and for each subcategory two bars are depicted, where the left one, with normal colors, represents the status of all the controls associated to the target profile, where green and

red represent the percentage of addressed (green) and not addressed (red) controls. The right one, more transparent colors, provides the same information for all controls that are above the target maturity level for the subcategory and mouse-overing on the elements in the gap bars reveals the associated controls along the different maturity levels. The second visualization created is called "Minimum Coverage" and represents an optimized order of the controls that still need to be addressed to reach the target profile, which can vary depending on the strategies employed (figures 4.41, 4.40). On the right side of the main display, there are two bar charts that represent the overall statistics. In the upper image, the frequency distribution of subcategories is visible with respect to degree of coverage, while in the lower image, the frequency distribution of controls is visible with respect to degree of addressing.

As highlighted by the developers themselves, Crumbs offers a visualization methodology that has not been adopted by other solutions operating in the same field, making it undoubtedly a unique solution of its kind. While it has proven to be a useful solution from the perspective of experts considered for a prototype review, it should not be considered a complete replacement for traditional methods, as it does not allow for the modification of the states of various subcategories. Updating the addressed fields requires a modification to the original document, which poses a significant limitation for the prototype. In addition to this limitation, it is important to note the instability of the offline version of the tool compared to the online demo version, resulting in variations in the page's visualizations that hinder its proper usage.

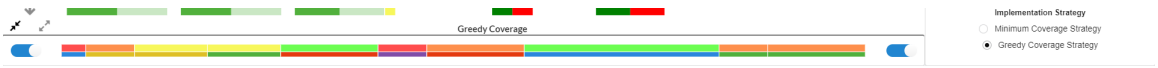


Figure 4.40: Crumbs Greedy Coverage Strategy

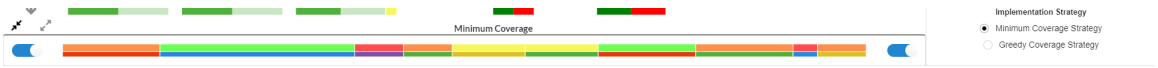


Figure 4.41: Crumbs Minimum Coverage Strategy

SAGE

The tool presented in this section differs from previous ones as it is not an application that monitors or allows interactive visualization of inputted data, instead it autonomously calculates attack graphs from intrusion alerts. SAGE [56] (which stands for IntruSion alert-driven Attack Graph Extractor) enables the generation of these attack graphs without the manual support of domain experts, making the use of these strategies faster and more accessible, addressing their previously perceived limitations of being costly and ineffective in real-world scenarios.

Attack graphs are visual representations of potential attack paths within a computer network. They depict the interconnected relationships between vulnerabilities, hosts, and exploits, illustrating how an attacker can traverse through the network. By mapping out these paths, security analysts can gain valuable insight into possible attack vectors and prioritize mitigation strategies to strengthen network defenses. Attack graphs aid in identifying and understanding potential threats, facilitating proactive measures to enhance network security.

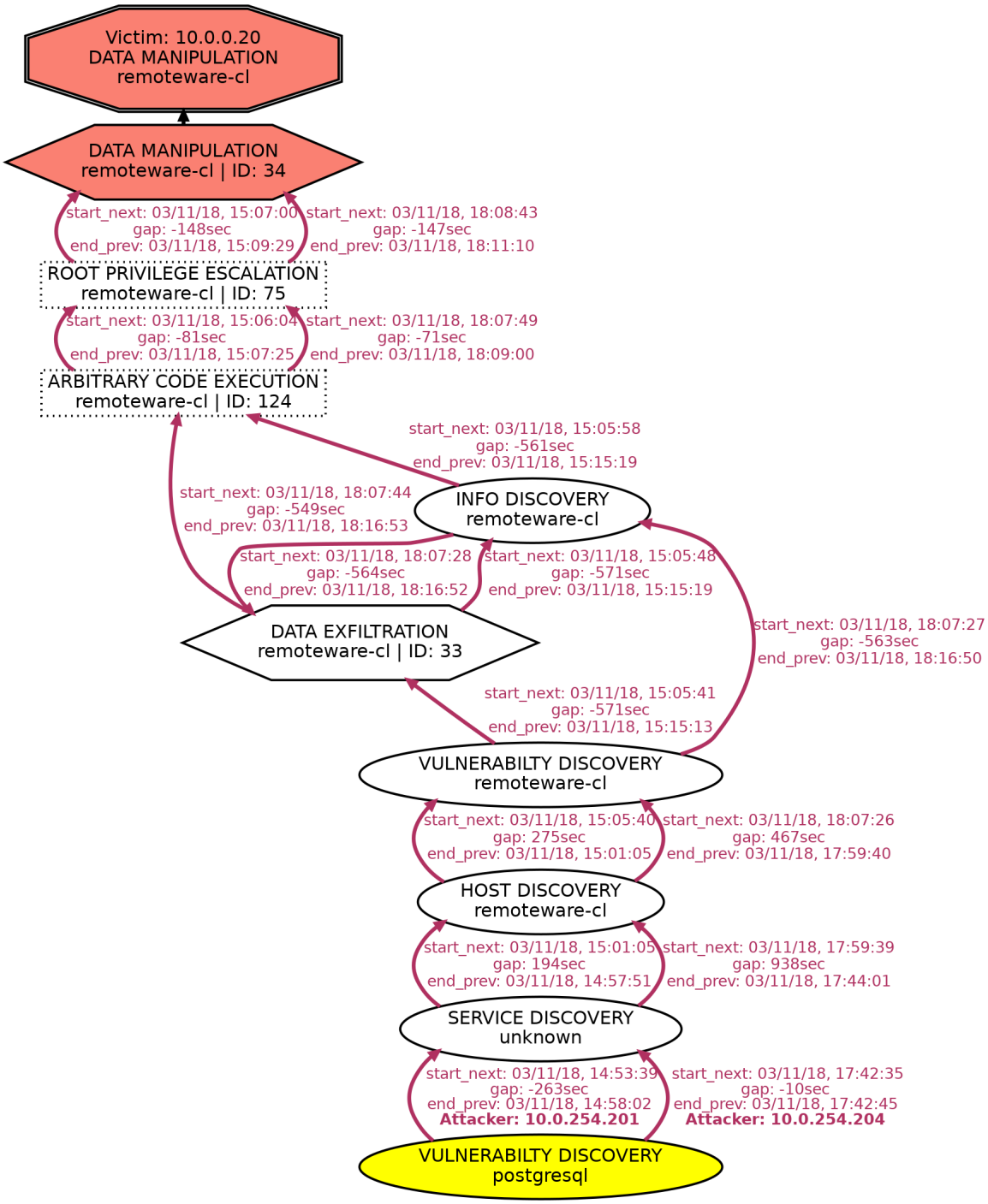

Victim: 10.0.0.20
DATA MANIPULATION
remoteware-cl
DATA MANIPULATION
remoteware-cl | ID: 34
start_next: 03/11/18, 15:07:00
gap: -148sec
end_prev: 03/11/18, 15:09:29
start_next: 03/11/18, 18:08:43
gap: -147sec
end_prev: 03/11/18, 18:11:10
ROOT PRIVILEGE ESCALATION
remoteware-cl | ID: 75
start_next: 03/11/18, 15:06:04
gap: -81sec
end_prev: 03/11/18, 15:07:25
start_next: 03/11/18, 18:07:49
gap: -71sec
end_prev: 03/11/18, 18:09:00
ARBITRARY CODE EXECUTION
remoteware-cl | ID: 124
start_next: 03/11/18, 15:05:58
gap: -561sec
end_prev: 03/11/18, 15:15:19
start_next: 03/11/18, 18:07:44
gap: -549sec
end_prev: 03/11/18, 18:16:53
INFO DISCOVERY
remoteware-cl
start_next: 03/11/18, 18:07:28
gap: -564sec
end_prev: 03/11/18, 18:16:52
start_next: 03/11/18, 15:05:48
gap: -571sec
end_prev: 03/11/18, 15:15:19
DATA EXFILTRATION
remoteware-cl | ID: 33
start_next: 03/11/18, 18:07:27
gap: -563sec
end_prev: 03/11/18, 18:16:50
start_next: 03/11/18, 15:05:41
gap: -571sec
end_prev: 03/11/18, 15:15:13
VULNERABILTY DISCOVERY
remoteware-cl
start_next: 03/11/18, 15:05:40
gap: 275sec
end_prev: 03/11/18, 15:01:05
start_next: 03/11/18, 18:07:26
gap: 467sec
end_prev: 03/11/18, 17:59:40
HOST DISCOVERY
remoteware-cl
start_next: 03/11/18, 15:01:05
gap: 194sec
end_prev: 03/11/18, 14:57:51
start_next: 03/11/18, 17:59:39
gap: 938sec
end_prev: 03/11/18, 17:44:01
SERVICE DISCOVERY
unknown
start_next: 03/11/18, 14:53:39
gap: -263sec
end_prev: 03/11/18, 14:58:02
Attacker: 10.0.254.201
start_next: 03/11/18, 17:42:35
gap: -10sec
end_prev: 03/11/18, 17:42:45
Attacker: 10.0.254.204
VULNERABILTY DISCOVERY
postgresql

Figure 4.42

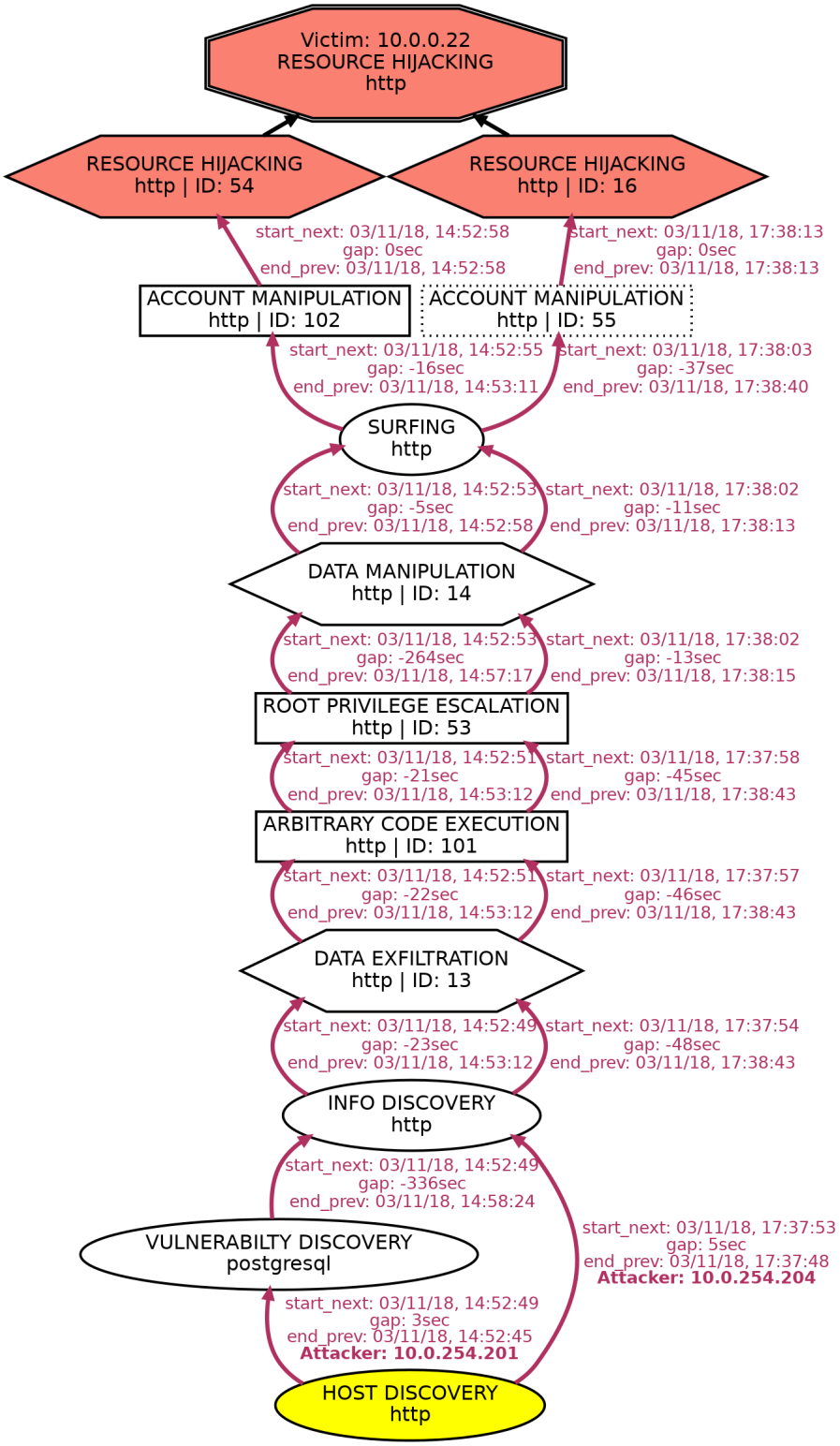

Victim: 10.0.0.22
RESOURCE HIJACKING
http
RESOURCE HIJACKING
http | ID: 54
RESOURCE HIJACKING
http | ID: 16
start_next: 03/11/18, 14:52:58
gap: 0sec
end_prev: 03/11/18, 14:52:58
start_next: 03/11/18, 17:38:13
gap: 0sec
end_prev: 03/11/18, 17:38:13
ACCOUNT MANIPULATION
http | ID: 102
ACCOUNT MANIPULATION
http | ID: 55
start_next: 03/11/18, 14:52:55
gap: -16sec
end_prev: 03/11/18, 14:53:11
start_next: 03/11/18, 17:38:03
gap: -37sec
end_prev: 03/11/18, 17:38:40
SURFING
http
start_next: 03/11/18, 14:52:53
gap: -5sec
end_prev: 03/11/18, 14:52:58
start_next: 03/11/18, 17:38:02
gap: -11sec
end_prev: 03/11/18, 17:38:13
DATA MANIPULATION
http | ID: 14
start_next: 03/11/18, 14:52:53
gap: -264sec
end_prev: 03/11/18, 14:57:17
start_next: 03/11/18, 17:38:02
gap: -13sec
end_prev: 03/11/18, 17:38:15
ROOT PRIVILEGE ESCALATION
http | ID: 53
start_next: 03/11/18, 14:52:51
gap: -21sec
end_prev: 03/11/18, 14:53:12
start_next: 03/11/18, 17:37:58
gap: -45sec
end_prev: 03/11/18, 17:38:43
ARBITRARY CODE EXECUTION
http | ID: 101
start_next: 03/11/18, 14:52:51
gap: -22sec
end_prev: 03/11/18, 14:53:12
start_next: 03/11/18, 17:37:57
gap: -46sec
end_prev: 03/11/18, 17:38:43
DATA EXFILTRATION
http | ID: 13
start_next: 03/11/18, 14:52:49
gap: -23sec
end_prev: 03/11/18, 14:53:12
start_next: 03/11/18, 17:37:54
gap: -48sec
end_prev: 03/11/18, 17:38:43
INFO DISCOVERY
http
start_next: 03/11/18, 14:52:49
gap: -336sec
end_prev: 03/11/18, 14:58:24
VULNERABILTY DISCOVERY
postgresql
start_next: 03/11/18, 17:37:53
gap: 5sec
end_prev: 03/11/18, 17:37:48
Attacker: 10.0.254.204
start_next: 03/11/18, 14:52:49
gap: 3sec
end_prev: 03/11/18, 14:52:45
Attacker: 10.0.254.201
HOST DISCOVERY
http

Figure 4.43

The tool processes raw Suricata [57] intrusion alerts and converts them into consolidated sequences. These sequences are utilized to generate a machine learning model that summarizes the attack paths present in the data. Attack graphs are then derived from this model, customized to specific objectives and victims, proposing some concise and comprehensible data, effectively condensing extensive alert data to showcase the progression of an attack.

Using the tool's repository [58], it is possible to download the code available in both the development version and the Docker version, allowing for quick configuration and subsequent utilization. To test its functionality, the Docker version was used, which can be found in the branch named "docker" within the repository, and Suricata alerts from a college cyber defense competition [59] were employed for this purpose. After following the brief instructions for execution, from a simple file containing 161 alerts, 51 attack graphs were generated, out of which 2 are reported in the images (figure 4.42, 4.43). This alert-driven Attack Graphs show paths towards an objective. Vertex labels (in red) are attack stage, targeted service and state identifier. Actions are identified by their shape, low-severity actions are ovals, medium severity are boxes and high-severity are hexagons. The first action in a path is yellow, while the objective-variants are red and actions that occur too infrequently in the analysis are dotted. Furthermore, the temporal information contained in the alerts is also displayed above their respective boxes. In addition to calculating the attack graphs, the tool also provides a final statistic regarding the frequency of various alerts present in the analyzed file. An example of the analysis test is shown in the figure (figure 4.44).

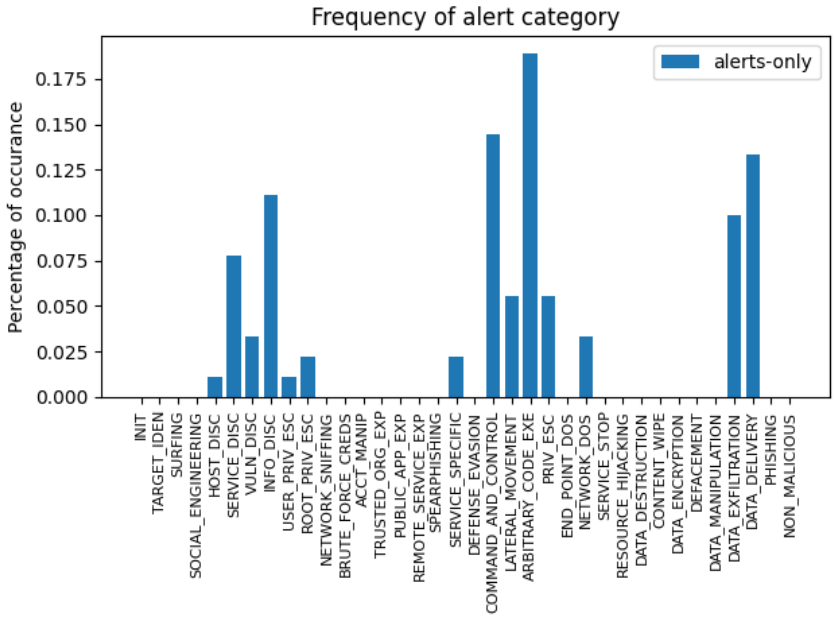


Figure 4.44: Statistics on alert data

SAGE has several strengths. Its main purpose is to reduce, or rather correlate, the various pieces of information contained in the alerts and provide the analysts with the right information through proper visualization. In addition to this, it can also be used for other operations, such as visually comparing different strategies adopted to attack the same victim, which is a highly useful analysis for an analyst. Another utility mentioned by the developers is the ability to utilize the tool and its results to improve or add IDS rules. Moreover, the option to use the tool without the need for specific setups or systems, thanks to the Docker container, allows for quick and simplified usage. The main limitation, however, is due to the only type of file that can be analyzed, which is Suricata alerts, although Suricata is one of the network analysis and threat detection software widely used by private and public organizations, it does not guarantee the tool's usability in certain analysis contexts.

Vulnex

The last tool presented in this section is called Vulnex [60], which stands for Vulnerability Explorer, a tool to audit software development organizations. Vulnex is designed for explore and assess the mitigation of OSS vulnerabilities, through a visualization of vulnerabilities related to the code, allowing the estimation of the overall risk exposure of development organizations. Nowadays, developers and security analysts in development companies require a well-defined analysis due to the widespread use of open-source components in various development areas.

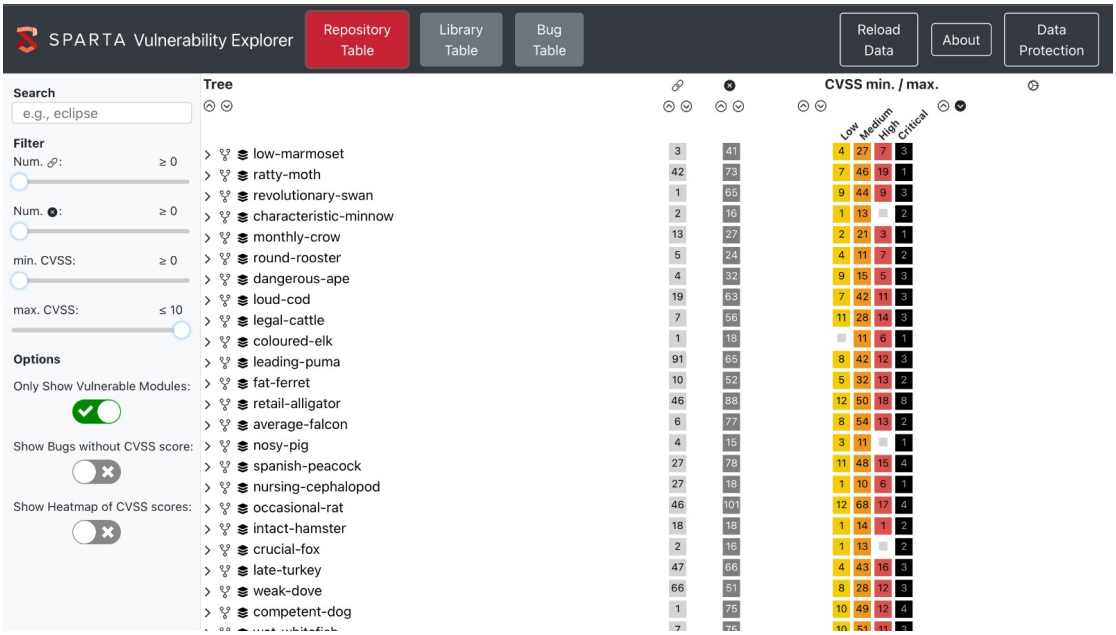


Figure 4.45: Vulnex default view

For its operation, the tool requires a previously conducted code analysis, specifically through Eclipse Steady [61], which supports static and dynamic analysis. Afterwards, by downloading the tool from the dedicated repository [62], it is possible to connect it to the Steady backend using the appropriate settings and launch the Docker build, which will deploy the backend and frontend services of the application. Once the setup and launch are complete, the interactive visualization can be accessed through a web browser. By default, when downloading the application code, a demo mode is set, allowing the visualization of an analysis conducted by the developers on the public repository of the Eclipse Foundation [63]. The subsequent images will specifically pertain to this demo mode, given the inability to test personal codes, limitation that will be described later.

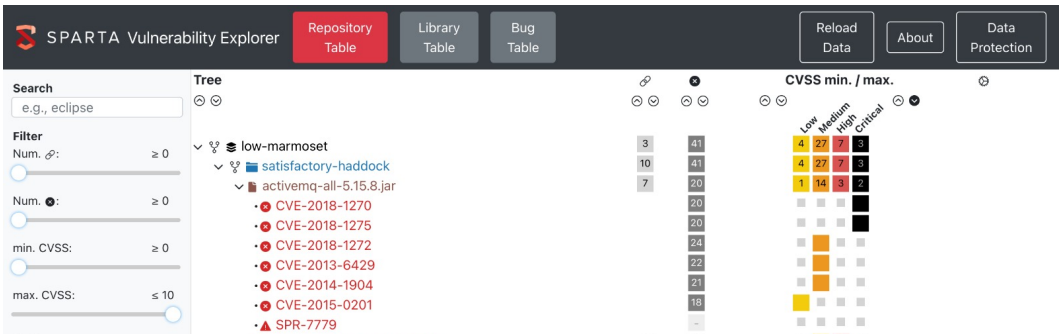


Figure 4.46: Repository view

After launching the application via the Docker container, we can see the main screen of the tool as shown in the figure (figure 4.45). In this case as well, we find a single-page view

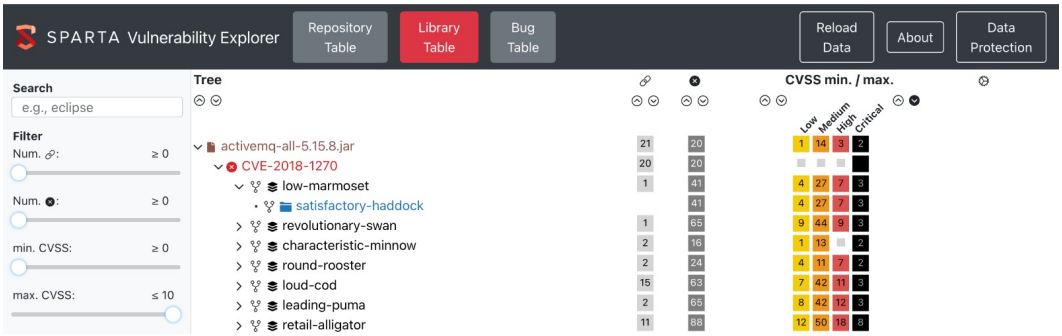


Figure 4.47: Library view

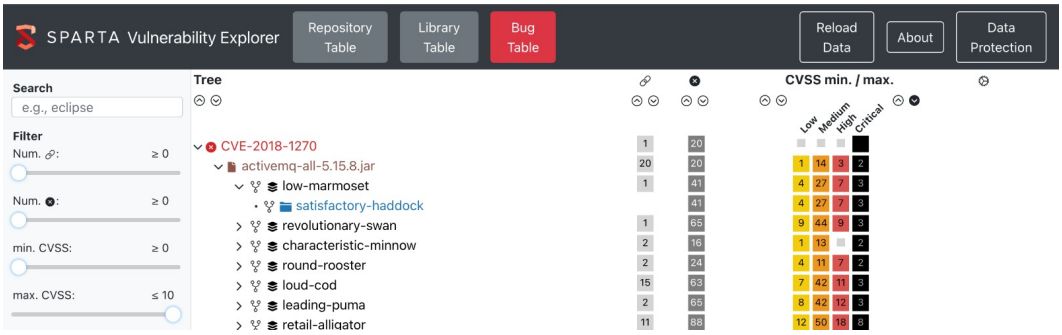


Figure 4.48: Bug view

where in the center, we have the main view, while on the side and at the top, we have the available filter section and various visualization modes, respectively. The three different types of visualizations (Repository, Library, Bug) primarily differ in the order in which the dependency tree is calculated, allowing for differentiation between the different perspectives required for the analyses. An example of the different dependencies is shown in the figure (figure 4.46, 4.47, 4.48). Beside the visualization of the dependencies we find the number of occurrences of the vulnerabilities classified into 4 ranges (Low, Medium High and Critical), which refers to the National Security Database [64] for coloring. In addition to this representation, it is possible to obtain a distribution of the CVSS scores using the appropriate button which changes the display mode to a heatmap (figure 4.49). Finally, for each repository, it is possible to inspect its structure through a tree representation, which shows the repository in question as a single vertex, the various modules related to it and finally the vulnerabilities present (figure 4.50).

The tool, which aims to allow analysts to detect severe and relevant vulnerabilities and determine impacted libraries, modules, and repositories, has received positive evaluations from security experts who conducted the initial user feedback sessions. From their comments, it has also emerged that the visualizations that provide an overview of the general status were more utilized and preferred compared to specific detailed visualizations. In general, the positive evaluations align with the fact that the tool, through its various types of visualizations, can address different questions that analysts need to consider during their analysis, such as “Which repositories contain the most severe vulnerabilities?”, “Which severe vulnerabilities are present?”, or by simultaneously representing multiple projects, it can also answer questions like “Which dependencies contain severe vulnerabilities and are often used across different applications?”, thereby providing a more diverse and comprehensive analysis. Regarding the current limitations, when using the demo version, it can be observed that the data calculation timings are not optimized, requiring several minutes for the rendering process as shown in the figure. Additionally, compared to the presentation described in the paper, there are missing representations, such as the “Meta information” matrix. Furthermore, a significant issue encountered during the usage test was the connection of the application with the Eclipse Steady backend, which prevented the loading of personal projects, despite similar problems being addressed through the GitHub page’s issue section by other community users. The dependency on Steady also presents another limitation of the application, as it requires

a prior code analysis, and the exclusivity of the connection with Steady only provides utility for applications written in Java and Python, thus not covering a wide range of solutions adopted by development companies.

In terms of the tool's development process, we can identify the main phases described in previous chapters. For the *Visualization Goals* definition phase, there were collaborations between the tool developers and industry experts such as security analysts and software developers. Among the objectives defined in this phase, we can find references to goals such as "provide views to detect vulnerable repositories and projects to apply countermeasures" and "the tool needs to show the impact of specific bugs on the organization's codebase", which by the same experts have been defined as achieved by the described version of the tool. In the subsequent *Data Preparation* phase, as mentioned above, the developers chose the public repository of the Eclipse Foundation as the reference dataset and during the data cleaning sub-phase, they replaced the original repository and module names with pseudonyms to avoid attributing problems to individual projects. For the subsequent two phases, *Exploration* and *Visualization*, specific details are not provided as the choice of a simple visualization type does not require extensive studies, as an example, the color scheme used, dictated by the National Security Database, leverages an already defined standard.In this last phase, *Feedback and fine-tuning*, similar to the entire development process, collaboration with domain experts is emphasized which, in addition to providing positive evaluations on the usefulness of the tool, also offer critical feedback aimed at improving or adding functionalities, such as "adding additional information about the open-source libraries to the tool", or provide guidance on how to resolve critical vulnerabilities, given the need to resolve these vulnerabilities within several hours. Despite this feedback, it is noticeable that there have been no changes regarding these functionalities in the repository, although there have been recent updates in the past few months, they primarily focus on improving the codebase and do not include any modifications to its functionality.

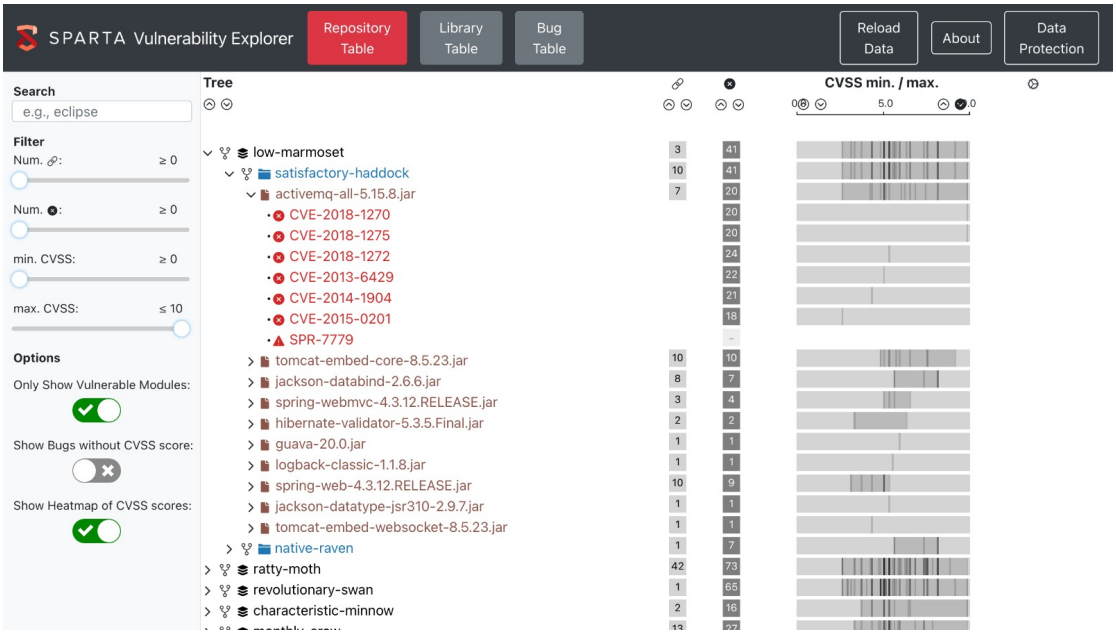


Figure 4.49: Heatmap View

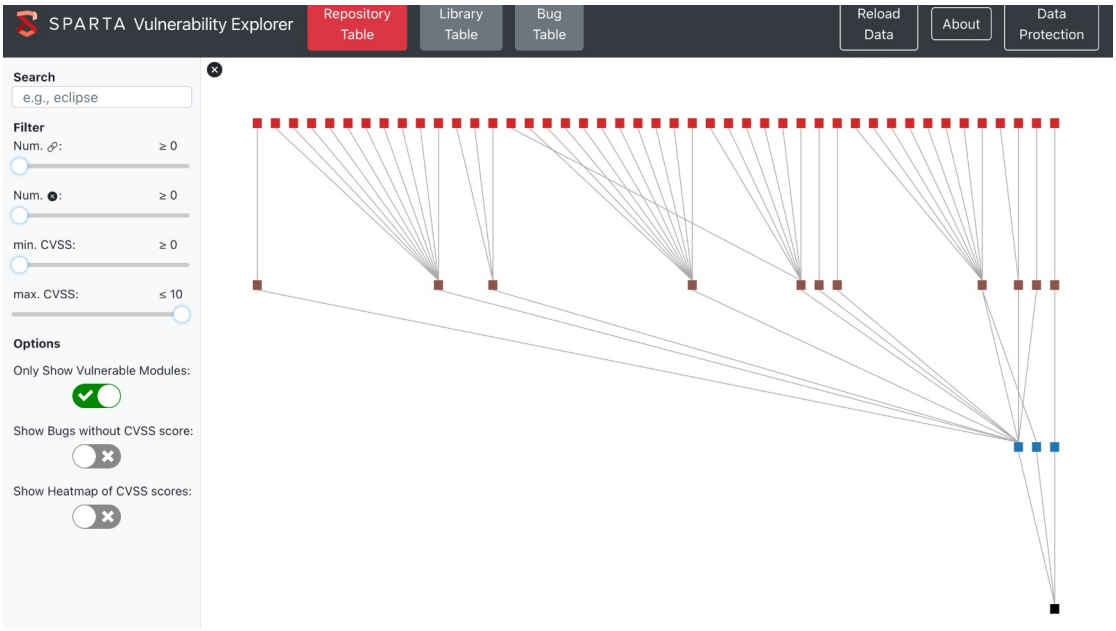


Figure 4.50: Vulnex Repository Tree view

4.2 Unavailable Tools

In this section, briefly mentioned are the various tools encountered during the research that are not available or have not been manually analyzed for different reasons.

Flowtag FlowTag [65] is a network trace viewer that offers an interactive and intuitive interface. It processes PCAP files to generate a comprehensive database of flows, which can be visualized and analyzed by the user. The tool enables users to filter and focus on specific flows of interest, examine the payload, and annotate them with relevant keywords. FlowTag is implemented in Ruby using the Tk interface and incorporates a combination of flow-based visualization, a parallel-coordinates view for flow selection, and keyword tagging. This integrated approach facilitates swift analysis of packet captures. Additionally, FlowTag allows users to export tagged flows along with the flow database and associated tagging information for further analysis or sharing. The main reason why the tool could not be analyzed is due to its codebase [66] being published 10 years ago and never updated, rendering it largely deprecated and incompatible with modern systems.

InetVis InetVis is an advanced 3-D scatter-plot visualization tool designed specifically for network traffic analysis. It functions akin to a media player, but tailored to network traffic. It proves to be highly advantageous in identifying and observing scan activities and detecting anomalous traffic patterns. In InetVis, network packets are graphically represented based on the following parameters: the destination address within the home network is plotted along the x-axis, the source address within the external Internet range is plotted along the z-axis and ports, distinguishing between TCP and UDP, are plotted along the y-axis. ICMP traffic is depicted below the TCP/UDP cube, marked by a grey/white ICMP plane. InetVis encompasses several key features that aid in the exploration of network traffic and facilitate the development of valuable insights, which collectively enhance the usability and effectiveness of InetVis as a tool for comprehensive network traffic analysis. Due to the outdated version of the original tool published in 2007 [67], there has been a recent attempt to update the application to make it available on modern systems, in 2018. However, the result achieved or, at least, the version published and currently available [68] of this work is not functional on either Windows or Linux systems.

J-Viz J-Viz [69] is a specialized security visualization tool designed to detect algorithmic complexity attacks within Java bytecode. This tool utilizes connected directed graphs derived

from the bytecode and applies a designated node ordering, known as sibling-first recursive (SFR) numbering, to visualize the graphs effectively. Specifically, the graphs used in this context are derived by employing Shiver's k-CFA framework on Java bytecode. The tool incorporates valuable connections between the nodes of the input graph and the corresponding Java bytecode that generated it. Furthermore, it provides a decompiled version of the Java bytecode for better understanding. By employing the canonical drawing paradigm, J-Viz proves to be highly effective in identifying potential security vulnerabilities associated with algorithmic complexity attacks. The utilization of J-Viz as a security visualization tool should contributes to the exploration and identification of algorithmic complexity vulnerabilities within Java bytecode, further enhancing the understanding and mitigation of potential security risks in software systems. The tool is made available through its page on SourceForge [70], but the provided source code, most likely due to its publication date, is mostly deprecated and does not allow the installation of the tool, thus rendering it unusable.

AlertWheel AlertWheel [71] is an innovative user interface that incorporates a unique radial overview visualization, along with filtering, drill-down capabilities, and the ability to save and annotate specific data subsets. The purpose of this interface is to support the workflow of network defense analysts. During the design process of AlertWheel, has been designed a novel techniques for presenting bipartite graphs effectively. Similar to other radial visualizations, AlertWheel also employs a radial layout to depict the simultaneous occurrence of various events, including their location, time, and nature. However, the specific radial layout used in AlertWheel is particularly well-suited for visualizing attacks targeted at a honeypot. This visualization approach proves valuable for network analysts, especially during the emergence of highly infectious malware outbreaks on the Internet. Currently, there is only one way to download the application, which is via its SourceForge page [72], which provides a Windows installation file. However, it appears that this file isn't working correctly.

BUCEPHALUS Bucephalus [73] is a Visual Analytics solution that offers a visual overview of the relationships between business functions, devices, and vulnerabilities. It provides users with a what-if analysis scenario, helping them make decisions on which vulnerabilities to prioritize for remediation. The tool has been developed and validated through a user-centered design process, with active involvement from security professionals. Due to contractual reasons between the research group and the involved company, the tool is not available to the public.

Fimetis The Fimetis tool [74] provides a comprehensive solution for exploring file metadata, including timestamps and ownership details, while effectively managing substantial volumes of such data. Fimetis offers a user-friendly interface with visual controls that facilitate seamless navigation through the dataset. Additionally, the analysis process benefits from a collection of predefined configurations that identify typical file patterns and operations, such as crontab rules, files with weak permissions, and indications of on-the-spot compilation. The application's user interface operates within a browser environment and leverages the indexing capabilities of ElasticSearch to efficiently process the data. Notwithstanding its relatively recent development in 2020, the provided codebase exhibits a multitude of errors, rendering the tool unusable in its current state.

V3SPA V3SPA [75] (Verification, Validation and Visualization of Security Policy Abstractions) is a tool for visualizing and analyzing SELinux and SEAndroid security policies. It addresses the challenges of policy development and maintenance by leveraging the Lobster domain-specific language to create policy abstractions. With integrated exploratory controls and filters, V3SPA enables rapid analysis of policy rules. It introduces innovative analysis modes including differential policy analysis, allowing visual comparison of policy versions, and information flow analysis for identifying unexpected permissions and examining policy design. V3SPA offers a comprehensive view of the entire policy while allowing selective filtering and reachability queries. It enhances the efficiency and effectiveness of policy authors and auditors, revolutionizing the way SELinux policies are built and analyzed. The codebase of the prototype can be accessed through its GitHub page [76]. However, despite various installation methods being documented, none of them are currently operational due to dependencies on outdated components that have not received updates in several years. Furthermore, the

developer has explicitly stated in the repository's issue section that the tool will not undergo further updates, and any installation-related challenges will remain unresolved.

CRUSOE CRUSOE [77] web application displays and illustrates the data regarding the ongoing cybersecurity status within a computer network. It provides multiple perspectives on the data gathered from other tools. Notably, it offers visual representations of an entity's context within the network and an overview of vulnerabilities and susceptible hosts present in the network. Additionally, the web application incorporates dashboards for other tools developed within the same project. These dashboards facilitate monitoring and verification of data collection and subsequent processes, offering an alternative to the default control mechanism of such tools. The source code of the application can be downloaded from the university's webpage [78] where it was developed. Additionally, the tool can be installed using Docker. However, the inability to use the application stems from the requirement of institutional credentials during its initialization, which, despite requests, have not been made available. This unavailability of the required credentials hinders users from effectively utilizing the application's functionalities.

CVExplorer CVExplorer [79] is an interactive system designed to visualize cybersecurity threats derived from the National Vulnerability Database. Its purpose is to serve as a reporting and alerting tool, contributing to the enhancement of security measures against cyber attacks and potentially reducing network vulnerabilities. The system consists of interconnected views that enable users to visualize the interrelationships among various aspects found within a vast collection of vulnerability reports. These aspects include the types and levels of vulnerabilities, vendors, and products involved. CVExplorer offers an intuitive interface and supports numerous interactive features, such as filtering and sorting vulnerability severity ratings. These functionalities allow users to swiftly narrow down their focus to specific areas of interest. As mentioned in the research paper, a link is provided for users to access a demonstrative video and an online demo of the prototype [80]. However, it is worth noting that the provided link is currently non-functional, rendering the online demo inaccessible at this time.

RiskID The RiskID [81] application utilizes visualizations to represent network connections in a graphical manner, facilitating easy comparison. As a result of two algorithms working behind the scenes, connections are actively organized and potential labels are predicted. These algorithms include a recommendation algorithm and a semi-supervised learning strategy. Combined with interactive adjustments to the user interface, they form a behavior recommendation system. According to the findings from this study, behavior recommendation enhances the quality of labels significantly. By analyzing interaction patterns, it was observed that the availability of behavior recommendation resulted in a more intuitive workflow being adopted. Even in this last case, the code available in its GitHub repository [82] is not functional due to several errors and dependencies. The development of this tool has been dormant for many years, suggesting a state of abandonment in its development.

4.3 Comparison with proprietaries tools

Assessing the direct or indirect benefits experienced by targeted users is crucial for evaluating the impact of scientific research. Recent studies focusing on addressing the "interactive visualization gap" have also contributed to the formulation of diverse design recommendations aimed at better integrating interactive visualization techniques into data science workflows [83]. In the domain of cybersecurity visualization, the adoption of user-centered design methodologies has played a significant role in the creation of visualization tools that effectively cater to the needs of target users. These tools have proven valuable for security experts and cyber operators, aiding them in addressing analytic inquiries arising from real-world scenarios they encounter. Evaluation studies have identified prevailing trends in evaluation methodologies, providing insights into how the research community measures the efficacy of proposed cybersecurity visualization techniques, systems, models, and tools. While these works are instrumental in highlighting emerging trends and guiding future directions for the development of effective tools, they fall short in providing an estimation of how research results and techniques are applied in real-world settings.

The objective of this section is to examine various options provided by enterprise-grade tools within the realm of security visualization. This analysis serves two main purposes: firstly, to offer an overview of the visualization methodologies employed, and secondly, to facilitate a comparative analysis with the research outcomes presented earlier. The selection of these tools was carried out through two distinct processes. Firstly, a comprehensive search was conducted both online and by referencing the relevant sources identified during the literature review phase of this research. Secondly, a meeting was held with the cybersecurity department at Politecnico di Torino. This dual approach to selection allowed for an extensive exploration of the available options on the internet, providing a comprehensive overview. Additionally, insights were gained into the specific tool choices made within the university context, enabling valuable feedback on the selected tools and facilitating considerations regarding their open-source counterparts.

The two tools selected in the first selection process are SolarWinds Security Event Manager [84] and IBM QRadar [85], which are two powerful security intelligence and analytics platforms used by organizations to enhance their cybersecurity capabilities.

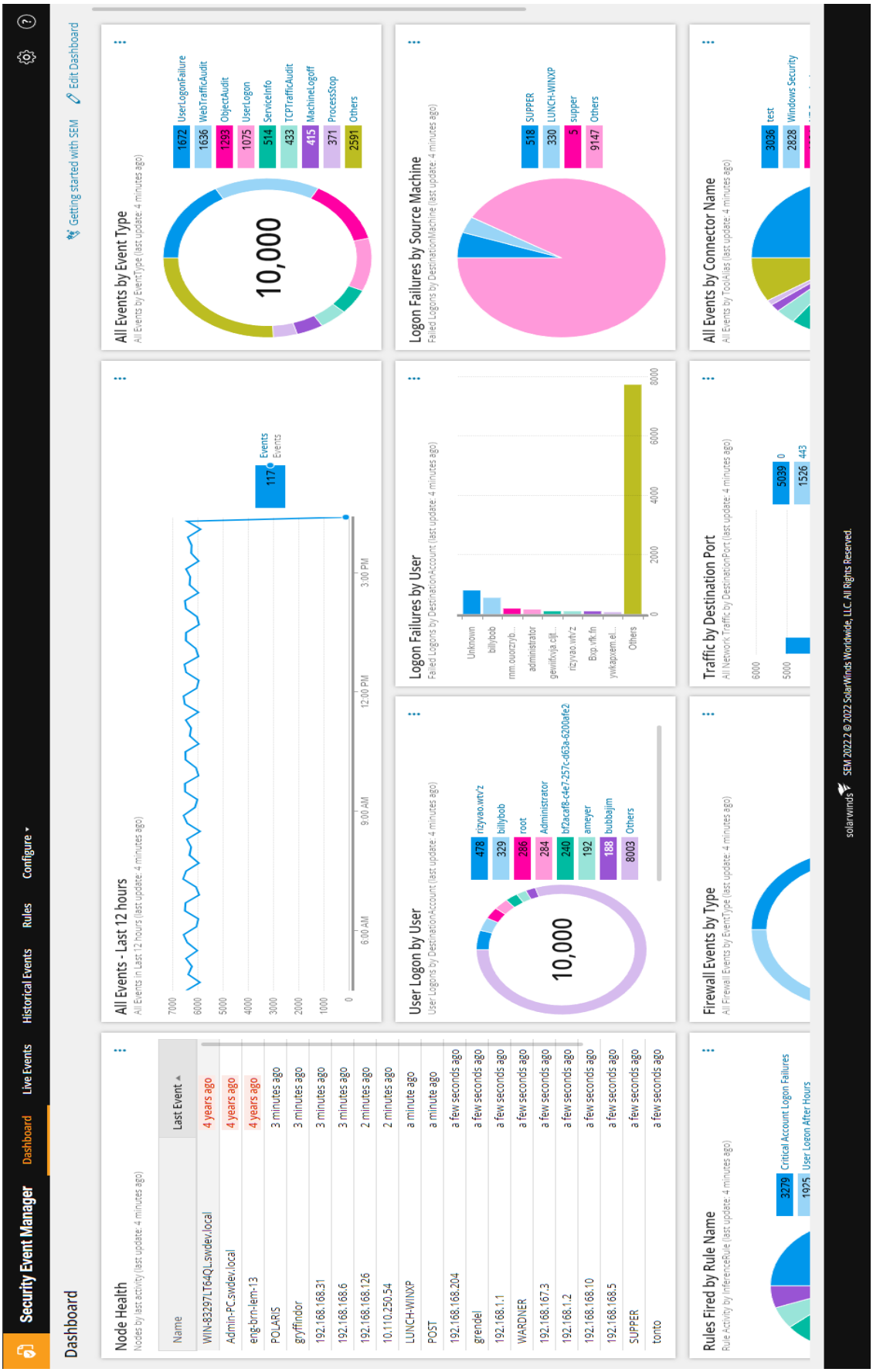


Figure 4.51: SolarWinds Security Event Manager dashboard first part

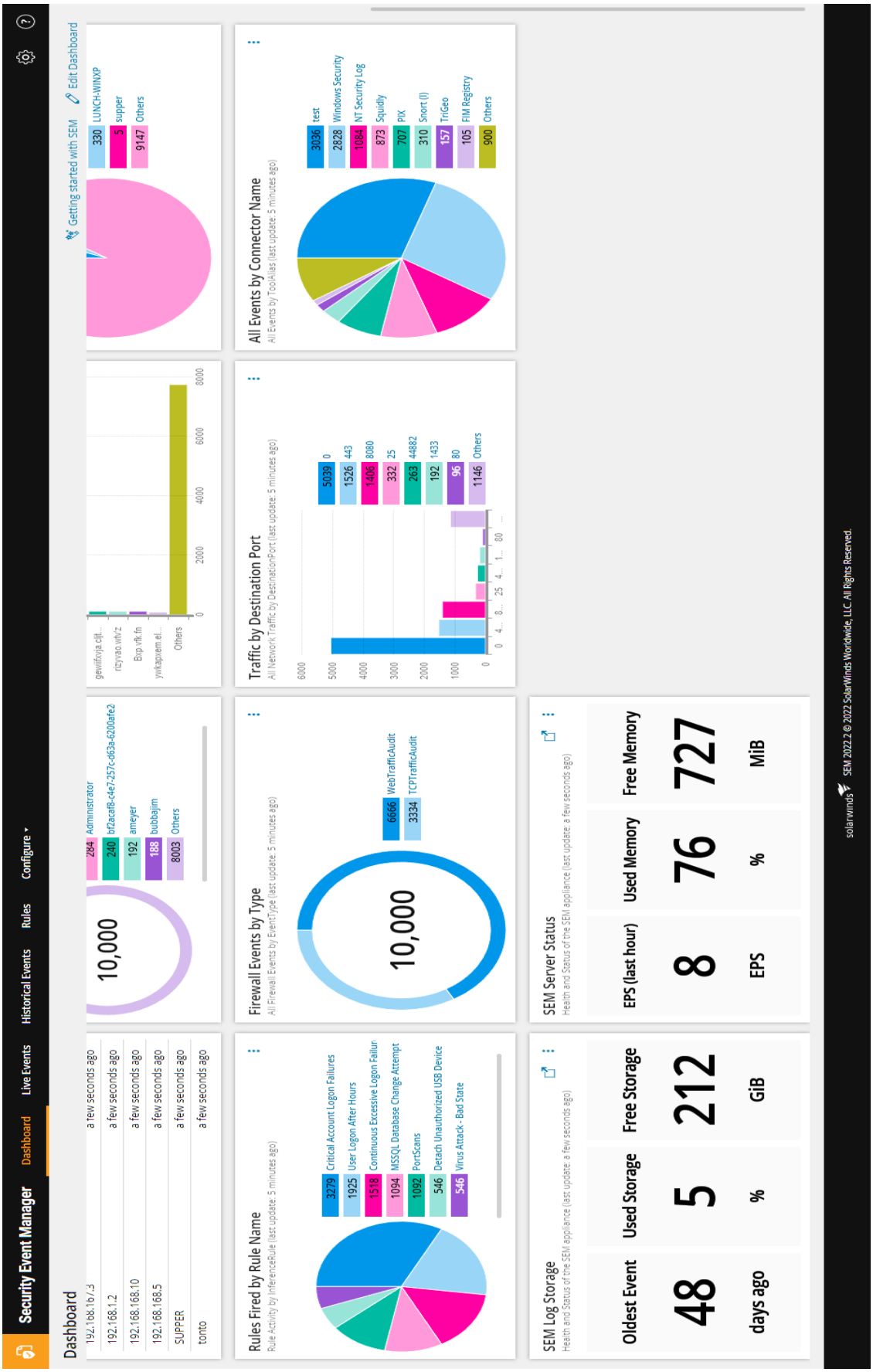


Figure 4.52: SolarWinds Security Event Manager dashboard second part

SolarWinds Security Event Manager is designed to centralize log management and provide real-time monitoring and analysis of security events. It collects logs and event data from various sources, applies correlation algorithms to identify patterns and anomalies, and generates alerts for proactive threat detection. The tool also offers log analysis capabilities and reporting features for investigating security incidents and meeting compliance requirements.

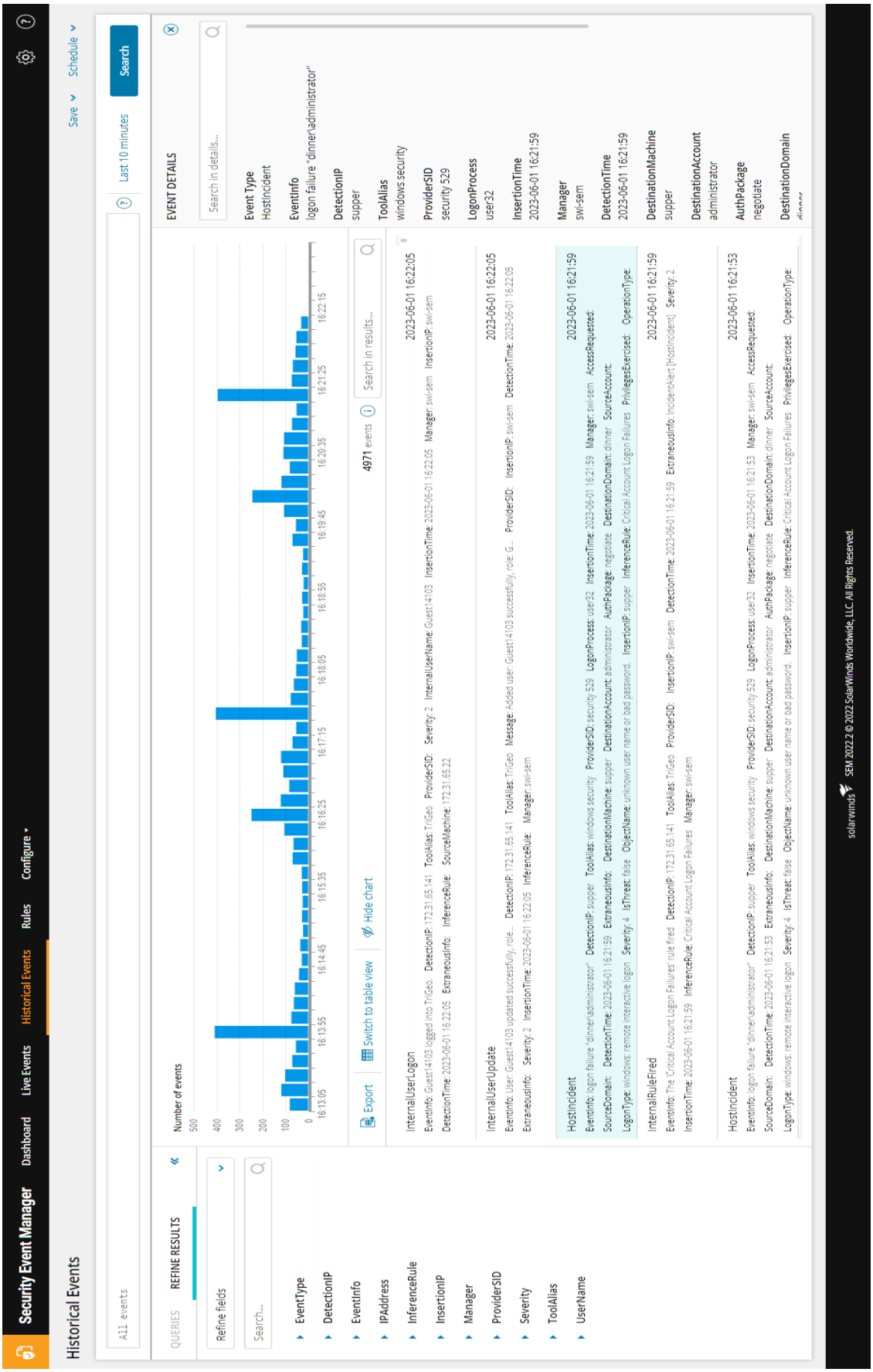


Figure 4.53: SolarWinds Security Event Manager dashboard Logs history

IBM QRadar, on the other hand, is a comprehensive security information and event management (SIEM) platform. It collects and analyzes logs and events from diverse sources, utilizing advanced analytics and machine learning algorithms to detect and prioritize threats. QRadar integrates with threat intelligence feeds to enhance its detection capabilities and includes user and entity behavior analytics features to identify anomalous user behavior. The platform also supports incident response and workflow orchestration, providing a centralized dashboard for managing security incidents and ensuring compliance with regulatory requirements.

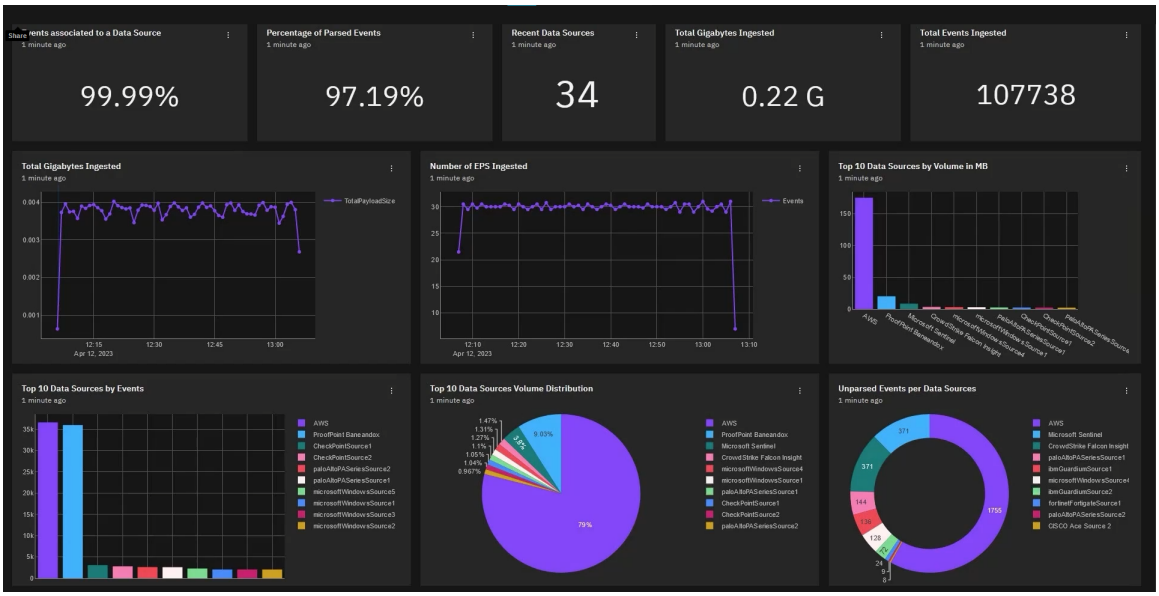


Figure 4.54: IBM QRadar dashboard first part

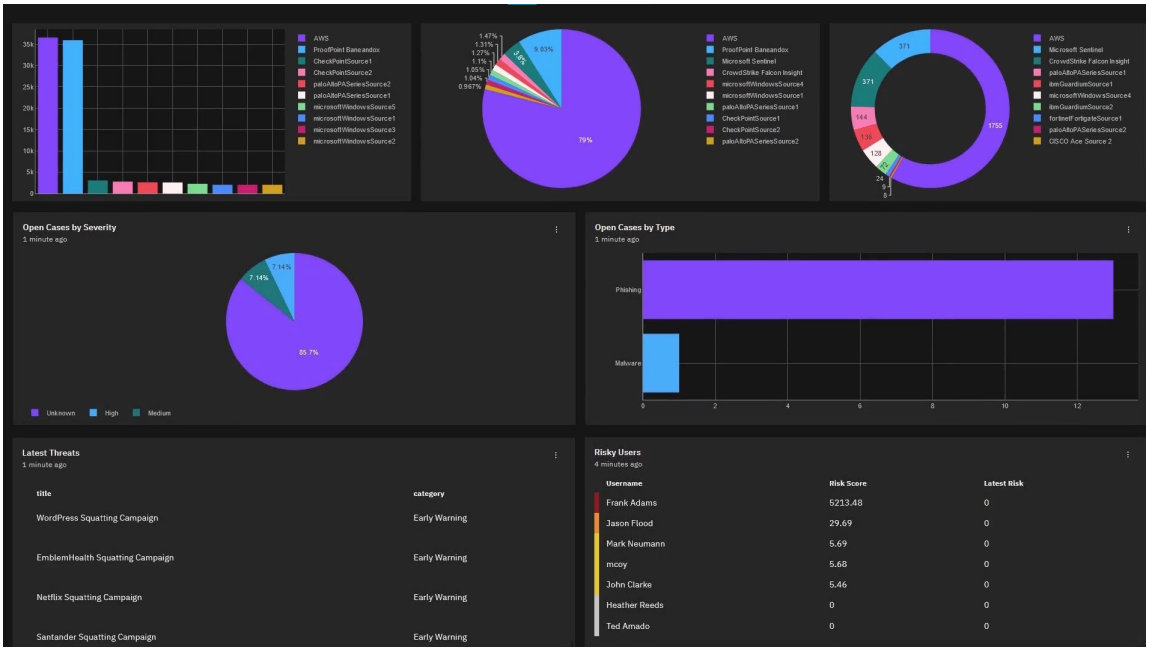


Figure 4.55: IBM QRadar dashboard second part

Both tools were examined by leveraging the availability of interactive online demo versions provided by the respective companies. This approach was chosen due to the inherent limitations associated with analyzing enterprise-scale infrastructures, as the requisite setup and data transmission processes would have entailed significant time and expertise beyond the scope of this thesis endeavor. The extracted images (figures 4.51, 4.52, 4.54, 4.55) reveal the presence of several

common elements observed in previous analyses, notably the utilization of a generic dashboard structure that offers a comprehensive overview of essential information. The visualizations employed in commercial products tend to adhere to conventional and widely-adopted methodologies, featuring classic representations such as histograms, pie charts, and line charts. In relation to more intricate visualization types or unconventional data representations, the findings indicate a scarcity of such instances. This can be ascribed, partially, to the tools' need to cater to a broad range of organizational contexts, thereby prioritizing more universally applicable visualizations. Furthermore, specific analysis requirements may not necessitate the incorporation of certain visualization techniques within the examined tools. Notably, there are discernible variations in visualizations for instance, SolarWinds offers a supportive Timeline for analyzing textual logs (figure 4.53), while QRadar presents its version of the attack graph to facilitate threat investigation phases (figure 4.56).

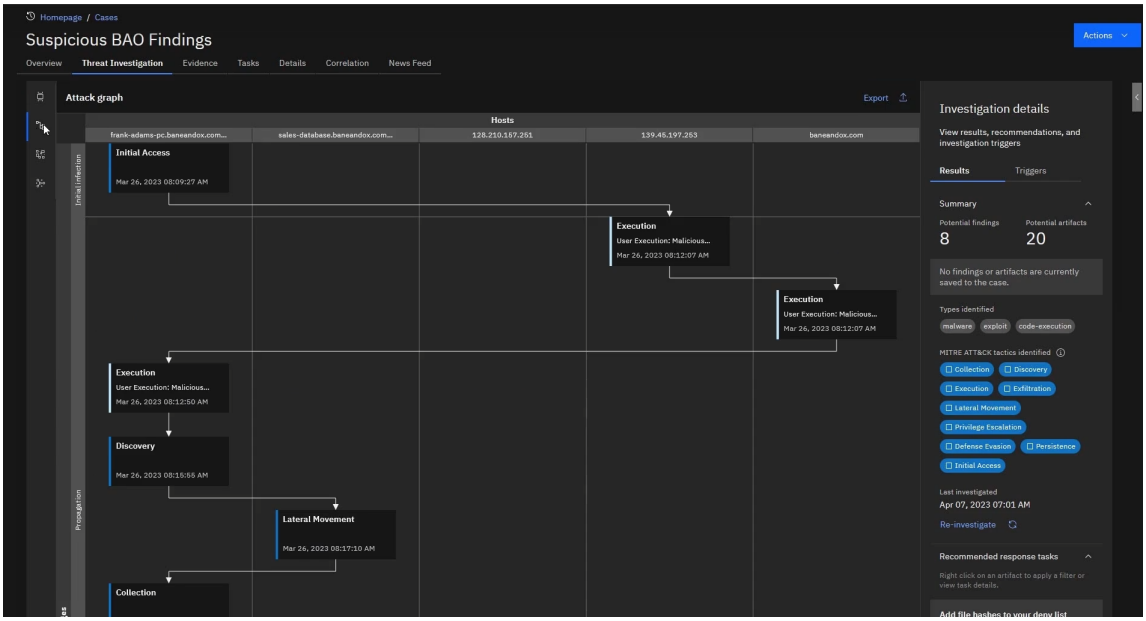


Figure 4.56: IBM QRadar attack graph

Upon examining the security analysis approaches employed by the Politecnico di Torino, this analysis focuses on the presentation of two solutions, one from the industry sector called Cycognito and the other from the open-source domain known as ntop, both incorporating visual elements. Cycognito [86] focuses on proactive attack surface management, offering continuous visibility into an organization's attack surface. It actively maps and profiles external assets, identifying potential security vulnerabilities and misconfigurations. With automated scanning and manual testing techniques, Cycognito assesses an organization's security posture, performs attack simulations, and validates the effectiveness of security controls.

Figure 4.57: Cycognito dashboard overview part1

Figure 4.58: Cycognito dashboard overview part2

On the other hand, ntop (which is available as both an open-source version and a commercial version) [87] specializes in network traffic monitoring and analysis. It captures and analyzes network traffic in real-time, providing administrators with visibility into network usage and performance. Ntop generates reports and statistics on bandwidth usage, top talkers, and network protocols, allowing organizations to understand traffic patterns, identify potential bottlenecks, and make informed decisions.

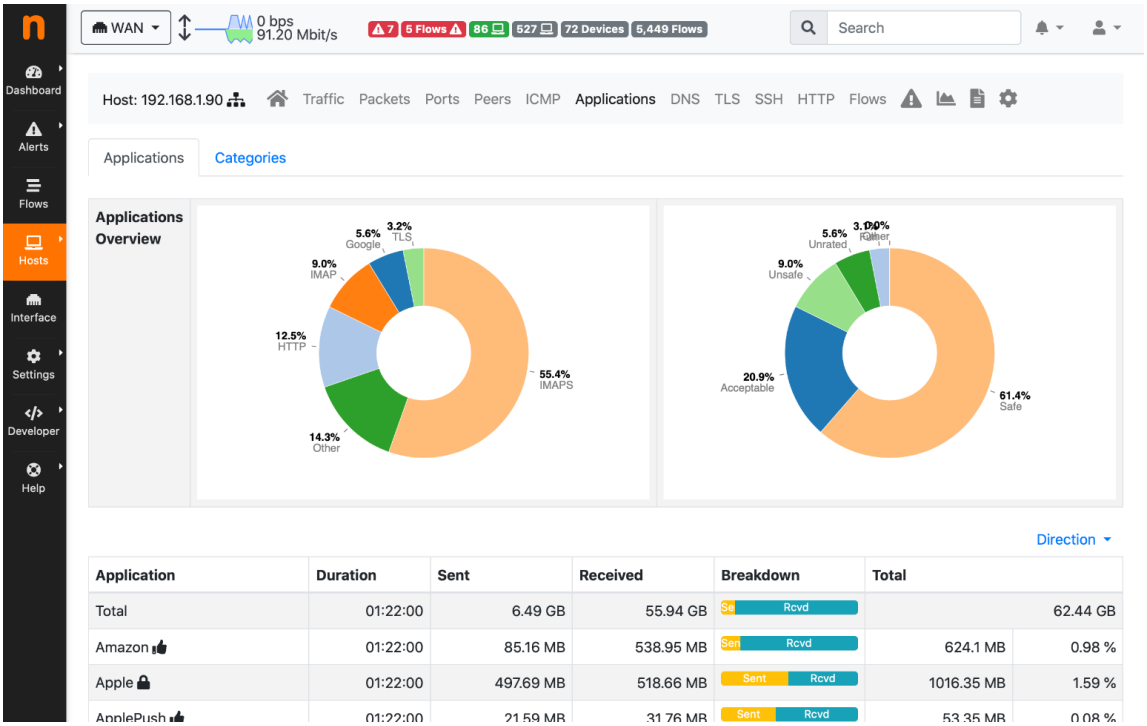


Figure 4.59: Ntop dashboard overview

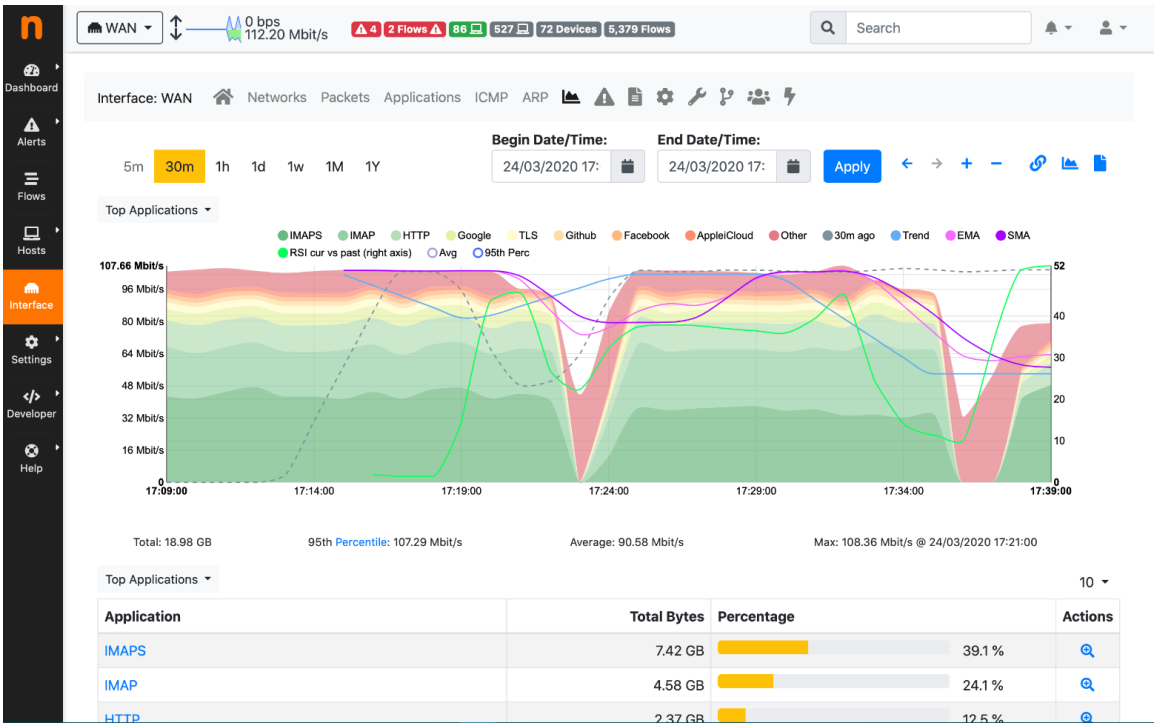


Figure 4.60: Ntop traffic view

Similarly, no distinctive visualization methodologies are observed in these cases, with the mentioned types of charts being prevalent once again. However, it is worth noting that ntop offers extensive customization options, enabling users to create a tailored dashboard for analyzing their network infrastructure. In addition to providing an overview of the adopted solutions, the insights gathered from discussions with industry experts have shed light on their perspectives regarding

the selected tools. One of the primary factors that influenced the preference for commercially-provided applications over open-source alternatives is the availability of comprehensive support services. These services encompass not only assistance with configuration processes but also prompt resolution of any encountered issues throughout the contractual period. The analysts emphasized the importance of such support services, as they enable organizations to leverage the tools more effectively within their existing infrastructures. Moreover, the experts highlighted that the configuration process for ntop, an open-source tool, was notably more complex compared to its commercial counterparts. The availability of dedicated support services provides added convenience and a different temporal dimension, both of which hold significant value in the realm of security data monitoring. Another significant finding discussed during the meeting, which supports the research findings [83], pertains to the dissemination of the extracted analysis information. Irrespective of the intricacy of the employed visualizations during the analysis phase, it was observed that visual elements are not extensively utilized in the subsequent reporting phase. This can be attributed to the perception that visual representations often require additional effort and training from the intended recipients of the reports, rendering them less helpful in practice. Consequently, a more conventional reporting approach, such as textual summaries or tabular representations, is commonly favored to effectively communicate the pertinent findings and insights derived from the analyses.

Another noteworthy aspect of interest in the capabilities of enterprise tools is the increasing trend towards offering cloud-based services that eliminate the need for local resource allocation. The ability to avoid resource utilization within the monitored infrastructure is undoubtedly advantageous in terms of performance. However, it is essential to consider that these cloud solutions, despite their significant benefits, are accompanied by variable costs depending on the scale of the infrastructure to be covered. Pricing structures typically range from a few thousand euros to several tens of thousands, depending on the specific requirements and scope of the deployment.

4.4 Security visualisation for general practitioners

In this chapter, is addressed an important gap in the existing literature regarding the application of security visualization for non-expert users, specifically focusing on general practitioners. While numerous studies have explored security visualization in the context of cybersecurity professionals and experts, there is a notable absence of research dedicated to understanding the needs and challenges faced by non-expert users in the field of cybersecurity. By not addressing the specific needs and challenges faced by non-expert users, we risk leaving a significant portion of users vulnerable to cybersecurity threats and lacking the necessary tools to make informed decisions. One of the key implications of this gap is the potential for a lack of awareness and understanding among non-expert users regarding the importance of cybersecurity. Without effective visualization techniques tailored to their needs, general practitioners may struggle to grasp the complexities of cybersecurity threats and the potential impact on their practices. This can lead to a lack of proactive measures and an increased susceptibility to cyber attacks.

In the research conducted by Legg [88], for instance, the focus is on enhancing cybersecurity situation awareness for non-expert users through the use of visual analytics. The study acknowledges that non-expert users may be inclined to take an active role in understanding cybersecurity if appropriate tools are available to better comprehend their cyber activity. To address this, the researchers developed an initial prototype tool called ePSA (Enhanced Personal Situation Awareness), which consists in two main components: one for collecting and storing network activity data, and another for analyzing and visualizing this data. The goal of ePSA is to provide non-expert users with a comprehensive overview of their online activity and promote their awareness of potential cybersecurity threats. Subsequently, a small-scale survey was conducted to gather insights from non-expert users regarding their cybersecurity concerns, whose results revealed that a significant portion of participants expressed worry about their cybersecurity when accessing the internet, but however, many participants lacked awareness of how their devices communicate online. The research emphasizes the importance of visualizations and interactions in helping non-expert users filter and understand potentially interesting activity, demonstrating how by providing intuitive representations of network data, these visualizations can encourage users to engage with cybersecurity tools and enhance their awareness of online activity.

Another noteworthy study, conducted by Sturdee et al. [89], focuses on how concepts related to cybersecurity are conceptualized and interpreted by non-expert users. Their research aim to explore the use of non-expert sketches as a tool for understanding cybersecurity concepts. The study involved collecting sketches from non-expert participants based on themes of risk, privacy, trust, and cybersecurity. The sketches were analyzed using a qualitative approach, and the findings were categorized into themes and sub-themes, and the study found that non-expert sketches can provide valuable insights into how people perceive and understand cybersecurity concepts. The analyzed sketches revealed common themes and sub-themes related to risk, privacy, trust, and cybersecurity, such as the use of physical dangers to depict risk and the depiction of humans to represent trust. The study also found that non-expert representations can help identify gaps in cybersecurity knowledge and understanding among the general public. The conclusion of the research suggests that non-expert representations of cybersecurity concepts can be a useful tool for education and awareness and recommends that cybersecurity professionals and educators incorporate similar methodologies into their training and awareness programs to improve public understanding of cybersecurity concepts. In the end the study also suggests that future research should explore the use of non-expert conceptual metaphors in other areas of cybersecurity, such as threat modeling and risk assessment.

These recommendations for future research directions are supported not only by individual studies but also by surveys focusing on security visualization [12]. Specifically, among the key target personas identified, the "higher-level decision maker" stands out as a critical stakeholder who necessitates a heightened level of cybersecurity awareness. In order for higher-level decision makers, including executives, to effectively and efficiently make cybersecurity decisions, it is crucial that they have access to comprehensive information about the organization's potential cybersecurity risks, opportunities, and challenges. This information should be presented in a format that is easily understandable and applicable to the business dimensions. Therefore, future research endeavors could be directed towards specifically designing cybersecurity awareness visualizations that cater to the needs of managers and higher-level decision makers. Gaining a deeper understanding of their information requirements and visualization preferences will facilitate the development of visualizations that are effective for this particular group.

In conclusion, this chapter highlights the absence of research in the literature regarding security visualization for non-expert users, specifically general practitioners, pointing out the need for further exploration and development. By addressing the specific challenges and requirements of non-expert users, we can pave the way for a more inclusive and accessible approach to cybersecurity.

Chapter 5

Discussion and Possible Solutions

5.1 Categorizations and statistics

In the field of cybersecurity, the effective management and analysis of security data are critical for identifying and responding to potential threats. Categorization plays a fundamental role in organizing and classifying cybersecurity tools based on their data, input, and processing capabilities. Additionally, statistical analysis provides valuable insights into the quantitative aspects of security data, enabling researchers and practitioners to extract meaningful information and draw conclusions. This subchapter explores the categorization of the various cybersecurity tools encountered during this research.

Categorizing cybersecurity tools is essential for understanding their functionalities, capabilities, and suitability for specific security tasks. A systematic categorization framework allows cybersecurity professionals to assess and compare various tools, facilitating informed decision-making in tool selection. The categorization process typically involves grouping tools based on their data types, input sources, and processing techniques. The initial categorization put forth pertains to the nature of input and its associated semantic interpretation, utilized by different tools to perform their functionality (table 5.1). Both the tested and unavailable tools are included in both this and the following categorizations, with clear separation by a line. Additionally, when applicable, it is duly noted the lack of utility in considering certain tools for specific categorizations.

Based on the preliminary analysis of the input table presented, it is apparent that the initial set of tools, which have been previously characterized as more general in nature, exhibit input types and data representations that are also of a broader scope, rather than being specifically tailored to security visualization. However, according to the output categorization presented in the table 5.2, it is noteworthy that, except for one particular tool included in this study, these tools uniquely offer the functionality to download the computed visualizations. This capability is significant in facilitating communication and information sharing pertaining to the analysis process. Regarding the more specific tools developed in the field of security visualization, it is evident that there is a clear trend in network analysis to utilize the widely recognized .pcap standard, as discussed in previous chapters, while in other cases, more general formats such as .JSON are employed for data representation. It is important to mention that in two specific cases, namely Vulnex and Code Pulse (see the description for the second phase of the analysis), the input procedure is automated and does not require a specific format to be manually entered, but instead, it involves a configuration process that can vary in complexity depending on the development context. As mentioned earlier, regarding the output, we can observe that the majority of the tools do not offer options for exporting the created visualizations. The only exception is SAGE, which differs from other tools as it does not provide real-time visualizations but calculates various visualizations and directly outputs the results. It is worth noting that ROPMate's output, as indicated in the table, is in the form of source code, which implies the possibility of obtaining the code created through the visualization of gadgets, providing a textual output rather than a visual one, while the RedEye's output is the same as the input.

NAME	INPUT TYPE	SEMANTIC
GraphViz	.dot file	Graph description
Tulip	.csv, .bib, .JSON	Relational data
Gephi	.csv, .sqlite, .GEXF	Graph description
Rumint	.pcap	Network packets
Code Pulse	.java, .NET	Application bundle
Pixel Carpet	.JSON	Log file
Visual Decision Support	No specific format	Backup file
Vulnex	Steady backend	Vulnerability analysis
NetCapVis	.pcap	Network packets
ROPMate	.JSON	Gadgets file
Vulnus	.csv or .html	Nessus scan
Crumbs	N/A	No input
RedEye	.sqlite, .redeye	Cobal Strike log
SAGE	.JSON	Suricata raw alert
Flowtag	.pcap	Network packets
InetVis	.pcap	Network packets
J-Viz	.java	Source Code
Alert Wheel	No specific format	IDS Alerts
BUCEPHALUS	No specific format	Business functions, devices, vulnerabilities
Fimetis	No specific format	System description
V3SPA	.zip	Security policies
CVExplorer	N/A	No input
Crusoe	No specific format	Log file, Network packets
RiskID	.pcap	Network packets

Table 5.1: Tools classified by input type and relative semantic

A subsequent categorization, still related to the input data provided to the applications, concerns the potential data processing that these data undergo before being visualized. In the provided table 5.3, we can observe a nearly equal division between the tools that process the data and the others. Furthermore, it is worth highlighting the wide variety of data processing techniques, with some clearly defined in their respective papers, while others are more generic and therefore their complexity is not assessable.

Another significant aspect related to the data utilized by the tools pertains to their specific purpose within the realm of cybersecurity. The table 5.4 showcases the aforementioned tools, organized and classified based on their distinct "security goal". As elucidated earlier through the input categorization, a predominant category observed is that of Network Analysis. Additionally, noteworthy categories encompass vulnerability and threat analysis, forensic analysis, and, to a lesser extent, security administration tools. These delineations align with the current state of the field, illustrating the prevailing emphasis on these particular domains within cybersecurity research and practice [11]. Within the same table, further classifications are provided concerning potential synergies that can arise among the various tools or alternative options that one tool may offer in comparison to another. It is worth emphasizing that, in this regard, the scope of consideration extends beyond the available and tested tools employed in the research, but the analysis encompasses potential relationships and alternatives, even among tools that were not accessible or directly examined during the study. When it comes to selecting among the available alternatives, a crucial factor to consider is the alignment of each tool with the specific security goal at hand. It is imperative that a viable alternative operates within the same domain as the tool being replaced and does not impose any significant limitations. However, achieving such alignment is not always feasible. For instance, in the case of ROPMate and SAGE, both tools fall under the same security goal category, yet upon closer scrutiny, ROPMate proves more suitable for analyzing application development aspects, while SAGE excels in examining the potential threats within a networked system. In the context of synergies, the term encompasses various forms of collaboration among applications, expanding their individual scopes of operation. However, delineating the precise interactions between two tools can pose a complex task.

NAME	OUTPUT TYPE	SEMANTIC
GraphViz	.jpeg, .png and more	Graphic and data formats for end user
Tulip	.jpeg, .png and more	Graphic and data formats for end user
Gephi	.jpeg, .png and more	Graphic and data formats for end user
Rumint	No output	No exportable output
Code Pulse	No output	No exportable output
Pixel Carpet	No output	No exportable output
Visual Decision Support	No output	No exportable output
Vulnex	No output	No exportable output
NetCapVis	No output	No exportable output
ROPMate	.py	Python code
Vulnus	No output	No exportable output
Crumbs	No output	No exportable output
RedEye	.sqlite, .redeye	Modified Cobalt Strike log
SAGE	.png	Attack graphs images
Flowtag	No output	No exportable output
InetVis	No output	No exportable output
J-Viz	No output	No exportable output
Alert Wheel	No output	No exportable output
BUCEPHALUS	No output	No exportable output
Fimetis	No output	No exportable output
V3SPA	No output	No exportable output
CVExplorer	No output	No exportable output
Crusoe	No output	No exportable output
RiskID	No output	No exportable output

Table 5.2: Tools classified by output type and relative semantic

NAME	PROCESSING
GraphViz	None
Tulip	None
Gephi	Different Layout algorithms
Rumint	None
Code Pulse	None
Pixel Carpet	None
Visual Decision Support	Different sources of combined, analyzed and filtered inputs
Vulnex	None
NetCapVis	Filtering and data reduction techniques
ROPMate	Analysis on gadgets extracted from the application
Vulnus	None
Crumbs	None
RedEye	Not specified
SAGE	Suffix-based probabilistic deterministic finite automaton
Flowtag	None
InetVis	None
J-Viz	SFR search tree construction
Alert Wheel	None
BUCEPHALUS	Different input combination
Fimetis	SFR search tree construction
V3SPA	Timeline calculation
CVExplorer	None
Crusoe	Different input combination
RiskID	Network pattern extractor

Table 5.3: Tools classified by data processing

NAME	SECURITY GOAL	SINERGY	ALTERNATIVES
GraphViz	Generic Analysis	All generic analysis	All generic analysis
Tulip	Generic Analysis	All generic analysis	All generic analysis
Gephi	Generic Analysis	All generic analysis	All generic analysis
Rumint	Network Analysis		FlowTag
Code Pulse	Vulnerability Analysis	Vulnex	
Pixel Carpet	Threat Analysis	VDS, Fimetis	
Visual Decision Support	Forensics Analysis	Pixel Carpet	Fimetis
Vulnex	Vulnerability Analysis	Code Pulse, CVExplorer	
NetCapVis	Network Analysis		InetVis
ROPMate	Threat Analysis		
Vulnus	Vulnerability Analysis	CVExplorer	
Crumbs	Security Administration		
RedEye	Threat Analysis		
SAGE	Threat Analysis		
Flowtag	Network Analysis		Rumint
InetVis	Network Analysis		NetCapVis
J-Viz	Vulnerability Analysis		
Alert Wheel	Network Analysis		
BUCEPHALUS	Vulnerability Analysis		
Fimetis	Forensics Analysis	Pixel Carpet	VDS
V3SPA	Security Administration		
CVExplorer	Vulnerability Analysis	Vulnex,Vulnus	
Crusoe	Threat Analysis		
RiskID	Network Analysis		

Table 5.4: Tools classified by security goals, sinergy and alternatives

An illustrative example of synergy can be observed in the amalgamation of forensic analysis tools and Pixel Carpet. This synergy arises from the capability of Pixel Carpet to scrutinize suspicious log files identified through the prior analysis conducted by the aforementioned tools, which themselves can be regarded as viable alternatives. It is important to note that such synergy does not entail direct interaction between the outcomes of the tools but rather implies that the effect of one tool's analysis may prompt the utilization of the other for distinct analytical purposes. Another compelling example within the same domain is demonstrated by the synergy between Vulnex and Code Pulse. Both tools operate within the realm of code vulnerability analysis, and their combined deployment could furnish a diverse yet mutually reinforcing set of analytical insights, thereby augmenting the overall depth of the undertaken analysis. An additional and distinct perspective can be offered for CVExplorer, which could provide external support for various tools that necessitate the exploration of information related to Common Vulnerabilities and Exposures (CVEs).

As evident from the table, there are few instances of synergies or proposed alternatives among the tools. This lack of correlation can be primarily attributed to two limitations identified during the analysis of tool development in the field of security visualization. The first limitation pertains to the number and timeframe of the study conducted for this thesis work. The volume of research in the field of Security Visualization grows annually, driven by the research community's efforts [12]. As a result, achieving a comprehensive catalog of various theoretical and development papers is a resource-intensive endeavor. Despite analyzing a considerable number of papers in this study, the coverage cannot be deemed exhaustive in terms of the available possibilities. A second limitation stems from the fact that the methodologies and products developed are not designed to operate in a collaborative context with other tools. The objective is not to create something that integrates with existing tools, but rather to present an idea or implementation within a specific and limited context. Consequently, most of the tools are in a prototypical state, geared toward demonstrating a concept or solution in a particular domain rather than facilitating integration with other tools. Thus, the scarcity of identified synergies or alternatives

in the examined tools can be attributed to the combination of limited research scope and the inherent nature of the developed methodologies and products, which prioritize specific contextual applicability over comprehensive collaboration with other tools.

In summarizing the various tools based on the different development phases encountered in their descriptions, we obtain an overview presented in the table 5.5. For convenience, the phases presented in 3.1 have been denoted using abbreviated terminology, and the symbols used correspond to the following: Y = phase present, N = phase not present, and / = information not available. It is evident that the majority of encountered cases have considered all development phases in the presentation of their applications. A more thorough analysis of the categorization reveals that the Exploration Phase and the Feedback and Fine-Tuning Phase are less frequently represented. Regarding the Exploration Phase, it can be observed that in certain cases, its inclusion was deemed unnecessary as the selection of visualization methodologies favored simplified and immediate choices, rather than elaborate decision-making processes. In other instances, the selection of components was influenced by the requirement to adhere to specific standards, as exemplified by Vulnex. Concerning the Feedback and Fine-Tuning Phase, the lack of specific details or information in the descriptions of tools, particularly within the framework presented by Balakrishnan [5], makes it challenging to establish a standardized procedure for all development processes. A dedicated section in a succeding chapter will be allocated to provide a comprehensive examination of this phase in subsequent sections.

NAME	GOALS PHASE	DATA PHASE	EXPLOR. PHASE	VISUAL. PHASE	FEEDBACK PHASE
GraphViz	/	/	/	/	/
Tulip	/	/	/	/	/
Gephi	/	/	/	/	/
Rumint	/	/	/	/	/
Code Pulse	Y	Y	Y	Y	Y
Pixel Carpet	Y	N	Y	Y	Y
Visual Decision Support	Y	Y	Y	Y	Y
Vulnex	Y	Y	N	N	Y
NetCapVis	Y	Y	N	N	Y
ROPMate	Y	Y	Y	Y	Y
Vulnus	Y	N	Y	Y	Y
Crumbs	/	/	/	/	/
RedEye	/	/	/	/	/
SAGE	Y	Y	N	N	N
Flowtag	/	/	/	/	/
InetVis	/	/	/	/	/
J-Viz	Y	N	N	Y	N
Alert Wheel	Y	Y	Y	Y	N
BUCEPHALUS	Y	Y	Y	Y	Y
Fimetis	Y	Y	Y	Y	Y
V3SPA	Y	Y	Y	Y	N
CVExplorer	Y	Y	Y	Y	N
Crusoe	Y	Y	Y	Y	N
RiskID	Y	Y	Y	Y	Y

Table 5.5: Tools classified by SANS phases covered

Lastly, an additional categorization is introduced, which involves evaluating the effectiveness of the proposed solutions and providing a comprehensive overview of the availability of the various tools (table 5.6). As extensively elucidated in the tool descriptions, a significant proportion of the cases examined are in a prototypical phase, thereby underscoring the constrained range of options within the domain of open-source Security Visualization. In relation to the more generic solutions, despite their strong support and regular updates, they were not deemed to have a substantial impact within the realm of security visualization due to their inherent generality. Conversely, the solutions specifically developed to harness visualization capabilities in support of

cybersecurity exhibited a notably higher impact, generally warranting positive evaluations. Regrettably, their limited availability presents a significant hurdle, often impeding the full utilization of their functionalities and the application of their methodological approaches by analysts. As for the unavailable tools, they were excluded from the categorization due to their inaccessibility for practical analysis.

NAME	JUDGMENT	AVAILABILITY
GraphViz	Weak	Available
Tulip	Weak	Available
Gephi	Weak	Available
Rumint	Medium	Available
Code Pulse	Good	Available
Pixel Carpet	Medium	Prototype
Visual Decision Support	Medium	Prototype
Vulnex	Medium	Prototype
NetCapVis	Medium	Prototype
ROPMate	Good	Prototype
Vulnus	Good	Prototype
Crumbs	Medium	Prototype
RedEye	Good	Available
SAGE	Good	Available
Flowtag	/	Not Available
InetVis	/	Not Available
J-Viz	/	Not Available
Alert Wheel	/	Not Available
BUCEPHALUS	/	Not Available
Fimetis	/	Not Available
V3SPA	/	Not Available
CVExplorer	/	Not Available
Crusoe	/	Not Available
RiskID	/	Not Available

Table 5.6: Tools classified based on final judgment and availability

5.2 Gaps evaluations

In addition to categorizing cybersecurity tools, it is also important to identify gaps in security measures tested and propose solutions to address them. This subchapter explores the process of evaluating gaps in cybersecurity visualizations and provides possible solutions for mitigating these gaps between theory and practice. By identifying and addressing these gaps in security, organizations and individuals can better protect themselves against potential threats and minimize the impact of security breaches.

The initial category under examination focuses on the disparity between the ideal objectives of security visualization and the actual outcomes attained, highlighting the presence of missing or inadequately developed elements that warrant further investigation. Upon analyzing the objectives outlined in Chapter 3.3, it becomes apparent that a significant portion of both research endeavors and real-world implementations has predominantly centered around the first two objectives , namely **Understanding security data** and **Detecting and responding to security incidents**, in fact a considerable emphasis has been placed on devising diverse solutions to facilitate the comprehension of security-related data.

However, the treatment of the remaining objectives has been comparatively limited, leading to evident gaps, as seen in the case of **Effectively communicating security information**. Despite the widespread recognition of the significance of information exchange in security and the existing research identifying it as a promising area for further exploration [12], it has been observed, in

support of what has been expressed by expert feedback, that the majority of methodologies do not adequately facilitate information exchange using visualizations to enhance communication and data comprehension both between experts of the same level and between different stakeholders, such as security experts who communicate with business executives. Another compelling piece of evidence supporting this observation can be derived from Table 5.2, which demonstrates that, in most cases, the ability to extract the produced visualization, even in the standard version intended for experts, has not been adequately developed. This discrepancy underscores a prevailing emphasis on data comprehension rather than effective data dissemination, resulting in an initial gap that could hinder the actual understanding and subsequent mitigation of the identified security issues. Hence, it can be asserted that collaboration and information sharing should be regarded as indispensable components of security visualizations in both research and real-world contexts, in future advancements within the field.

When considering another objective previously defined as **Supporting decision-making**, it is noteworthy to observe that this characteristic is evident in certain open-source prototype solutions, specifically Vulnus, which offer a range of strategies to assist analyst users in their decision-making processes. This supportive feature facilitates the comparative analysis of diverse outcomes derived from multiple strategies, thereby providing an comprehensive overview of the potential system states, which is crucial in evaluating the various ramifications that could ensue. In contrast to the research domain where this feature is partially present, our examination of real-world applications, specifically enterprise tools, within the scope of our study reveals a notable absence of this characteristic. In the majority of cases, these tools primarily offer ordered information without providing calculated strategies or insights into the ensuing system state, thereby lacking active support for analysts. Therefore, it is apparent that solutions specifically designed for organizations and industries should integrate these visualization methodologies, aiming to provide a form of support that utilizes advanced data analysis techniques derived from AI and ML, as well as contemporary visualization methods that encompass that range from Vulnus' proposed solution to the realm of visual simulations [90].

Upon closer examination and comparison between the theoretical realm, which encompasses the described or hypothesized possibilities derived from research, and the practical realm, which entails the utilization of currently available resources in real-world scenarios among the various analyzed alternatives, it becomes evident that there are some disparities. Specifically, in the practical domain, the predominant visualization techniques and methodologies employed are largely characterized by simplistic representations of standard graphs, which highlights a notable gap in the utilization of recent research advancements, leading to the potential use of suboptimal visualizations that do not effectively fulfill their intended purpose. As the complexity and sophistication of cybersecurity threats continue to evolve, it is crucial to employ visualizations that can capture and convey the intricacies of these threats. Relying solely on standard graphs may lead to oversimplification or incomplete representations, making it challenging for stakeholders, including security experts and business executives, to fully grasp the nuances of the security landscape. To address the gap between theoretical possibilities and practical implementations in security visualization, several strategies can be pursued, like foster closer collaboration and knowledge sharing between researchers and practitioners in the field of security visualization, a choice already widely used on the research side, which can help bridge the gap by incorporating real-world challenges, practical constraints, and industry insights into research efforts, or as evidenced by theoretical research, adopt a user-centered design approach to develop security visualization tools that address the specific needs, requirements, and cognitive abilities of different stakeholders, even in the case of distributed tools for large organizations offering a multitude of services.

Furthermore, it is crucial to examine the potential disparities between the products developed within a research context and their practical adoption. As elucidated by the valuable input provided by experts from Polito, there is a prevailing inclination towards favoring commercial tools over open-source alternatives, nevertheless this inclination does not solely stem from the factors elucidated in the preceding chapter but is also underpinned by a multitude of factors underscored by numerous surveys conducted in diverse research works. The findings obtained from a study examining decision-making processes in corporate environments have revealed significant insights that not only corroborate the observations made during the Politecnico encounter but also provide additional motivations for the widening gap between research products and their

implementation in practice [83]. For example, also this analysis underscores the impact of installation and setup requirements on analysts' decision-making regarding visualization tools, but furthermore, it provides additional insights such as personal constraints in certain environments that limit the adoption of alternative solutions outside of predefined corporate standards, as well as the impossibility of incorporating external solutions that could compromise corporate privacy.

A significant aspect that arises is the potential absence of privacy safeguards and security standards that guarantee the quality of open-source applications. Among the various options explored in this research, encompassing both available and unavailable tools, a single instance stands out where developers consistently provide status checks on the testing of their application, that is RedEye. This particular tool, developed not solely within a research context but by a group of researchers from CISA for a government-funded project, exemplifies the rarity of ongoing verification efforts. This observation is not merely based on the absence of evaluations, but it is also rooted in the lack of standardization within the research context, as previously anticipated in the exposition of the conducted theoretical research. This deficiency translates into a preference among stakeholders for established technical products that provide robust assurances. The absence of standardized evaluation frameworks contributes to the perception that open-source solutions may entail greater uncertainty and potentially lack the necessary guarantees to meet the rigorous demands of practical implementation. Consequently, this preference for established and trusted technical products perpetuates the gap between research endeavors and the adoption of solutions in real-world scenarios, hindering the integration of innovative open-source tools into practical security visualization practices.

5.3 SANS Refining Framework

In light of the preceding chapters and the identified gap concerning the absence of a standardized framework, it is important to acknowledge that Balakrishnan's proposed framework does not serve as the exclusive reference for the field of security visualization. Thus, this chapter undertakes a comprehensive exploration to scrutinize and propose necessary refinements to the different stages that compose it, which demand reevaluation. The analysis of the previously presented table 5.5 underscores a significant disparity in the emphasis given to the *Visualization* phase and the *Feedback and fine-tuning* phase within the analyses of tools developments.

Regarding the *Exploration* phase, the original proposal entailed conducting an analytical analysis to determine the optimal approach for visualizing the data. Building upon this initial premise, and considering the range of research possibilities available, it is imperative to extend this analysis to encompass existing solutions. This extension should encompass both complete solutions that are already deployable and prototypes that offer room for further extensions or modifications. The objective is to align these solutions with the predefined visualization goals established in the previous phase (*Visualization Goals* phase). By doing so, a comprehensive evaluation can be conducted to identify the most suitable visualization techniques and strategies that effectively address the requirements of security data visualization. The current state of the literature reviews conducted in research [13] presents diverse lists and categorizations of solutions, encompassing both available and non-accessible alternatives, which are described to a sufficient extent through their corresponding research papers and provide insights into the characteristics and potentials of various products. While these sources should not be regarded as exhaustive catalogs of security tools utilizing visualization techniques, they nonetheless serve as valuable resources, offering a diverse range of possibilities and expanding the knowledge base on available visualization techniques in the field of security. Leveraging these resources enables researchers and practitioners to broaden their comprehension of visualization techniques and explore novel avenues for augmenting the comprehension and analysis of security data. Nevertheless, it is crucial to acknowledge, as previously highlighted in the description of the comparison table of tools and their covered phases, that the selection process of a suitable visualization tool or methodology for an organization necessitates the consideration of various factors. These include ease of use, customization options, integration with existing security infrastructure, and compliance with specific organizational and industry standards. Additionally, potential limitations arising from the requirement to adhere to specific standards, whether internal or external, must be taken into account (As described for

example in the case of Vulnex 4.1). Hence, it would be beneficial to allocate a section within this phase to analyze the constraints that developers may encounter during the development or selection of a visualization tool. These considerations ensure that the chosen visualization solution aligns effectively with the organization's needs, objectives, and technical environment, facilitating seamless integration and optimal utilization in the context of security operations.

Transitioning to the concluding phase of the framework, namely the *Feedback* and *Fine-tune* phase, it is noteworthy that since its introduction in Chapter 3.1, there is a conspicuous absence of explicit guidance concerning this phase. This is primarily due to the contextual dependence of the available options for executing feedback-related activities within the framework, however this absence creates potential gaps that impede the practical application and adoption of these solutions in real-world settings. In many of the analyzed cases, it's evident that during this developmental phase, diverse strategies were employed, such as expert feedback on tool characteristics (for example Pixel Carpet [36]), execution of specific test cases by experts and non-experts alike to assess various methodologies or proposed solutions (for example Vulnus [50]), and the implementation of a set of simulated tasks aimed at demonstrating the efficacy of the product (for example [42]), all approaches which align with the user-centered design methodology. Subsequently, in some cases, there were modifications aimed at improving and updating the various tools. However, in all cases, the focus was primarily on feature enhancements, with little attention given to critical factors such as privacy and security. As highlighted in the chapter on gaps and supported by the research, this lack of standardization and subsequent absence of necessary validation hinder the practical adoption of such tools [12]. Therefore, it is needed to incorporate within the development process of a security visualization application the necessary functionalities that ensure both privacy and security, promoting their wider use. Some examples of main points aimed at ensuring these factors are:

- The anonymization of sensitive data. By removing or obfuscating personally identifiable information and other sensitive data, user privacy can be safeguarded, and compliance with relevant data protection regulations can be maintained.
- Controlling access to visualizations. Robust access controls and authentication mechanisms should be implemented to ensure that only authorized individuals can access the visualizations, thereby minimizing the risk of unauthorized data exposure, particularly if they contain sensitive information.
- Secure storage and transmission of data used. Employing encryption and other appropriate security measures ensures that data is protected from unauthorized access, tampering, or theft throughout its lifecycle.
- Staying informed about vulnerabilities related to visualization tools. Remaining up-to-date on known vulnerabilities or security issues enables proactive measures to be taken to mitigate risks and safeguard the integrity of the visualization environment.

In alignment with the modular approach observed in the *Data preparation* phase, where the initial data collection and subsequent data cleaning and filtering are distinct sub-phases, a similar division can be proposed for this phase. In this way, strategies can be integrated with the existing feedback mechanisms in the first phase to incorporate these features within the developed applications, enabling their applicability in contexts where they have found limited utilization thus far.

An alternative approach, diverging from incorporating these features in the *Feedback and fine tune* phase, is to introduce a distinct and novel phase with the primary objective of integrating essential security features into applications, rather than including them in the fine-tuning of the development process (figure 5.1). This proposed phase would be dedicated solely to addressing the inclusion of these necessary security components, ensuring their effective implementation and adherence to relevant standards and guidelines.

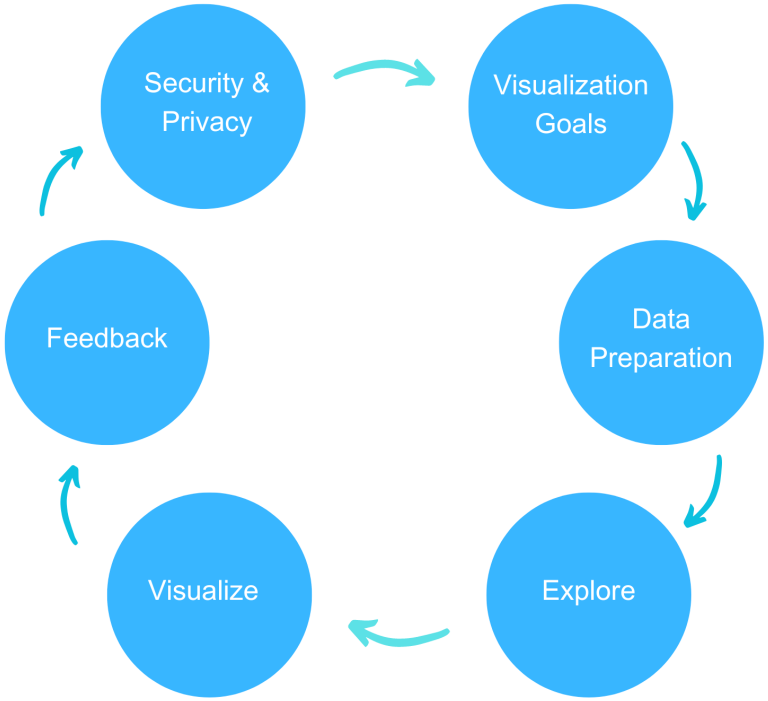


Figure 5.1: Security Data Visualization Process alternative with Security and Privacy phase

Regarding the remaining phases, namely *Visualization Goals*, *Data Preparation*, and *Visualization*, as highlighted in the summary table, they exhibit a higher presence within the development phases identified in the conducted analysis. In addition to their substantial presence, these phases do not display significant deficiencies or noticeable gaps, apart from minor additions that could be made, although of lesser importance. For instance, in the first phase, it would be beneficial to supplement the original author's recommendations with an additional user-driven methodology, alongside the goal-driven and use-case-driven approaches. This inclusion is warranted considering the established trend in the field, as evidenced by both the state of research and the outcomes achieved, further substantiating the efficacy of this user-centric approach. In the context of the second phase of data preparation, it would be appropriate to adopt security strategies, such as data anonymization previously suggested, already in this early state. This approach aligns with contemporary development methodologies employed in application development contexts, wherein the integration of security components throughout the development lifecycle is emphasized. In relation to the *Visualization* phase, no specific guidelines or recommendations are deemed necessary. This is attributed to its inherent interdependence with the preceding phase, namely Exploration, which indirectly enhances the visualization outcomes. The visualization phase benefits from the accumulated knowledge and insights acquired during the exploration phase, thus improving the overall results.

In addition to presenting the aforementioned methodology, the paper proceeds to propose a set of potential strategies for validating the proposed framework. The validation of a methodology is of paramount importance as it ensures its credibility, reliability, and applicability within the research domain or alignment with industry best practices.

Expert consultation Seeking feedback and input from subject matter experts in the field of cybersecurity and visualization. Engage with individuals who possess deep knowledge and expertise in these domains to validate the appropriateness and effectiveness of the chosen methodology. By consulting experts, it's possible to gather valuable insights and perspectives that can strengthen the credibility and robustness of the methodology. These experts can provide critical feedback on the research design, methodology implementation, and the alignment of the approach with industry best practices.

User testing Conduct user testing sessions with individuals who would be potential users of the methodology or the resulting visualizations. Involve participants who have practical experience and domain knowledge in cybersecurity to evaluate the usability, clarity, and usefulness of the methodology. Through user testing, can be gathered firsthand feedback on how well the methodology supports their decision-making processes and addresses their specific needs. This feedback can help identify any potential flaws or areas for improvement, enabling you to refine and optimize the methodology based on user perspectives.

Comparative analysis Performing a comparative analysis by comparing the methodology used in the study to other existing methodologies or frameworks in the field. Evaluating the strengths and weaknesses of each approach and assess whether the chosen methodology adequately addresses the research objectives. Identifying the unique contributions of the proposed methodology, such as novel visualization techniques or integration of user-centered design principles, that differentiate it from other approaches. This comparative analysis will provide a comprehensive understanding of the advantages and distinctiveness of your chosen methodology.

Moreover, each of these methods not only possesses individual validity but can also be employed in conjunction with the anothers, thereby offering a diversified approach. The choice of which method or combination to utilize is bestowed upon the user, as it is contingent upon the specific organizational scenarios at hand. This discretionary freedom allows for the customization of the validation process to effectively address the unique requirements and constraints of each organization. By presenting multiple options and affording users the opportunity to make informed decisions, the proposed framework demonstrates its adaptability and versatility, thus rendering it applicable across diverse contexts.

In conclusion, this chapter has proposed different modifications and enhancements to the existing framework and possible validation methods interchangeable and combinable with each other. These modifications aim to address the gaps and limitations identified in the original proposal, ultimately facilitating the practical adoption and utilization of security visualization solutions in real-world contexts. The *Exploration* phase benefits from the inclusion of existing solutions, allowing for a comprehensive evaluation of visualization techniques that effectively address the requirements of security data visualization. The *Feedback and Fine-tune* phase plays a crucial role in refining the developed applications. Incorporating privacy and security considerations within this phase ensures that the visualization tools adhere to necessary standards and guidelines, promoting their wider adoption. The *Visualization Goals*, *Data Preparation*, and *Visualization* phases are found to be robust and well-defined in the original framework. However, minor additions can be made to further improve these phases, as expressed above. Overall, the proposed modifications aim to enhance the effectiveness, security, and practicality of the framework. By addressing the identified gaps and incorporating necessary considerations, the framework becomes more comprehensive and applicable in diverse security visualization contexts.

5.4 Practical Strategies

Based on the evaluations conducted on both theoretical research and various tools within the field of security visualization, this chapter aims to outline the limited practical strategies currently available due to the predominantly prototypical status of the field (table 5.6). By synthesizing the findings from these evaluations, this chapter sheds light on the scarcity of established strategies and highlights the need for further development and exploration in the field.

The evaluations carried out have revealed that the landscape of practical strategies in security visualization is characterized by a paucity of mature and widely adopted approaches. Many existing solutions are still in prototype stages or experimental phases, hindering their practical applicability. This dearth of available strategies poses a significant challenge for researchers and practitioners seeking well-established frameworks to address the complexities and demands of the security field.

While the evaluations have identified some promising methodologies and tools, it is important to acknowledge the nascent state of their development and implementation. These emerging

strategies hold immense potential but require further refinement and validation to ensure their effectiveness and reliability in real-world scenarios. The limited availability of practical strategies underscores the pressing need for continued research and development efforts in security visualization to bridge the gap between theoretical advancements and practical applications.

In addition to the prevalent prototypical nature identified, the research has highlighted a significant limitation in the overall development state of these solutions. Consequently, the majority of these prototype tools are rendered inapplicable in most cases. As a primary concern, it is crucial to address the need for further advancements aimed at transforming these prototypes into functional applications suitable for real-world contexts. However, it is important to note that such development efforts incur high costs, making this strategy feasible only in specific cases where it is deemed necessary and viable.

Based on the identified available solutions that have exhibited superior efficacy compared to others, it is possible to ascertain potential strategies to be pursued (table 5.6). These strategies are derived from the insights garnered through the evaluation and analysis of existing options. By leveraging the strengths and accomplishments of these solutions, organizations can establish a framework for formulating and implementing their own effective strategies.

A prime example of this is the potential utilization of the Code Pulse 4.1 tool within a development and penetration testing context to accurately measure the coverage of the attack surface in real-time during both manual and automated testing. The tool's functionalities enable swift identification and feedback on the coverage of the tested application. As stated in its description, its ability to integrate with well-established development environments such as Java and .NET ensures considerable compatibility in a real-world landscape, making it a readily applicable tool in practical settings.

Shifting our focus from vulnerability analysis to threat analysis, the previous chapters have presented two solutions that demonstrate practical applicability in real-world contexts, which hold the potential to facilitate effective threat identification strategies. In the first case SAGE 4.1 is a supervised sequence learning tool that generates succinct attack graphs (AG) directly from raw intrusion alerts, without relying on prior knowledge. Its functionality allows analysts to identify vulnerable systems and gain detailed insights, enabling efficient resource organization and prioritization. Notably, the tool's compatibility with Suricata, a widely adopted intrusion detection system, enhances its usability. Furthermore, the availability of a Docker containerized version enables seamless integration into diverse existing infrastructures. These characteristics make SAGE well-suited and readily applicable in real-world contexts.

In the second case, we encounter a solution that distinguishes itself from other tools by being purposefully developed to be applicable in real-world contexts. RedEye 4.1 is an open-source analytical tool specifically designed to assist Red Teams in visualizing and reporting command and control activities. Unlike the previously discussed examples, the significance of RedEye extends beyond a specific task or a set of features. It stands out as the sole instance within this study that offers an open-source alternative capable of replacing potentially paid counterparts. This unique characteristic of RedEye makes it a noteworthy solution within the context of this research. Its open-source nature not only promotes transparency and collaboration but also provides organizations with a cost-effective option for enhancing their red teaming activities. By leveraging RedEye, analysts can effectively visualize and report command and control activities, enabling them to identify potential threats and devise appropriate countermeasures. Furthermore, the tool's flexibility and adaptability make it well-suited for integration into existing infrastructures, thereby facilitating seamless adoption and utilization. The availability of comprehensive documentation and active community support further contributes to its practicality and viability as a real-world solution.

It is important to acknowledge that the discussion thus far has focused on a subset of solutions deemed suitable for practical implementation. Another possibility lies in building one's own application, leveraging the practical and theoretical foundations laid by the tools discussed in this work. By utilizing these tools as a starting point, organizations can tailor their application to meet their specific needs and requirements. This approach offers the advantage of customization and flexibility, allowing for the integration of additional functionalities or the adaptation of existing ones. Considering the extensive prototypical state observed and the consequent inability

to offer specific features and functionalities in a real-world setting, this alternative holds validity, particularly for those seeking to circumvent the adoption of proprietary solutions. Nevertheless, it is worth noting that the research findings indicate a lack of robust open-source alternatives at present, further emphasizing the need for further development and advancement in this domain.

Chapter 6

Conclusions

The aim of this research was to provide an understanding of the use of visualizations in cybersecurity and to identify the current state of the art, the real-world effectiveness and limitations of the tools developed and used in this field. To achieve this, a mixed-methods approach was used, combining a systematic review of the literature and a practical test of available visualization tools.

The systematic review of the literature revealed that visualizations have been widely used in cybersecurity to support various tasks such as threat detection, forensic investigation, and network analysis and showed how there is currently a steadfast research support and ongoing innovation in the field, particularly concerning the development of new methodologies and and the adoption of user-design methodologies. The analysis of the literature has also enabled the identification and definition of ideal objectives for security visualization, which can be summarized as providing an in-depth understanding of security data, detecting and responding to security incidents, effectively communicating security information, supporting decision-making, and enhancing security awareness among users and system operators.

In addition to the literature analysis, a practical analysis was also conducted, selecting the available solutions encountered during the review to evaluate their actual effectiveness in handling security data. Furthermore, proprietary tools from the current real-world scenario were selected to enable a comparison between the two approaches.

The results of the study showed that visualizations can be effective in enhancing cybersecurity by providing insights into complex data and facilitating decision-making spanning across various fields within the realm of security. However, the comparison with proprietary tools has highlighted that the latter are still preferable in terms of adoption. This preference stems from the prototypical nature of the majority of solutions encountered in the research, as well as the potential dedicated support and services that these proprietary solutions can offer. Furthermore, it was possible to highlight how current solutions in the market do not leverage the research findings in terms of innovative visualization methodologies, indicating the potential for improvements derived from such research.

The study also identified several limitations and challenges associated with the use of visualizations in cybersecurity, such as the lack of standardization and evaluation of visualizations and the disparity of solutions towards different types of end users. Following this, a possible implementation of a development framework was provided, obtained through a refinement of a previously presented framework, pointing out the various steps that as a result of the review showed critical issues and need for improvement.

Overall, the methodology used in this research explores the possibilities of security visualization and enumerates its various gaps and limitations. The obtained results can be leveraged to enhance the design and development of applications aimed at improving the security of computer systems and networks.

6.1 Future Works

The conducted study has identified several areas of interest where further research and the development of solutions are needed to address the identified gaps. The need to promote information exchange in security among users at the same level, as well as among different types of end users, emerged as one of the main gaps identified during the conducted research, based on the analyzed studies and received feedback. In this regard, it would be beneficial to explore the various possibilities offered by the tools considered in this study and propose the development of a methodology or solution aimed at addressing this gap.

Given the large percentage of tools in the prototype state encountered, it would be useful to start with one of the proposed solutions and refine it through the framework produced in this work, so as to propose a solution that can be used in a real-world context and evaluate its actual functionality and efficiency through a broader evaluation than those conducted and cited. This would provide actual evidence of the contribution of security visualization techniques. Simultaneously with this approach, it would be possible to develop a new type of solution, using the framework produced and proposed in this work, in order to verify its validity.

Another interesting field of application is definitely is the investigation on how visualizations can be utilized in the context of cybersecurity awareness training. Conducting a study to assess the effectiveness of visualizations in conveying complex cybersecurity concepts in an accessible and engaging manner to learners, thereby enhancing their understanding and awareness of digital threats, is an aspect undervalued by current research, but potentially significant.

Bibliography

[1] K. Eberhard, "The effects of visualization on judgment and decision-making: a systematic literature review", Management review quarterly, 8 2021, DOI 10.1007/s11301-021-00235-8

[2] A. Treisman, "Preattentive processing in vision", Computer Vision, Graphics, and Image Processing, vol. 31, no. 2, 1985, pp. 156–177, DOI https://doi.org/10.1016/S0734-189X(85)80004-9

[3] C. Intelligence, "Data visualization techniques for cyber security analysts, Guest Blog by Cambridge Intelligence", https://medium.com/cfs2020/data-visualization-techniques-for-cyber-security-analysts-guest-blog-by-cambridge-intelligence-1b3d8ddbfc56

[4] R. Marty, "Applied security visualization", Addison-Wesley Professional, 2008

[5] B. Balakrishnan, "Security Data Visualization", tech. rep., SANS Institute Inc, December 2014. https://www.sans.org/white-papers/36387/

[6] R. Amar, J. Eagan, and J. Stasko, "Low-Level Components of Analytic Activity in Information Visualization", 10 2005, DOI 10.1109/INFVIS.2005.1532136

[7] "VizSec", https://vizsec.org/

[8] S. Mckenna, D. Staheli, and M. Meyer, "Unlocking user-centered design methods for building cyber security visualizations", 2015 IEEE Symposium on Visualization for Cyber Security (VizSec), 2015, pp. 1–8, DOI 10.1109/VIZSEC.2015.7312771

[9] A. Komlodi, P. Rheingans, U. Ayachit, J. Goodall, and A. Joshi, "A user-centered look at glyph-based security visualization", IEEE Workshop on Visualization for Computer Security, 2005. (VizSEC 05)., 2005, pp. 21–28, DOI 10.1109/VIZSEC.2005.1532062

[10] C. L. Paul and K. Whitley, "A taxonomy of cyber awareness questions for the user-centered design of cyber situation awareness", Human Aspects of Information Security, Privacy, and Trust: First International Conference, HAS 2013, Held as Part of HCI International 2013, Las Vegas, NV, USA, July 21-26, 2013. Proceedings 1, 2013, pp. 145–154, DOI 10.1007/978-3-642-39345-7_16

[11] A. Komadina, Å. Mihajlović, and S. Groš, "Analysis of the design space for cybersecurity visualizations in vizsec", 2022 IEEE Symposium on Visualization for Cyber Security (VizSec), 2022, pp. 1–11, DOI 10.1109/VizSec56996.2022.9941422

[12] L. Jiang, A. Jayatilaka, M. Nasim, M. Grobler, M. Zahedi, and M. A. Babar, "Systematic literature review on cyber situational awareness visualizations", IEEE Access, vol. 10, 2022, pp. 57525–57554, DOI 10.1109/ACCESS.2022.3178195

[13] R. Damasevicius, J. Toldinas, A. Venckauskas, S. Grigaliunas, and N. Morkevicius, "Technical threat intelligence analytics: What and how to visualize for analytic process", 2020 24th International Conference Electronics, 2020, pp. 1–4, DOI 10.1109/IEEECONF49502.2020.9141613

[14] S. Venkatraman and M. Alazab, "Use of Data Visualisation for Zero-Day Malware Detection", Security and Communication Networks, vol. 2018, 12 2018, pp. 1–13, DOI 10.1155/2018/1728303

[15] L. Hao, C. G. Healey, and S. E. Hutchinson, "Ensemble visualization for cyber situation awareness of network security data", 2015 IEEE Symposium on Visualization for Cyber Security (VizSec), 2015, pp. 1–8, DOI 10.1109/VIZSEC.2015.7312766

[16] "H. S. Qiu, Streaming data visualization for network security, PhD thesis, Ph.D. thesis, Princeton University, 2017", https://www.cs.princeton.edu/~jrex/thesis/sophie-qiu-thesis.pdf

[17] S. McKenna, D. Mazur, J. Agutter, and M. Meyer, "Design activity framework for visualization design", IEEE Transactions on Visualization and Computer Graphics, vol. 20, no. 12, 2014, pp. 2191–2200, DOI 10.1109/TVCG.2014.2346331
[18] F. Carroll, "Usable security and aesthetics: Designing for engaging online security warnings and cautions to optimise user security whilst affording ease of use", Proceedings of the 2021 European Symposium on Usable Security, New York, NY, USA, 2021, pp. 23–28, DOI 10.1145/3481357.3481376
[19] F. Carroll and R. Lewis, "Internet security aesthetics: Can internet transparency afford social trust?", 2022 International Conference on Computer Communications and Networks (ICCCN), 2022, pp. 1–2, DOI 10.1109/ICCCN54977.2022.9868880
[20] "A Periodic Table of Visualization Methods, KPI Library, 2014", https://www.visual-literacy.org/periodic_table/periodic_table.html
[21] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations", 9 1996, DOI 10.1109/vl.1996.545307
[22] J. C. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull, "Graphviz and Dynagraph - Static and Dynamic Graph Drawing Tools", Springer Berlin Heidelberg, 1 2004
[23] S. Barnum, "Standardizing Cyber Threat Intelligence Information with the Structured Threat Information eXpression (STIX)", Whitepaper, 1 2014
[24] "Graphviz", https://graphviz.org/
[25] "RUMINT", http://www.rumint.com/
[26] "L.F. Haaijer, DDoS Packet Capture Collection, (2022)", https://github.com/StopDDoS/packet-captures
[27] "Tulip site", https://tulip.labri.fr/site/
[28] "Tulip SourceForge download page", https://sourceforge.net/projects/auber/
[29] "Tulip Github page", https://github.com/Tulip-Dev/tulip
[30] "Gephi site", https://gephi.org/
[31] T. M. J. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement", Software - Practice and Experience, vol. 21, 11 1991, pp. 1129–1164, DOI 10.1002/spe.4380211102
[32] "CodePulse", http://code-pulse.com/
[33] "Contoso University", https://github.com/alimon808/contoso-university
[34] H. Radwan and K. Prole, "Code pulse: Real-time code coverage for penetration testing activities", 2015 IEEE International Symposium on Technologies for Homeland Security (HST), 2015, pp. 1–6, DOI 10.1109/THS.2015.7225269
[35] D. Keim, "Designing pixel-oriented visualization techniques: theory and applications", IEEE Transactions on Visualization and Computer Graphics, vol. 6, no. 1, 2000, pp. 59–78, DOI 10.1109/2945.841121
[36] J. Landstorfer, I. Herrmann, J.-E. Stange, M. Dörk, and R. Wettach, "Weaving a carpet from log entries: A network security visualization built with co-creation", 2014 IEEE Conference on Visual Analytics Science and Technology (VAST), 2014, pp. 73–82, DOI 10.1109/VAST.2014.7042483
[37] "Pixel Carpet Demo", https://fhp.ivoherrmann.com/pixel-carpet/
[38] "Pixel Carpet Github page", https://github.com/solemone/pixel-carpet
[39] F. Böhm, L. Englbrecht, S. Friedl, and G. Pernul, "Visual decision-support for live digital forensics", 2021 IEEE Symposium on Visualization for Cyber Security (VizSec), 2021, pp. 58–67, DOI 10.1109/VizSec53666.2021.00012
[40] "Visual Decision-Support for Live Digital Forensics Github page", https://github.com/bof64665/LDF_ReactFrontend
[41] "A. J. Ferrante, Project CATO, Technical report ", https://assets.documentcloud.org/documents/6668313/FTI-Report-into-Jeff-Bezos-Phone-Hack.pdf
[42] A. Ulmer, D. Sessler, and J. Kohlhammer, "Netcapvis: Web-based progressive visual analytics for network packet captures", 2019 IEEE Symposium on Visualization for Cyber Security (VizSec), 2019, pp. 1–10, DOI 10.1109/VizSec48167.2019.9161633
[43] "WireShark: Network Analyzer", www.wireshark.org
[44] "NetCapVis demo", https://netcapvis.igd.fraunhofer.de/
[45] M. Angelini, G. Blasilli, P. Borrello, E. Coppa, D. C. D'Elia, S. Ferracci, S. Lenti, and G. Santucci, "Ropmate: Visually assisting the creation of rop-based exploits", 2018

IEEE Symposium on Visualization for Cyber Security (VizSec), 2018, pp. 1–8, DOI 10.1109/VIZSEC.2018.8709204
[46] "ROPMate Github", https://github.com/pietroborrello/RopMate
[47] "ROPDaemon Github", https://github.com/pietroborrello/RopDaemon
[48] "CISA site", https://www.cisa.gov/news-events/alerts/2022/10/14/cisa-releases-redeye-red-team-campaign-visualization-and-reporting
[49] "RedEye github", https://github.com/cisagov/RedEye
[50] M. Angelini, G. Blasilli, T. Catarci, S. Lenti, and G. Santucci, "Vulnus: Visual vulnerability analysis for network security", IEEE Transactions on Visualization and Computer Graphics, vol. 25, no. 1, 2019, pp. 183–192, DOI 10.1109/TVCG.2018.2865028
[51] "The MITRE Corporation. Common Vulnerabilities and Exposures (CVE)", https://cve.mitre.org/.
[52] "Forum of Incident Response and Security Teams (FIRST). Common Vulnerability Scoring System (CVSS-SIG)", https://www.first.org/cvss/
[53] "Vulnus prototype site", http://151.100.59.83:11768/vulnus/prototype/
[54] M. Angelini, S. Lenti, and G. Santucci, "Crumbs: A cyber security framework browser", 2017 IEEE Symposium on Visualization for Cyber Security (VizSec), 2017, pp. 1–8, DOI 10.1109/VIZSEC.2017.8062194
[55] "Crumbs site", http://awareserver.dis.uniroma1.it:11768/crumbs/
[56] A. Nadeem, S. Verwer, and S. J. Yang, "Sage: Intrusion alert-driven attack graph extractor", 2021 IEEE Symposium on Visualization for Cyber Security (VizSec), 2021, pp. 36–41, DOI 10.1109/VizSec53666.2021.00009
[57] "Suricata site", https://suricata.io/
[58] "SAGE repository", https://github.com/tudelft-cda-lab/SAGE/
[59] "Suricata alerts dataset", https://github.com/FrankHassanabad/suricata-sample-data
[60] F. L. Dennig, E. Cakmak, H. Plate, and D. A. Keim, "Vulnex: Exploring open-source software vulnerabilities in large development organizations to understand risk exposure", 2021 IEEE Symposium on Visualization for Cyber Security (VizSec), 2021, pp. 79–83, DOI 10.1109/VizSec53666.2021.00014
[61] "Eclipse Steady", https://github.com/eclipse/steady
[62] "Vulnex repository", https://github.com/dbvis-ukon/vulnex
[63] "Eclipse Foundation", https://github.com/eclipse
[64] "NIST site", https://nvd.nist.gov/
[65] C. P. Lee and J. A. Copeland, "Flowtag: A collaborative attack-analysis, reporting, and sharing tool for security researchers", Proceedings of the 3rd International Workshop on Visualization for Computer Security, New York, NY, USA, 2006, pp. 103–108, DOI 10.1145/1179576.1179597
[66] "Flowtag Github page", https://github.com/chrislee35/flowtag
[67] "InetVis site", https://www.cs.ru.ac.za/research/g02v2468/inetvis.html
[68] "InetVis Github page", https://github.com/yestinj/inetvis
[69] M. J. Alam, M. T. Goodrich, and T. Johnson, "J-viz: Finding algorithmic complexity attacks via graph visualization of java bytecode", 2016 IEEE Symposium on Visualization for Cyber Security (VizSec), 2016, pp. 1–8, DOI 10.1109/VIZSEC.2016.7739575
[70] "J-Viz SourceForge site", https://sourceforge.net/projects/jviz/
[71] M. Dumas, J.-M. Robert, and M. J. Mcguffin, "Alertwheel: radial bipartite graph visualization applied to intrusion detection system alerts", IEEE Network, vol. 26, no. 6, 2012, pp. 12–18, DOI 10.1109/MNET.2012.6375888
[72] "AlertWheel SourceForge site", https://sourceforge.net/projects/alertwheel/
[73] M. Angelini, G. Blasilli, S. Bonomi, S. Lenti, A. Palleschi, G. Santucci, and E. D. Paoli, "Bucephalus: a business centric cybersecurity platform for proactive analysis using visual analytics", 2021 IEEE Symposium on Visualization for Cyber Security (VizSec), 2021, pp. 15–25, DOI 10.1109/VizSec53666.2021.00007
[74] M. Beran, F. Hrdina, D. Kouřil, R. Ošlejšek, and K. Zákopčanová, "Exploratory analysis of file system metadata for rapid investigation of security incidents", 2020 IEEE Symposium on Visualization for Cyber Security (VizSec), 2020, pp. 11–20, DOI 10.1109/VizSec51108.2020.00008

[75] R. Gove, "V3spa: A visual analysis, exploration, and diffing tool for selinux and seandroid security policies", 2016 IEEE Symposium on Visualization for Cyber Security (VizSec), 2016, pp. 1–8, DOI 10.1109/VIZSEC.2016.7739580
[76] "V3SPA Github page", https://github.com/twosixlabs/V3SPA
[77] J. V. Komárková, M. Husák, M. Lastovicka, and D. Tovarnák, "CRUSOE: Data Model for Cyber Situational Awareness", 8 2018, DOI 10.1145/3230833.3232798
[78] "CRUSOE University page", https://is.muni.cz/publication/1724716/
[79] V. Pham and T. Dang, "Cvexplorer: Multidimensional visualization for common vulnerabilities and exposures", 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 1296–1301, DOI 10.1109/BigData.2018.8622092
[80] "CVExplorer Demo site", https://idatavisualizationlab.github.io/CVSS/
[81] J. L. Guerra, E. Veas, and C. A. Catania, "A study on labeling network hostile behavior with intelligent interactive tools", 2019 IEEE Symposium on Visualization for Cyber Security (VizSec), 2019, pp. 1–10, DOI 10.1109/VizSec48167.2019.9161489
[82] "riskIDemo Github page", https://github.com/jorgeguerra881215/riskIDemo
[83] N. Rakotondravony and L. Harrison, "Visualization for cyber security and security analysts' use of visualizations: Is there a gap?", 2020
[84] "SolarWinds Security Event Manager site", https://www.solarwinds.com/security-event-manager
[85] "IBM Security QRadar SIEM site", https://www.ibm.com/products/qradar-siem
[86] "Cycognito site", https://www.cycognito.com/
[87] "Ntop site", https://www.ntop.org/
[88] P. A. Legg, "Enhancing cyber situation awareness for non-expert users using visual analytics", 2016 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (CyberSA), 2016, pp. 1–8, DOI 10.1109/CyberSA.2016.7503278
[89] M. Sturdee, L. Thornton, B. Wimalasiri, and S. Patil, "A Visual Exploration of Cybersecurity Concepts", 6 2021, DOI 10.1145/3450741.3465252
[90] V. Rusnak. and M. Drasar., "Towards a visual analytics workflow for cybersecurity simulations", Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - IVAPP, 2023, pp. 179–186, DOI 10.5220/0011695000003417
[91] "CodePulse User Guide", https://github.com/codedx/codepulse/wiki/user-guide
[92] "Npmjs site", https://www.npmjs.com/

Appendix A

User's Manual

The following section has the purpose of explaining how to install and utilize any of the tool proposed in this research.

GraphViz

GraphViz can be downloaded from its official website [24] in both executable and compileable versions, for all various widely-used operating systems. Once the tool is installed, it can be used via the command line interface. The website provides comprehensive documentation for both the GraphViz application itself and its specific extension, .DOT, which is utilized by the tool.

Rumint

Rumint is available for download on its official website [25], offering both a compiled version and an executable version. It should be noted that Rumint is only available for Windows operating systems. Once installed and launched, you can load your pcap files in Rumint using the dedicated dropdown menu, keeping in mind the limitation of 30,000 packets. Once the file is loaded, you can manage the flow of packets using its VCR-like interface. To open the various visualization modes, including simultaneous views, you can select the desired view from the menu and click on it. All views follow the controlled flow of the application's main window. To replicate the tests conducted in this work, you can download the pcap files used from [26].

Tulip

Tulip is available through its SourceForge page [28] or you can access its source code on GitHub [29]. If you choose to download the executable, please note that it is only provided for Windows. To replicate the examples provided, you can import your CSV data for visualization using the dedicated import procedure. This can be done by opening the Tulip application and using the appropriate icon for importing data. After importing the data, you can create a new panel and select the geographic view type. If Tulip does not automatically recognize the fields representing geolocation data, you can manually modify them through the view settings. Once you have obtained the desired view, you can modify various layout aspects according to your preferences. In this case, the modifications include changing the shape and color gradient. All these modifications can be easily made by following the documentation available on the Tulip website [27].

CodePulse

Codepulse is available for Windows and can be downloaded from the official website of the application [32]. Once installed, you can launch CodePulse and initiate the first basic analysis by

simply dragging and dropping the zip file of your project into the application. For the second part of the analysis, it is necessary to connect CodePulse to your application. A comprehensive guide is provided [91], detailing the various scenarios based on different types of applications to link to the tracker, both for Java and .NET projects. Once the two parts are connected, you can proceed with the various tests according to your needs.

Pixel Carpet

Pixel Carpet is the first prototype encountered in this section. There are two possibilities for its usage: firstly, you can explore its functionalities through an online prototype available at the following link [37], or secondly, there is the option to download the source code [38] and run it locally on your machine for a more hands-on experience. By choosing the second option, you can insert your log files into the code by modifying the relevant line of code within the main file of the application. Once you launch the JavaScript application, for example using Node as performed in the previous test conducted in this work, you can access via browser the application with the url http://127.0.0.1:5500/.

Code Listing A.1: Command used for launch the application

```
node main.js
```

Visual Decision-Support for Live Digital Forensics

In the case of Visual Decision-Support for Live Digital Forensics, the only option to use the application is to run it locally using the code provided by the developers [40]. Along with the source code, the repository includes a brief but comprehensive guide on how to run the application. One additional requirement is to have a recent version of npm [92] installed on your machine.

NetCapVis

The NetCapVis prototype is only available through the online demo [44]. Please note that the demo allows you to either use the tool with a predefined data file or upload your own pcap file with a size limitation of 1 MB. In addition to the functionalities already mentioned in its description, a tutorial can be carried out that delves into the various elements that make up the application.

RopMate

Also in the case of RopMate [46], the developers provide along with the source code a set of instructions, both for setting up the environment and for starting the tool, which can always be visited via browser on localhost:5000.

RedEye

Regarding RedEye, similar to the previous tools, instructions for execution can be found on the tool's GitHub page [49]. Users can download the appropriate version based on their needs, including the executable and source code, tested for the main operating systems. The source code can be built manually or using Docker. It is important to note that both Red and Blue Team modes can be started from the same RedEye application binary.

Vulnus

As mentioned in its description, Vulnus is only available through its online demo version [53]. The only recommendations provided are to use Firefox as the preferred browser, as recommended by the developers, and to use monitors and resolutions that allow for a complete display of the application.

Crumbs

Through the Crumbs presentation page [55], you can access both the online demo and download the source code. The demo does not provide customization options, while within the downloaded folder, you can make modifications regarding the Excel file that is loaded when the application is launched locally. The folder contains a file with instructions and guidelines to follow. The only prerequisite is to have Python installed on your machine (the instruction file specifies the various commands based on the version used).

SAGE

For SAGE as well, there are two main usage modes, both of which are described step by step in the repository [58]. Regarding the tests conducted for this work, the Docker version was preferred over the executable version from the terminal for simplicity of adoption. If you opt for a different choice, it is important to install the various requirements listed in the instructions. Both versions have been tested and are functional. If you wish to replicate the tests conducted in this work, you can obtain the alerts used through [59].

Vulnex

Finally, as far as Vulnex is concerned, via its Github page [62] it's possible to download its source files to be built and ran via Docker, as indicated by the guide. As mentioned during its description, the tool basically shows an example, defined as a demo, which is given in Chapter 4.1, but it is also possible to link and analyse one's own application via the Eclipse Steady Backend [61], through the indications provided in the guide available in the repository. Unfortunately, this was not possible during the course of our work, so we cannot guarantee that the procedure indicated will work.