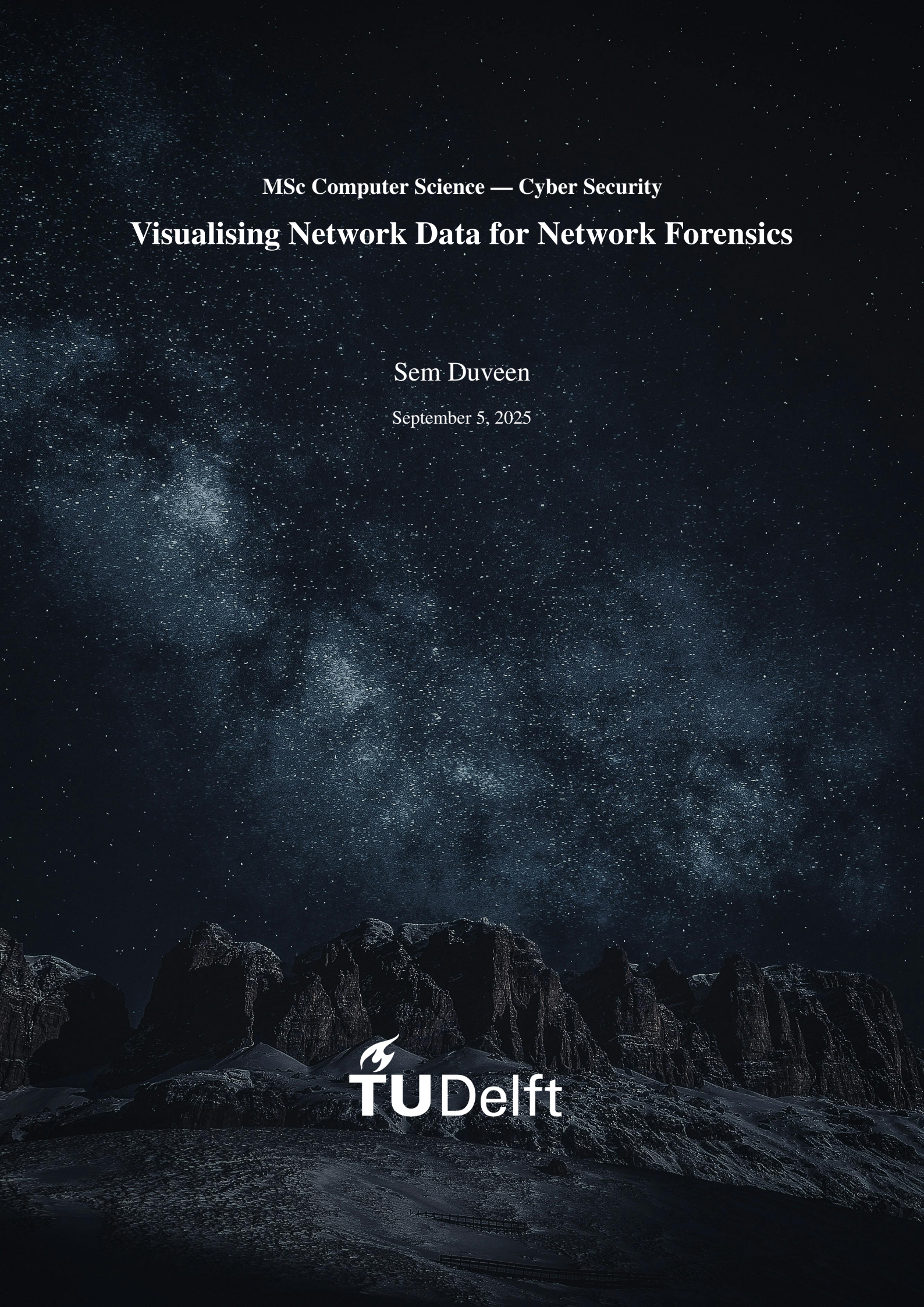

MSc Computer Science — Cyber Security

Visualising Network Data for Network Forensics

Sem Duveen

September 5, 2025

TU Delft

Visualising Network Data for Network Forensics

Name:	Sem Duveen
Student Number:	4906810
Study Programme:	MSc Computer Science
Research Group:	Cyber Security
Thesis Advisor:	Harm Griffioen
Daily Supervisor:	Martin Skrodzki
Committee Member:	Georgios Smaragdakis
Submission Date:	05/09/2025
Defense Date:	12/09/2025

Abstract

Cybercrime is becoming increasingly sophisticated, creating major challenges for cybersecurity and, in particular, network forensics. Network forensics analysts play an essential role in detecting intrusions, investigating incidents, and preventing attacks through the analysis of network traffic. Yet the volume and complexity of network data, combined with the limits of current tools, make this work time-consuming and error-prone. This study addresses these challenges by designing and developing a prototype visualisation tool tailored to the needs of network forensics analysts. The research follows a structured four-phase methodology. First, interviews with professional analysts identify key tasks and challenges, forming the basis for the tool's requirements. Second, the What-Why-How framework maps these tasks to appropriate visualisation techniques. Third, iterative prototyping translates these insights into an interactive prototype tool, refined through user feedback. Finally, an impact analysis assesses effectiveness, intuitiveness, and correctness by evaluating tasks performed by both experienced and inexperienced participants. The evaluation shows that the prototype significantly improves analysis workflows, lowering cognitive load, increasing efficiency, and reducing errors. Both experienced and inexperienced participants performed tasks faster using the prototype, demonstrating both accessibility and a reduced learning curve. The findings also highlight the value of a user-centred, task-oriented approach to visualisation design. This research contributes to network forensics by presenting a structured approach to developing domain-specific visualisation tools. The results highlight their potential to enhance the speed, accuracy, and reliability of analysis, strengthening cybersecurity as a whole.

Preface

After a year of hard work and stress, I can finally say this study is finished. There were a lot of times when I doubted it would ever be. The journey was anything but smooth, with setbacks, delays, health problems, and a healthy dose of stress. But even through it all, I am proud and happy to have reached the finish line. However, I definitely would not have been able to do this on my own, so some thanks are in order.

First, I want to thank Harm and Martin for their guidance and supervision. During times when progress felt impossible, our bi-weekly meetings gave me confidence in what I had already accomplished, as well as the pressure I needed to keep going. I always felt I could turn to them for help or support, and for that I am sincerely grateful. In a similar sense, I also want to thank Stefan. While not officially a supervisor, he was always willing to step in to fact-check information. On top of that, finding interview participants, receiving feedback on designs, and performing the impact analysis would have been nearly impossible without his help.

While I will refrain from revealing any names, I also want to thank everyone who has contributed to this study by either participating in the interviews or the impact analysis. Without them, this study would have, of course, not been possible. You all have my deepest gratitude for making this study possible at all.

Finally, I owe my greatest thanks to my family, friends, and girlfriend. They have all supported me through what might have been one of the toughest periods I have experienced. Thank you for letting me live like a hermit and still being there for me whenever I reached out. The moments I could talk or vent to you about my problems were what kept me going. I will make up for this by listening to *your* problems next time.

And with that, the sappy part is over. I have never been good at writing this kind of thing, but I am glad I at least managed to do it this time. One last thank you to everyone involved, and I hope you enjoy this study. It was about time I finished it.

Sem Duveen
September 5, 2025

Contents

1 Introduction

In today's world, cybercrime is often associated with modern technologies and information systems. However, its origins date back much further in history. The earliest recorded instance of cybercrime dates back to 1834 [1], when, through unauthorised access, financial market information was stolen from the French telegraph system. Remarkably, this occurred long before the invention of the internet in 1969 [2] or even the first general-purpose computer in 1837 [3]. Since then, cybercrime has continuously evolved, becoming increasingly sophisticated and widespread [4]. In our current digital and interconnected society, the reliance on online systems has caused the attack surface to grow ever larger, providing attackers with more opportunities to exploit weaknesses in systems and networks.

The increasing frequency and severity of cyberattacks have made cybersecurity a field of high importance. Within cybersecurity, *network forensics* is a significant field that handles the detection, prevention, and investigation of cybercrime. It involves acquiring, monitoring, and analysing network traffic data to detect intrusions, gather information, and aid with legal investigations [5]. Professionals in this field are known as network forensics analysts. Their work is essential for understanding the nature and scope of cybercrime, allowing organisations to respond effectively and prevent future attacks.

In a time when digital evidence can determine the outcome of an investigation, network forensics has become a critical component of modern cybersecurity. Network forensics analysts contribute not only to cybercrime detection and response, but also to the improvement of long-term security. Their expertise is increasingly seen as indispensable, not just in responding to cyber threats, but in anticipating and mitigating them before they occur. As a result, the demand for skilled network forensics analysts has only continued to rise [6].

Despite a growth in the number of network forensics analysts, a significant shortage of skilled professionals remains [7]. This shortage is partially caused by the various challenges network forensics analysts face during analysis, two of which are particularly significant. The first of these challenges is the volume of raw data that must be analysed [8]. Network traffic produces massive amounts of data, much of which is irrelevant to a given investigation. However, to find patterns, anomalies, or other signs of malicious activity, analysts must go through all the data. This is a difficult and time-consuming task that requires a deep understanding of both normal and malicious network traffic. Furthermore, the raw data is very complex in nature [8]. Without prior knowledge of what to look for, identifying relevant indicators is extremely difficult and inefficient.

The second challenge lies in the limitations of the tools used by network forensics analysts. Current network forensics tools do not sufficiently support the management of the overwhelming volume [9] and complexity of network data [10]. Many existing tools require analysts to manually explore large datasets, apply filters, and interpret results with little automation or guidance. This manual approach not only slows down investigations but also increases the risk of missing critical information within the irrelevant traffic. Furthermore, analysts are often required to rely heavily on their expertise and intuition to identify subtle patterns or anomalies, which makes the process more time-consuming and prone to error. The limitations of current tools combined with the scale and complexity of network data contribute significantly to the ongoing shortage of skilled professionals in the field.

To meet the growing demand for skilled network forensics analysts, it is essential to reduce the complexity of the analytical process and make it more accessible. One promising approach is the use of visualisations to represent network data, enabling analysts to identify patterns more efficiently than when working with raw data alone. Research in cognitive science indicates that humans are by nature better at finding and interpreting visual patterns compared to textual or numerical ones [11]. By utilising network data visualisations, the process of detecting anomalies and suspicious activity can become faster and more intuitive. Additionally, visualisations can improve the situational awareness of analysts, which in turn allows for more effective and swifter decision-making [12]. As such, visualisations of network data can help overcome some of the key challenges associated with network forensics analysis.

To improve the currently complex and challenging workflow of network forensics analysis, this study aims to design and develop a functional prototype of an interactive visualisation tool fine-tuned to the

needs of network forensics analysts. The tool is intended to support analysts in exploring, interpreting, and managing large volumes of complex network traffic data more efficiently.
The development of this tool is guided by the following research question:

How can network data visualisations be designed to support forensic analysts in overcoming challenges and improving the efficiency of their workflows?

To answer this question, the study outlines a structured process for creating a visualisation tool specifically adapted to the needs of network forensics analysts. The research process is divided into four phases, each addressing one of the following sub-questions:

SQ1: What are the key tasks and challenges that network forensics analysts face during the analysis process?

SQ2: Which types of visual representations best support specific tasks or challenges in network forensics analysis?

SQ3: How can these task-visualisation pairs be integrated into the design of a prototype tool?

SQ4: What impact does the use of the prototype have on the analysts' efficiency and effectiveness during analysis?

The answers to these questions guide the development of a working prototype that serves as a proof of concept for a tool designed to support more efficient and potentially more accurate network forensics analyses. This prototype may then also serve as a foundation for future visualisation tools that are fine-tuned to a specific group of people.
The study is structured into several sections, each building upon the previous. The *Background* section introduces information necessary to understand and correctly interpret the remainder of the thesis. This is followed by the *Related Works* section, which reviews existing literature and projects related to this study. It also highlights the research gap, demonstrating the relevance and necessity of this research. Next, the *Methodology* and *Results* sections outline the research questions, their importance, the steps taken to answer them, and the outcomes of each of those steps. The study is conducted in four main stages: (1) conducting interviews with network forensics analysts to identify key tasks and challenges, (2) linking those tasks to suitable visualisation techniques, (3) developing a functional prototype tool based on those insights, and (4) evaluating the prototype to assess its potential effectiveness and usability. The *Discussion* section then interprets the key findings, explores their implications, and considers the study's limitations, validity, and possible future improvements. Finally, the *Conclusion* section summarises the main insights and contributions of the research and provides a final answer to the research question.

2 Background

This section introduces the foundational knowledge required to understand the context of this study and the topics discussed within. It provides an overview of three important domains relevant to this study: computer networks, network security, and UI/UX design.

2.1 Computer Networks

Various types of communication networks exist, such as telephone networks for making calls and cable networks for video transmissions [13]. Among these networks, computer networks are one of the most important components of the Internet. A computer network is a system that links two or more computing devices, such as mobile phones, computers, and servers, with the purpose of sharing information [14]. In contrast to more specialised networks, computer networks are distinguished by their versatility. They can transmit various types of data and support a wide range of applications [13].

2.1.1 History of the Internet

Contrary to common belief, the Internet was not the first computer network. The development of the Internet began in 1969 with the creation of the Advanced Research Projects Agency Network (ARPANET), the first functional computer network. The concept of ARPANET was created by the Information Processing Techniques Office and was further developed and completed by Joseph Licklider and Lawrence Roberts [15]. The initial version of the ARPANET consisted of only four connected devices capable of host-to-host communication. Over time, this network grew, ultimately becoming the base for the modern Internet [16]. ARPANET remained operational until its official retirement in 1990 [15].

2.1.2 OSI Model

The Open Systems Interconnection (OSI) model is a conceptual framework that is used to understand how networks communicate with each other. This model describes networks as 7 layers that communicate with the layers above and below. [17].

- **Layer 1 – Physical:** Responsible for the transformation of signals into raw bitstreams and their physical transmission over hardware such as wires. [18].
- **Layer 2 - Data Link:** Regulates how raw bits from the physical layer are broken down into smaller, structured frames [18].
- **Layer 3 - Network:** Handles the routing of data between networks [19]. It assigns addresses to packets with which they can find the correct path [18].
- **Layer 4 - Transport:** Ensures correct, reliable, and safe data transmission [18].
- **Layer 5 - Session:** Manages the creation, maintenance, and termination of sessions between different devices [17].
- **Layer 6 - Presentation:** Translates human-readable code into machine code that can be used by the lower layers [18].
- **Layer 7 - Application:** Serves as the visual and interactive interface between users and networks [18].

2.1.3 Network Data

While it may not be apparent to users, network data is not transmitted as only a simple message. Instead, it must follow a specific format and include extra information to ensure its correct delivery to the intended destination. The following provides an overview of the attributes and structure of network data.

IP Address Before being able to understand the structure of network data, it is essential to know how data is directed from one specific machine to another. To achieve this, each device is assigned a unique numerical identifier known as an Internet Protocol (IP) address [20]. IP addresses come in two formats: IPv4 and its newer variant, IPv6. An IPv4 address consists of four decimal numbers ranging from 0 to 255, separated by periods. For example, 192.168.1.181 would be a valid IPv4 address. IPv6 addresses are slightly more complex, consisting of eight segments containing hexadecimal values ranging from 0000 to FFFF, separated by colons. An example of a valid IPv6 address is 27e1:7291:c69e:b659:8298:6234:3632:1929. These IP addresses are included in network data as source and destination identifiers, allowing for accurate data delivery from one machine to another [21]. For this reason, IP addressing operates at layer 3 of the OSI model [22].

MAC Address While IP addresses provide global identification [21], devices must also be uniquely identified locally within a local area network (LAN). This local identification is achieved through the use of a Media Access Control (MAC) address, a 12-digit hexadecimal identifier [23]. MAC addresses are typically structured as six pairs of hexadecimal digits separated by colons. An example of a valid MAC address is 48:bd:02:a9:53:5a. MAC and IP addresses work together to allow for communication on both a global and local scale. Unlike IP addresses, which work at layer 3, MAC addresses function at layer 2 of the OSI model [22].

Packets Network data is transmitted as small segments called packets that, when combined, form a larger message. These packets are sent to the other device, potentially arriving out of order. The receiving device then reassembles the packets to reconstruct the original message [24]. The process of formatting the raw data into packets happens in the Data Link Layer of the OSI model [19]. Although this method may initially seem unnecessarily complex, transmitting an entire message in one piece is not practical in a real-world setting. Because messages are transmitted as bits over physical wires, if multiple machines are required to use the same wire, it can't be occupied for too long at a time. Sending the entire message at once could block a wire for an extended time, blocking access for other devices [24]. By sending packets instead of the entire message, multiple devices can transmit data over the same wire simultaneously.

Sessions To communicate over a computer network, devices must first establish a session. A session enables a temporary two-way exchange of data between two or more entities [25]. Once the data exchange has finished, the session is ended. During a session, multiple packets are transmitted in both directions, each fulfilling a specific purpose. Some packets carry data, while others are responsible for initiating or terminating the session [26]. These functions are handled by layer 5 of the OSI model [19].

HTTP and HTTPS A large portion of internet communication happens through the Hypertext Transfer Protocol (HTTP). When a device requires a resource, it sends an HTTP request to a specific URL known as an endpoint, which is defined and made available on the server. The server responds with an HTTP message that includes the requested data and a status code indicating the outcome of the request [27], [28]. HTTP defines several request methods, each serving a specific purpose [29].

- **GET:** Retrieves data from a specified resource without modifying the state of the recipient's resources.
- **POST:** Sends data to the server, typically resulting in a change in state or creation of a resource. It does not necessarily expect data as a response.
- **HEAD:** This method returns information about a resource on the recipient's side.
- **PUT:** Replaces a specified resource or creates it if it does not exist.
- **DELETE:** As the name implies, this method deletes a resource from the recipient's server.
- **TRACE:** TRACE is used for diagnostics, troubleshooting, and debugging. It responds with a log of requests and responses.

- **OPTIONS:** Returns the HTTP methods supported by the server for a specific endpoint.
- **CONNECT:** Creates a connection with a resource on the server.
- **PATCH:** Like PUT, it is used to update resources. PATCH, however, is much more efficient as it does not have to contain the complete updated resource.

The HTTP responses sent by the recipient after receiving an HTTP request, besides possible data, contain a status code. These can be divided into five groups. Status codes 100–199 are informational responses. These codes indicate that the request was received and understood. Codes 200–299 are successful responses. Meaning that, in addition to receiving and understanding, the requested action has also been approved. Codes 300–399 are reserved for redirection messages. When receiving this code, the client needs to perform additional actions to complete their request. Codes 400–499 are client error responses. These will be sent as a response when an error is triggered by the client. Finally, codes 500–599 are used whenever the server was unable to fulfil the request without the client triggering an error [30].
A secure version of HTTP, known as Hypertext Transfer Protocol Secure (HTTPS), is used to protect data during transmission. Unlike HTTP, HTTPS encrypts transmitted data through the use of TLS, preventing unauthorised parties from reading intercepted traffic [31]. [31]. Besides being more secure, HTTPS also supports all the same methods and status codes as HTTP [27].

HTTP Headers HTTP headers are key-value pairs containing additional information sent along HTTP requests and responses [32]. They can control caching behaviour, manage the state of a session, define content types, and give authentication information. Headers are not directly visible to users, but are critical for browsers and servers to interpret how a request or response should be handled [33].

Common request headers include [33]:

- **Accept:** Specifies the media types that the client is capable of accepting from the server. For example, a client may send `Accept: application/json, text/html` to indicate a preference for JSON or HTML responses.
- **User-Agent:** The *User-Agent* header identifies the software or browser making the request. By examining this header, the server can adapt its response to ensure compatibility.
- **Authorization:** This header carries credentials used to authenticate the client when attempting to access protected resources.
- **Content-Type:** The *Content-Type* header indicates the media type of the data contained in the request body, such as `application/json`. This allows the server to correctly interpret and process the payload.
- **Cookie:** Through the *Cookie* header, the client sends previously stored cookies back to the server.

Common response headers include [33]:

- **Content-Type:** This header specifies the media type of the data returned by the server. It may also include optional parameters that further describe the content.
- **Cache-Control:** The *Cache-Control* header controls caching behaviour on the client side. For instance, `Cache-Control: max-age=3600, public` directs the client to cache the response for one hour and allows caching by public caches.
- **Server:** This header provides information about the server that generated the response, including its version and platform, such as `Apache/2.4.10 (Unix)`.
- **Set-Cookie:** The *Set-Cookie* header instructs the client to store a cookie with specified attributes like expiration, domain, path, and security flags.
- **Content-Length:** This header indicates the size of the response body in bytes, allowing the client to anticipate the amount of data to be received.

VPN A Virtual Private Network (VPN) is a service that provides users with a secure and private connection to the Internet. When using a VPN, all of the data transmitted between the user's device and the VPN server is encrypted. Additionally, a VPN hides the user's actual IP address and replaces it with one from the VPN provider. These two network security techniques cause communication using a VPN to be completely anonymous and secure [34]. VPNs are typically used for protection on public Wi-Fi networks, to access content restricted by region, or to avoid internet activity from being tracked [35].

TOR The Onion Router (TOR) is a free, open-source software that ensures anonymous communication over the Internet. Unlike VPNs, which use a single server to encrypt and route traffic, TOR uses a technique called onion routing [36]. Using this method, Internet traffic is encrypted and sent through multiple randomly selected servers, called nodes, before reaching its destination. Each node decrypts only enough information to know where to send the data next, ensuring that no node has complete knowledge of the source and destination [37]. The encryption provides strong anonymity, making TOR a preferred tool for users seeking to keep their Internet activity private. However, TOR can be significantly slower than traditional browsing due to the multiple layers of routing [38].

2.1.4 Important Network Protocols

Modern computer networks rely on a wide range of communication protocols, each designed to fulfil a specific function within layers of the OSI model. The following highlights and describes some of the more common protocols:

- **TCP:** A connection-oriented layer 4 protocol that ensures reliable, ordered, and error-proof delivery of data. It uses a three-way handshake to establish a safe connection [39].
- **UDP:** A connectionless layer 4 protocol that provides minimal services without guaranteeing delivery or correct order [40]. It is suitable for time-sensitive applications like streaming and DNS queries.
- **FTP:** A layer 7 protocol used to transfer files over TCP. FTP lacks encryption, making it unsuitable for sensitive transfers [41].
- **FTPS / SFTP:**
  - **FTPS:** FTP secured with encryption [42].
  - **SFTP:** A different secure file transfer protocol based on SSH [43].
- **SSH:** A cryptographic layer 7 protocol providing secure remote login and command execution. It provides authentication, encryption, and data integrity[43].
- **DNS:** A layer 7 protocol used for distributed naming that resolves human-readable domain names into IP addresses. It primarily uses UDP but may use TCP for zone transfers [44], [45].
- **ICMP:** A layer 3 protocol used for error reporting and diagnostic functions such as 'ping'. It is not used for regular data transfer [46].
- **SMTP / SMTPS:** SMTP is a layer 7 protocol used for sending emails between servers. SMTPS is the encrypted version of SMTP [47], [48].

2.2 Network Security

Network security refers to the process of protecting a network from various types of internal and external threats [49]. It involves implementing various threat mitigation and detection techniques into a computer network.

2.2.1 Cybercrime

Although the term cybercrime is widely used and generally well-understood, there is no universally accepted definition. Current attempts at defining cybercrime vary significantly depending on the context in which it occurs [50]. Broadly speaking, cybercrime refers to criminal activities with computers or computer networks being tools or targets [51]. This includes a wide range of offences, such as hacking and denial-of-service attacks, to traditional crimes where computers or networks are used to carry out illegal activities.
An important distinction in defining cybercrime lies in the actions being illegal. For example, unauthorised access to computer systems, commonly known as hacking, is considered a cybercrime. However, with the system owner's permission, it is not regarded as illegal. Such hackers are known as white hat hackers [52]. These hackers identify and report security vulnerabilities to help improve the system's security. As such, there exists a thin line between cybercrime and cybersecurity, and many more such grey areas exist within this field.

2.2.2 Network Forensics

Network forensics is a branch of cybersecurity focused on capturing, monitoring, and analysing computer network traffic to gather information, detect intrusions, and investigate cyber incidents [5]. Unlike traditional forensics, which deals with static data, network forensics can involve the real-time capture, recording, and analysis of network traffic data. The goal is to identify the source, nature, and impact of malicious activities such as unauthorised access, data exfiltration, or denial-of-service attacks.

2.2.3 Network Forensics Analysis Tools

Network forensics analysis tools are software applications designed to support investigations within the field of network forensics. These tools provide various functionalities essential to the analysis process. Some tools are specialised for specific types of analysis, while others offer a broader range of functionalities. Popular tools include Wireshark, Arkime, and various other commercial and open-source tools.

Arkime. Arkime, which used to be known as Moloch, is a web-based, open-source tool designed for network analysis, created in 2012 for the purpose of replacing America Online's (AOL) commercial full-packet systems. It uses a search database based on ElasticSearch or OpenSearch to handle network data analysis, search within the network data, and export the network data. It utilises the packet capture (PCAP) format to index and store network traffic. The web interface then allows for the user to easily search and browse within the imported PCAPs [53]. The web interface is not the only way to make use of Arkime's features. It also provides an Application Programming Interface (API), which allows users to use features by sending HTTP requests to a set of endpoints. Through the use of the API, all sessions can be acquired, generated from the PCAPs that have been imported into Arkime. There are various filters available when requesting sessions, making the use of the API as versatile as the web interface.

Wireshark. Wireshark is a popular open-source network protocol analyser that allows users to capture and analyse network traffic in real time or from a capture file [54]. It provides detailed information about many protocols and analysis of network packets down to the byte level with broad filtering functionality. Wireshark also supports exporting network traffic data in multiple formats for further external analysis [55]. This wide range of functionality is contained within an interface that makes it usable for users of various skill levels.

2.3 UI / UX Design

User Interface (UI) and User Experience (UX) design play an important role in shaping how users interact with digital products. UI design concerns the visual and interactive elements of a digital product with the goal of providing a user-friendly and visually appealing interface that is intuitive to navigate [56]. Examples are the design choices for the layout, colour scheme, icons, and other similar concepts. In contrast, UX design concerns the complete interaction journey of users within a digital product, with the goal of meeting user expectations and creating an enjoyable and meaningful experience [56]. This includes the user experience, usability testing, information strategy, and more. Effective UI/UX design ensures that applications are not only visually appealing but also intuitive, efficient, and feel good to use. The design process is often iterative, involving user feedback, testing, and continuous refinement to meet the needs and expectations of the target audience.

Before completing a fully functional digital product, designers typically follow several stages of prototyping, each varying in complexity and detail. These stages are called low fidelity, medium fidelity, and high fidelity[57]. They serve to iteratively refine the design and functionality of the product. Starting with simple, abstract representations and gradually increasing in functionality, visual detail and interactivity, each stage plays a crucial role in shaping the final user experience.

2.3.1 Low-Fidelity Design

Low-fidelity designs are rough, abstract representations of a digital product's interface. They are used in the early stages of design to outline the basic layout and connections between screens without focusing on visuals or interactivity [57]. By keeping the designs minimal, teams can easily test different layouts and make changes without a significant time investment. These designs often take the form of hand-drawn sketches or digitally created greyscale representations called wireframes.

Wireframes. Wireframes are an abstract and minimal skeletal framework of a digital product. They focus on content placement and overall page structure rather than colours, fonts, or detailed graphics. Wireframes serve as blueprints that help designers and stakeholders agree on the functionality and layout before moving forward [57].

2.3.2 Medium-Fidelity Design

Medium-fidelity designs are more detailed and look more realistic than low-fidelity designs. These designs now define fonts, spacing, colour, and other basic details. Unlike low-fidelity designs, medium-fidelity designs may include some simple interactive components such as clickable buttons or links between pages that simulate navigation [57]. This allows for more accurate user testing without committing to final functionalities. These more detailed, low-interaction designs are called mockups.

Mockups. Mockups are mostly static, detailed designs that show what the final product will look like visually. Mockups serve as a reference for developers and stakeholders to understand the visual direction before moving to full prototyping. This allows for changes within functionality to still be easily changed [57].

2.3.3 High-Fidelity Design

High-fidelity prototypes closely resemble the final product in appearance and functionality. These interactive prototypes include simulations for user interactions, animations, and responses, providing an almost complete experience [57]. They are used for thorough usability testing and to gather detailed feedback on interactions and functionality before development. Since the functionality is not truly implemented in this stage, such prototypes allow for relatively quick functionality changes.

3 Related Work

This section presents a review of existing studies concerning network forensics and data visualisation.. Each subsection outlines the aims, conclusion, and significance of three selected studies, highlighting their contribution to their respective field. The section concludes by discussing the problems unaddressed by these studies and outlining how the current study seeks to address them.

3.1 Network Forensics

Qureshi et al. [58] identify challenges within network forensics such as high-speed data storage, data integrity, data privacy, locating intruder IPs in distributed systems, and more. They propose a thematic taxonomy, a system for classifying information based on themes or topics [59], that groups existing network forensics techniques based on the types of data they use and how they are applied. This classification is meant to make it easier to access network data, improve the accuracy of investigations, and reduce missed detections of attacks. The study also highlights the need to solve challenges to make network forensics tools more effective for both organisations and law enforcement. However, since the taxonomy is a classification framework, its benefits are shown in how it guides researchers and practitioners in selecting, developing, and applying forensic methods. All in all, the taxonomy aims to improve accessibility to network infrastructures and artefacts, assist in evidence collection, and reduce false negatives, serving as a roadmap for further improvements of network forensics techniques.

Another notable study by Pilli et al. [8] presents a survey on network forensics frameworks and tools, while proposing a fine-tuned generic process model for network forensics. Based on existing digital forensic models, their proposed framework highlights steps such as traffic capture, event correlation, trace examination, and attribution. The authors also provide definitions and categorisations of network forensics concepts, helping to clarify its role within cybersecurity. In addition to reviewing various network forensics analysis and monitoring tools, the study identifies research gaps, particularly in tool integration, process standardisation, and real-world applicability. To address the identified limitations, the authors mention the development of a network forensics system that aggregates open-source tools. By highlighting both conceptual and implementation-level challenges, the study lays the foundation for the advancement of forensics tools focused on improving attribution, response, and evidence admissibility.

In contrast to typical approaches that focus primarily on capturing and storing network traffic, Joy et al. [60] explore the extraction of high-level application usage patterns from low-level network metadata. Their work highlights a weak point in existing network forensics analysis tools. More specifically, the inability to effectively interpret user behaviour. By demonstrating how user interactions, such as web browsing, instant messaging, and file sharing, can be deduced from metadata, the authors propose a more contextualised approach to network forensics. Their prototype framework visualises these interactions, helping reduce analysis fatigue and improve situational awareness. This shift from raw packet to behaviour-centric analysis improves network forensics capabilities, especially for cases involving complex user activity. The study also outlines future work aimed at scaling the system to accommodate large volumes of network traffic. As such, it sets the stage for network forensics analysis tools that are not only technically capable but also practically useful in real-world investigations.

3.2 Data Visualisation

Research in cognitive science suggests that humans are inherently better at identifying visual patterns than textual or numerical ones [11], making data visualisation a natural fit for forensic analysis. In this context, O'Connor and Walley [61] propose a generic framework for data analysis and visualisation based on pattern recognition, originally applied to biological monitoring of river water. Their system processes multivariate data through clustering and visual ordering, before presenting the results in the form of interactive feature maps and indicator templates. Although developed for ecological datasets, the

framework's emphasis on making complex patterns accessible to non-specialists is relevant to forensic scenarios. The system highlights how visual representations of multivariate clusters can enhance pattern recognition and reduce cognitive load, aligning with the objectives of this study to support network forensics analysts through more intuitive visual interfaces. This framework provides both a methodology and functional principles for the development of visualisation tools in other high-volume, high-complexity fields such as cybersecurity.

In addition to improving visual processing, visualisation tools benefit from methods that enhance the statistical reliability of observed patterns. A study by Savvides et al. [62] presents a framework for assessing the statistical significance of patterns identified during visual data exploration. Traditionally, patterns discovered through visual inspection are viewed with scepticism. This work challenges this by showing that the knowledge of the analyst can reduce the number of comparisons. The framework includes tabular and time-series data and supports various test statistics and null distributions, enabling users to distinguish between meaningful and coincidental visual patterns in real time. The framework lays a foundation for visual analysis tools in complex domains, such as network forensics, where iterative exploration of high-dimensional data is common. Incorporating such significance testing into network forensics visualisation tools could reduce false positives, increase user trust, and connect exploratory and confirmatory analysis.

Complementing these technical approaches, a recent study by Sturdee et al. [63] explores how visual representations can help with understanding complex cybersecurity concepts. By asking participants to create hand-drawn sketches illustrating their mental models of cybersecurity concepts, the authors demonstrate the utility of visual tools in articulating and communicating complex concepts. They propose that such sketches can serve as the base for a shared visual vocabulary in cybersecurity interfaces and research, allowing for clearer communication and improved cognitive processing.

3.3 Research Gap

Despite the advances in network forensics research, several important gaps remain unaddressed. Existing studies have proposed taxonomies for classifying forensic techniques [58], process models for network forensics workflows [8], and methods to infer user behaviour from network metadata [60]. However, many current network forensics tools still struggle to manage the large volumes and complexity of network traffic data. Analysts often face difficulties in filtering relevant information and detecting indicators of malicious activity due to the limitations of these tools, which rely heavily on manual data exploration and the analyst's expertise [9], [10].
Furthermore, while visualisations have been recognised as a valuable approach to improving pattern recognition and reducing cognitive load during the analysis of complex data [11], [61], their integration within network forensics tools remains limited. Existing forensic tools primarily present data through raw packets and sessions rather than through interactive visual interfaces tailored to analysts' specific needs. This shortfall hinders analysts' situational awareness and slows down the analysis.
Lastly, there is a notable absence of user-centred design approaches that directly involve network forensics analysts in the development of visualisation tools. Few studies have systematically linked analyst tasks and challenges with appropriate visualisation techniques or produced functional prototype tools evaluated for practical effectiveness in real-world contexts [63].
This study aims to address these gaps by conducting an in-depth investigation of network forensics analysts' workflows and challenges, identifying visualisation methods best suited to their tasks, and designing a prototype interactive visualisation tool that supports efficient exploration and interpretation of complex network data. The study also aims to evaluate the prototype's impact on analysts' efficiency and effectiveness, thereby contributing a user-centred and practically validated visualisation approach to the field of network forensics.

4 Methodology

This section outlines the research approach taken to acquire the information necessary to answer the research question. Each step of the process is described in a dedicated subsection, aligned with the sub-questions introduced below. The methodology follows a structured progression, starting with the formulation of research questions, followed by identifying key tasks of network forensics analysis through interviews, pairing tasks with visualisations using the What-Why-How framework, iterative prototyping, and concluding with an impact analysis. These steps collectively provide the results presented in Section 5.

4.1 Research Questions

The aim of this research is to develop a visualisation tool that effectively supports network forensics analysts during their analysis processes. To achieve this objective, the following research question is proposed:

How can network data visualisations be designed to support forensic analysts in overcoming challenges and improving the efficiency of their workflows?

Given the broad scope of this question, it is divided into four sub-questions to guide the research in a structured manner. Each sub-question targets a specific aspect of the problem, together contributing to answering the main research question.

The first sub-question focuses on the extraction of tasks for which the visualisation tool will be used:

SQ1: What are the key tasks and challenges that network forensics analysts face during the analysis process?

Understanding the tasks and challenges encountered by analysts is the first step toward identifying areas where visualisations can offer meaningful support. The insights gained from this sub-question will impact the design process, ensuring the tool addresses real-world needs.

Building on this understanding, the second sub-question explores the mapping of tasks to appropriate visualisations:

SQ2: Which types of visual representations best support specific tasks or challenges in network forensics analysis?

Once the tasks and challenges are identified, the next step is to determine the most effective visualisations for each. Since mismatched visualisations can hinder rather than help, this question ensures that they are selected on clarity and usability.

With suitable visualisations identified for each task, the third sub-question considers their integration into a functional prototype:

SQ3: How can these task-visualisation pairs be integrated into the design of a prototype tool?

Only mapping tasks to visualisations does not provide an answer to the main question. This question focuses on how to incorporate these mappings into a functional prototype, allowing for practical evaluation of the proposed solutions.

Finally, the fourth sub-question assesses the real-world effect of the tool's use in practice:

SQ4: What impact does the use of the prototype have on the analysts' efficiency and effectiveness during analysis?

The answer to this question tells us if the designed visualisation tool improves, impairs, or does not change the analysis process of network forensics analysts. Finally, using this answer, an answer to the main question can be formed.

4.2 Interviews

The first step in developing a visualisation tool for network forensics analysts is to identify the key tasks and challenges that need to be supported through visualisations. To learn what they are, analysts were interviewed about their typical workflows and challenges. We consulted with a network forensics analyst while creating the questions to ensure that they all made sense and to minimise the interview time by filtering out unnecessary questions. The interview is conducted with four anonymous network forensics analysts and takes approximately one hour each. Their answers are written down instead of recorded to ensure this anonymity. To ensure that participants provide their perspectives rather than adapting their responses to the perceived goals of the research, the analysts are initially told the partial truth that the goal of the interview is to improve their workflow. The actual goal of developing a visualisation tool is only disclosed later during the visualisation section of the interview. To get an answer to our first sub-question without introducing our own biases, the interview begins with an open-ended question that allows analysts to describe their process on their terms:

Could you walk me through a typical case from start to finish? Explain each step clearly and why you performed that step.

Although potentially lengthy, the answer to this question will cover most of the tasks and challenges network forensics analysts typically face during their analysis. However, additional information is still needed to complete the requirements for the visualisation tool. Therefore, the interview also includes structured questions concerning the analysts' **workflow**, the **data** they deal with, and their use of **visualisations**.

4.2.1 Workflow

This section consists of four questions designed to uncover the steps of the analysis process that need visual support. These questions will go into the personal experience of the analysts, hence the answers given by different people can vary greatly. When discussing the interview with the consulting analyst, they noted that a significant part of the workflow consists of assigning various roles to entities within the network. These roles range from victims to different types of adversaries. Thus, the first question is as follows:

What roles do you typically assign to a host within the network when determining its function? How do you decide what role to assign?

As there are many roles to assign, which can differ per analyst, it is important to find out what roles analysts most frequently assign and how they decide that the role fits best. With this information, we can then design visualisations that can assist the analysts with that task. To gain a fuller understanding of the analyst's workflow, it is also essential to learn about the other actions they perform. We ask them the following two questions:

- **What are the three most frequent actions you perform during the analysis process?**

- **Which actions take the most time to perform during your analysis?**

As an analysis can consist of many actions, we have to narrow them down to create the supporting visualisations within a realistic time frame for this research. The focus was put on the actions that are performed most frequently and the actions that take the most time to perform. By supporting these actions with visualisations, we hope to make them more efficient, thus saving a significant amount of time overall. To complete our understanding of the analyst's workflow, we also ask them a more subjective question to identify the weak points of the analysis process:

What parts of the analysis process do you think would benefit the most from change?

Although the answer will be subjective and differ between analysts, this question reveals where the analysts themselves see the most need for visual support, completing our understanding of their workflow.

4.2.2 Data

This section focuses on the data used in the analysis. Understanding the types of data analysts use and how they handle it is essential for selecting appropriate visualisations for certain tasks. By gathering this information, we can narrow the suitable range of visualisations and fine-tune them to fit this context, as the visualisations that can be used depend on the structure and volume of the dataset. An important aspect of understanding the dataset is knowing how it is typically aggregated, as different aggregations need different visualisations. Therefore, we ask the following question:

What levels of aggregation do you most commonly use?

In the context of network data, *aggregations* refer to the transformation of raw data through the grouping or summarisation of various aspects within the dataset [64]. This can, for example, be grouping on time intervals between sessions, IP addresses or protocols. Knowing what the more commonly used data aggregations are reveals what parts of the dataset need more focus and what parts can be mostly ignored. Besides aggregations, there is also another important form of data transformation called data enrichment. This concept is delved into with the second question:

What kinds of data enrichment do you use during your analysis?

Data enrichment refers to improving the quality of raw data by adding other relevant data from external sources [65]. In this context, data can be enriched with geolocations, domain names, blacklists, and many other types of data. As we only have a raw dataset, we need to know what enrichments are frequently used so they can be included in the visualisation tool. Without insight into the data enrichments used in practice, the visualisation tool risks being misaligned with the needs of the analysts.

The final two questions of this section aim to gather information regarding the dataset size:

- **Generally, how large are the datasets you analyse, approximately?**
- **When analysing large datasets, which aspects do you believe should be visualised in real-time versus included in a final report?**

It is important to know what dataset sizes we can expect, so we can ensure that our visualisations stay clear and understandable. Some visualisations scale better when handling large datasets. An example is the difference between a simple table and a network chart. A table can hold a vast amount of data with numerous fields while still being easily readable. On the contrary, a network graph can quickly become unreadable for larger datasets. If the number of nodes and edges becomes too large, the displayed data starts to overlap and becomes obfuscated.

When a dataset becomes too large for some important aspects to be visualised in real-time, alternative strategies need to be used. In this case, we can generate a report that does not show the data in real-time, but rather waits till all data is visualised. This report contains various types of important data that do not need to be shown in real-time. However, visualising all non-real-time data in a report is unrealistic. Therefore, we need to narrow down the report to the aspects that are important to the analysts. This final question gives us an outline of what needs to be included in the report and what can be visualised in real-time.

4.2.3 Visualisations

Visualisations can provide a lot of support during the analysis process, so it is likely that the analysts currently already utilise visualisations. As these are already used by analysts in practice, they are likely to be useful for analysis within this context. For that reason, we need to know more about these specific visualisations so they can be included in the design of the visualisation tool. Including these visualisations in the tool may streamline the analysis process by removing the need for the creation by hand or the use of various visualisation tools. To uncover what these visualisations are, we ask the following question:

Do you use any existing visualisation tools during the analysis? If so, which ones do you use and how do they help with your analysis process?

This question aims to identify the specific types of visualisations that analysts already rely on. This gives us some visualisations that can be directly included in the tool as they have already been shown to have practical value. The visualisations might be created by hand, but they can also be generated using existing tools. To figure out which tools the analysts use, we ask the following:

Do you use any existing visualisation tools during the analysis? If so, which ones do you use and how do they help with your analysis process?

Understanding which tools are used in practice helps find the interactions and features that enhance the user experience enough for the analyst to prefer the tool over manual creation. These features should be considered for the new tool, as they might be important for its value to the analyst.

4.2.4 Wrap-up

Finally, we provide the analysts with an opportunity to bring up any information that they believe to be relevant to this research. Since they are most familiar with their workflows, they may know important information that was not addressed in the previous questions. To also cover these potential insights, we ask the following final question:

Is there anything important that we have not discussed that you think would be valuable for this interview?

After this question, the interview concludes. Next, the transcripts of the analyst's answers need to be carefully read and formed into a detailed list of tasks and challenges. Common answers across multiple interviews will be prioritised during development, as they likely show a shared need among analysts. For a clear overview of all questions, the interview can be found in Appendix A.

4.3 What-Why-How Analysis

After completing the interviews, fitting visualisations must be chosen for each of the extracted tasks. Figuring out what visualisations best fit a certain task can be challenging due to the overwhelming number of possible visualisations for each task and data type. To make this process more structured,

the **What-Why-How (WWH)** framework was created. For this study, the WWH framework described by Tamara Munzner in her book *"Visualization Analysis & Design"* [66] is applied. In this framework, three questions need to be answered to figure out what visualisations best fit a task. **What** data will be visualised? **Why** is the task being performed? And **How** can the data best be visualised? Despite how they may seem, these are not open questions, but from a defined set of structured choices. Each question is provided with a set of abstract answers for extra structure. As the answers to the questions are abstract, there will not always be one best visualisation resulting from the framework. It is meant as a starting guide, not as an all-knowing oracle. By using this framework, certain visualisations can be excluded while creating a more structured overview of the visualisations that do fit. In this subsection, we describe how to answer and interpret each of the three questions, concluding with data visualisation principles that indicate when to choose various visualisation types.

4.3.1 What

The first step of the **WWH framework** provides a structured description of the data that needs to be visualised for a given task. This begins with identifying the correct data type from a set of five: **Items**, **Attributes**, **Links**, **Positions**, and **Grids**. Table 1 provides an overview of these types, including definitions and examples.

Type	**Definition**	**Examples**
Item	An individual, discrete entity.	People, genes, cities.
Attribute	A specific property that can be measured, observed or logged.	Salary, price, number of sales, temperature.
Link	A relationship between items, typically within a network.	Hyperlinks between web pages, roads connecting cities, calls between phone numbers.
Position	Spatial data, providing a location in 2D or 3D space.	Latitude-longitude as position on earth, three numbers specifying a location within space measured by a medical scanner.
Grid	Specifies the strategy for sampling continuous data in terms of both geometric and topological relationships between its cells.	Tile maps in 2D games, agricultural plots sampled in a rectangular pattern for soil or crop analysis, EEG/MEG sensor arrays on a fixed grid for brain activity mapping.

Table 1: Data types, their definitions, and examples

Next, the availability of the dataset must be specified. There are two options: **static** or **dynamic**. **Static** meaning the full dataset is available at once, and **dynamic** meaning the dataset is streamed over time during the visualisation process.

If **Attribute** has *not* been chosen as a data type, the **What** step ends here. However, if **Attribute** *is* chosen, a further specification is required for the attribute type. There are three types to choose from: **Categorical**, **Ordered Ordinal**, and **Ordered Quantitative**. These are described in Table 2, along with definitions and examples.

Type	Definition	Examples
Categorical	Does not have implicit ordering. Can only distinguish whether two things are the same or different.	Favourite fruit, names, file types.
Ordered Ordinal	We cannot do full-fledged arithmetic, but there is a well-defined ordering	Shirt size (small, medium, large), movie top 10 list.
Ordered Quantitative	A measure of magnitude that supports arithmetic comparison.	height, weight, temperature.

Table 2: Attribute types, their definitions, and examples

Finally, if either **Ordered Ordinal** or **Ordered Quantitative** has been chosen as an attribute type, the ordering direction must be specified. Ordering direction defines how values are arranged conceptually and visually. There are three types of ordering directions: **Sequential**, **Diverging**, and **Cyclic**. Descriptions and examples of each can be found in Table 3.

Type	Definition	Examples
Sequential	There is a homogeneous range from a minimum to a possible maximum value.	Age, income, test scores
Diverging	Can be deconstructed into two sequences pointing in opposite directions that meet at a common zero point.	Temperature, strength of glasses, financial returns from investments.
Cyclic	Values wrap around back to a starting point rather than continuing to increase indefinitely.	Time of day, seasons, musical notes in an octave.

Table 3: Ordering directions, their definitions, and examples

4.3.2 Why

The second step of the **WWH framework** offers a structured overview of the purpose behind the visualisation of a given task. This step breaks tasks down into two main components: **Actions**, which describe the user's goal when interacting with the visualisation, and **Targets**, which describe the specific aspects of the data that are of interest.

Actions. Users interact with visualisations through three types of actions: **Analyse**, **Search**, and **Query**. All user tasks involving a visualisation can be described using one or more of these action types.
Starting with **Analyse**, these actions are further divided based on the user's intent into two categories: **Consume** and **Produce**. Actions with the **Consume** goal aim to acquire information, while actions with the **Produce** goal involve the creation or transformation of information. The six specific **Analyse** actions are: **Discover**, **Present**, **Enjoy**, **Annotate**, **Record**, and **Derive**. A full overview of these is provided in Table 4.

Goal	Type	Definition
Consume	Discover	Using visualisations to find new knowledge that was not previously known.
	Present	Using visualisations for the communication of information, telling a story with data, or guiding an audience through a series of cognitive operations.
	Enjoy	Casual encounters with visualisations. The user is driven by curiosity.
Produce	Annotate	The addition of graphical or textual annotations associated with one or more pre-existing visualisation elements. Typically, a manual action.
	Record	Save or capture visualisation elements as persistent artefacts.
	Derive	Produce new data elements based on existing data elements.

Table 4: Actions of the Analyse type, their goals, and definitions

The next set of Actions is called **Search**, consisting of four different types: **Lookup**, **Locate**, **Browse**, and **Explore**. These types describe how users search through a visualisation based on what they know about the target (what they are looking for) and its location (where it is in the visualisation). Table 5 contains an overview of these actions, including their definitions.

Type	Definition
Lookup	User already knows both what they're looking for and where it is.
Locate	A known target at an unknown location.
Browse	User doesn't know exactly what they're looking for, but they do have a location in mind.
Explore	User doesn't know what they're looking for or what the location is.

Table 5: Actions of the Search type and their definitions

In contrast to **Search**, which focuses on user behaviour, the next set of actions, called **Query**, focuses on user intent. It describes what kind of answer the user is trying to get from the data: whether they are trying to **Identify**, **Compare**, or **Summarise** information from the visualisation. Table 6 shows an overview of each of these types and their corresponding definitions.

Type	Definition
Identify	Single target.
Compare	Multiple targets.
Summarise	Full set of possible targets.

Table 6: Actions of the Query type and their definitions

Targets. When using visualisations, users are typically interested in one or more of four types of targets: **All Data**, **Attributes**, **Network Data**, and **Spatial Data**. These target types represent the different aspects of a dataset that may be of interest during visual analysis.
The first type, **All Data**, refers to points of interest within the complete visualised dataset. This category includes three target types: **Trends**, **Outliers**, and **Features**. For definitions and examples of these target types, see Table 7.

Type	Definition	Examples
Trends	High-level characterisation of a pattern in the data.	Increases, decreases, peaks.
Outliers	Data that doesn't follow the main pattern.	Sudden spikes, sharp drops, isolated points.
Features	Definition is task-dependent, meaning any particular structures of interest.	Periodic gaps, directional flows, user behaviour patterns, symmetry.

Table 7: Targets of the All Data type, their definitions, and examples

In addition to full-dataset patterns, interesting aspects can also lie within attributes of individual data points. These aspects can be described using six distinct types: **Distribution**, **Extremes**, **Dependency**, **Correlation**, and **Similarity**. These types can be grouped based on the number of attributes they involve, which can be either one or many. Table 8 provides definitions for each type along with the number of attributes they concern.

Amount	Type	Definition	Examples
One	Distribution	Distribution of all values for an attribute.	Salary range in a company.
	Extremes	The minimum or maximum value across a range.	Highest temperature recorded, lowest stock price.
Many	Dependency	Values of one directly depend on those of another.	Fuel consumption depends on driving speed.
	Correlation	Tendency for values of one to be tied to those of another.	Height correlated with weight, hours studied correlated with exam scores.
	Similarity	A quantitative measurement calculated on all of their values, allowing attributes to be ranked with respect to similarity.	How closely two movies' viewer ratings match across genres.

Table 8: Targets of the Attributes type, their definitions, number of attributes involved, and examples

Finally, there are two target sets that each contain a single type: **Network Data** and **Spatial Data**. For **Network Data** the point of interest is its topology, specifically paths of one or more links that connect two nodes. For **Spatial Data**, it concerns its geometric shape, such as country borders, room layouts, or natural features like rivers and lakes.

4.3.3 How

The final step of the **WWH framework** is the **How** step. This step addresses how data can be visualised and involves many considerations, each with multiple options. To lessen the complexity of the analysis, we opted to only show the outcome of this step. We focus on how the outcomes of the **What** and **Why** steps help narrow down the range of suitable visualisation techniques for a given task. This, in turn, allows for a reasoned and justifiable choice of visual representation. An example of such reasoning is the visualisation of data with **Item** as data type, **Compare** as action, and **Distribution** as target. Here, we want to compare two or more items in order to find their distribution within the dataset.

4.3.4 Example Analysis

To provide a better understanding of the **WWH** analysis, we walk through each step of the process using an example task that is understandable even without knowledge of network forensics. Consider the following task:

Determine whether cities with higher air pollution levels also have more hospital visits for asthma.

What. We start by selecting the data types. For this task, the data types are **Item** and **Attribute**. The **Item** type represents the cities as they are individual, discrete entities. The **Attribute** type represents the air pollution level and the number of hospital visits concerning asthma for each city. For the dataset availability, both **Static** and **Dynamic** are possible choices. Since both are possible, we choose **Static** because it is the more common type. As the **Attribute** type has been chosen, we now must also define the attribute types. Both air pollution and hospital visits are numerical values that support arithmetic operations and therefore fit the **Ordered Quantitative** type. Lastly, since these are ordered types, we specify their ordering direction. Both pollution levels and hospital visit counts increase along a continuous scale with a defined minimum value and no cyclical wrap-around. This fits the **Sequential** ordering direction. For a structured overview of this step, consult Table 9.

Data Types	Item, Attribute
Data Availability	Static
Attribute Type	Ordered Quantitative
Ordering Direction	Sequential

Table 9: Structured overview of the What step for the example task

Why. For this step, we start by selecting the appropriate types of **Actions**. The most suitable **Actions** types are **Discover**, **Compare**, and **Summarise**. The task includes learning if there is any correlation or pattern that may not be known in advance, aligning with the **Discover** type. As the user wants to compare cities based on two attributes to see how they relate, the **Compare** type also fits. The user does not only want to compare two cities, but also create an overview of all cities in their dataset to increase the reliability and validity of their findings. Hence, the final type is **Summarise**. To finish this step, we describe the **Targets**. First, since the user's primary interest is to identify if higher pollution correlates with more hospital visits for asthma, the **Correlation** is selected. As the user is looking for a general pattern within the data, more specifically, if more pollution causes an upward trend of asthma cases, the **Trend** type also fits. See Table 10 for a summary of the selected **Actions** and **Targets**.

Actions	Discover, Compare, Summarise
Targets	Correlation, Trend

Table 10: Structured overview of the Why step for the example task

How. Now that the **What** and **Why** steps have been completed, we can narrow down our options for visualisations. For this task, the two visualisations that fit its requirements are a scatter plot and a grouped bar chart. A scatter plot is effective when working with two **Ordered Quantitative** attributes that follow a **Sequential** scale, such as air pollution and hospital visit rates. Since the data is **Static**, a scatter plot can display the entire dataset in a single, coherent view. Each point represents a city, with axes corresponding to pollution levels and asthma-related hospital visits. This directly supports the **Discover** and **Compare** actions by allowing viewers to identify patterns or relationships and make comparisons between cities. It

also allows for the **Summarise** action by providing an overall view of the data distribution, especially when enhanced with trend lines. This makes it particularly well-suited to revealing **Correlation** and **Trend**, the **Targets** of this task.
In contrast, a grouped bar chart could show pollution and hospital visits side-by-side per city, but as the dataset scales, it becomes cluttered and difficult to interpret. Identifying overall patterns or correlations from a bar chart becomes increasingly challenging with a large number of cities. Therefore, based on the WWH analysis, a scatter plot provides the most effective and interpretable means of visualising the relationship between air pollution levels and asthma-related hospital visits.

4.3.5 Data Visualisation Principles

Data visualisation is possible through a wide variety of visualisations, each optimised for specific types of data. The effectiveness of a visualisation depends not only on its design but also on the characteristics of the underlying data. Selecting the right visualisation type based on the data is therefore a crucial step in ensuring clarity, accuracy, and interpretability.

The items below include both full visualisation types and design elements. Visualisation types are stand-alone ways to represent data, while design elements are features applied within visualisations to enhance meaning and clarity.

Pie Chart. Pie charts are best used for part-to-whole relationships with a small number of categories. More specifically, pie charts are only suitable for data with five categories or fewer, since more would quickly clutter the chart. Furthermore, they can only be used for proportional data where the sum of all parts equals 100%. For detailed and precise comparisons, pie charts should be avoided as humans generally have difficulty judging angles [67]–[69].

Bar Chart. Bar charts are best used for comparing discrete categories or grouped data by length, which is generally easier for people to interpret than angles or areas. They work effectively with both nominal categories, where the order of the groups does not matter, and ordinal categories, where the order conveys meaningful information. They can handle larger numbers of categories than pie charts, but still risk overcrowding [70]–[72].

Grouped Bar Chart. Grouped bar charts are useful for comparing sub-categories within multiple groups, placing bars for each subgroup side by side for direct comparison. They are effective for showing differences across multiple categorical dimensions, making them suitable for datasets with a moderate number of series where both group-level and subgroup-level comparisons are important. However, they become difficult to read if there are too many groups or sub-groups [73].

Bucketed Bar Chart & Histogram. Bucketed bar charts and histograms are best for displaying distributions of continuous numeric data that have been divided into discrete intervals or "buckets." Histograms typically have no gaps between bins to emphasise continuity, whereas bucketed bar charts may have gaps and are sometimes used for ordinal or discrete groupings. They are commonly used to show how data points are distributed across ranges, making it easier to identify patterns such as skewness or bimodality [70].

Line Chart. Line charts are most effective for visualising continuous data across an ordered dimension. They are well-suited for identifying patterns and seasonality in time series datasets, allowing the audience to track changes and compare values at different points along the sequence. They are less suitable for discrete categories with no logical ordering [74], [75].

Variance Chart. A variance chart is a variation of the line chart designed to highlight deviations from a reference baseline, such as budget versus actual performance. It is most effective when the primary focus is on the magnitude and direction of the differences rather than the absolute values themselves [76].

Scatter Plot. Scatter plots are best for exploring the relationship between two continuous variables. They are valuable for detecting correlations, identifying clusters of similar values, and spotting outliers that deviate from the overall pattern. They require sufficient data points to reveal trends meaningfully [77], [78].

Bubble Plot. A bubble plot extends the scatter plot by encoding a third quantitative variable in the size of each bubble. This allows for visualisation of relationships among three variables, but it requires careful scaling to avoid perceptual distortions that could mislead interpretation. Including a legend for bubble size can help prevent misreading [79].

Parallel Coordinates Plot. Parallel coordinates plots are designed for high-dimensional, multivariate datasets. Each variable is represented by a parallel vertical axis, and individual data points are drawn as connected lines across the axes, making it possible to detect patterns, clusters, and correlations in complex datasets. These charts work best when the data is normalised or scaled, as large ranges can dominate the visual pattern [80].

Heatmap. Heatmaps use variations in colour intensity to show the magnitude of a variable across two dimensions. They are particularly effective for identifying patterns, clusters, and anomalies in large datasets, making them popular for correlation tables, performance metrics, and visualising cyclic or temporal data such as daily, weekly, or seasonal trends. Careful choice of colour scale is critical, as inappropriate scales can exaggerate or hide differences [81].

Network Graph. Network graphs represent entities as nodes and their relationships as connecting edges. They are suitable for visualising systems with complex link structures, such as biological pathways and social networks, enabling the discovery of central actors and key connection patterns. They can become cluttered with large networks unless filtering or clustering is applied [82].

Colour. Colour is a powerful visual encoding channel used to highlight differences or group related elements. While it can enhance clarity and impact, it must be applied thoughtfully to avoid issues such as misinterpretation from colour blindness or perceptual biases. Diverging, sequential, and categorical colour schemes should be chosen based on data type [83].

Labels. Labels provide precise textual identifiers for visual elements, ensuring the audience can directly associate data points or categories with their names or values. They are essential in cases where accuracy and clarity are paramount. Excessive labels can cause clutter, so selective labelling is recommended [84].

Badges. Badges are small symbolic indicators used to convey discrete statuses, category memberships, or alert signals. They are effective for quickly communicating categorical states in dashboards or summarised overviews where space is limited. Overuse can dilute their effectiveness [85].

Text. Text is used to present qualitative or categorical information where exact wording matters. It is ideal for definitions, explanations, and nuanced context that cannot be conveyed visually. However, excessive reliance on text can reduce visual engagement [84].

Table. Tables are optimal for presenting structured, relational data where precise numeric or categorical comparisons are required across multiple variables. They are most effective when the audience needs to look up specific values rather than discern broad patterns or trends. They are poor for showing trends at a glance and should be paired with charts for pattern recognition [86], [87].

4.4 Prototyping

The third step in developing a visualisation tool for network forensics analysts is to transform the task-visualisation pairs derived using the WWH framework (see Section 4.3) into a functional prototype. The goal of prototyping is to develop an interactive system that allows for the evaluation and refinement of the design through user feedback.

The prototyping process consists of three main phases: (1) concept design, where the prototype page division and layout are defined, (2) mockups, where the design and user interactions are refined and styled, and (3) implementation, where the design is translated into functional software using appropriate technologies and tools.

4.4.1 Concept Design

The prototyping process begins with outlining an abstract version of the visualisation tool based on task-visualisation pairs identified through the WWH analysis. There are likely to be a large number of pairs with various purposes associated with them. To manage this, the pairs are grouped into pages according to their analytical purpose and underlying data types. Each page contains visualisations and interactions that support the analysis. The pages are interconnected to provide a cohesive experience that resembles a complete application rather than separate dashboards.

As discussed in Section 4.3.3, the WWH framework does not necessarily narrow the visualisation options down to a single one. There can be multiple visualisations that suit a given task. Selecting the most appropriate one requires clear argumentation for why that visualisation fits best. Since visualisations vary in their optimal container sizes, the choice of visualisation must therefore be made before deciding on the layout of the pages to ensure clarity and effective use of space.

After selecting the final visualisation for each task-visualisation pair, an abstract version of each page is created. This involves producing multiple low-fidelity paper sketches of each page to explore various layout options and identify the most effective one. During this phase of the process, detailed elements are omitted. Instead, abstract placeholders representing visualisations are used to determine the most intuitive and effective layout. More detailed elements are added only after finalising the layout of each page. However, these details are conceptual rather than fully developed components and only indicate locations and ideas. As details are likely to change at a later stage, adding them prematurely would unnecessarily increase development time.

By the end of this phase, pages and layouts are established with placeholders indicating visualisation placement. This allows the next phase to focus entirely on refining content and interactions, without concern for layout or page division.

4.4.2 Mockups

Following the creation of the layout sketches, medium-fidelity mockups can be developed using those sketches as a foundation for the placement of components. These mockups are created within a design and prototyping tool called Figma [88]. This tool facilitates functionalities that allow for the creation of detailed and fine-tuned mockups, which can then be exported in various formats. Within Figma,

components can be easily created and reused, making the process of creating mockups more efficient.

To ensure that the visualisation tool is accessible to a broad range of analysts, an appropriate colour scheme must be selected during this phase of the process. While this may seem trivial, the choice of colour scheme is especially relevant for visualisations, as an improperly chosen palette can hinder interpretability for users with colour blindness. For instance, in a grouped bar chart, improperly chosen colours may prevent colour-blind users from being able to distinguish between groups, rendering the visualisation useless. Therefore, a colour-blind-friendly palette must be chosen to mitigate such situations. The need for such a colour palette is even more pressing because cybersecurity is a male-dominated field [89], and colour blindness affects approximately 1 in 12 men compared to 1 in 200 women [90].

After creating the initial mockups, feedback is requested from the consulting network forensics analyst. This feedback may include suggestions such as changing the type of a visualisation or removing one entirely. Incorporating this feedback into the tool not only enhances the effectiveness and correctness of the visualisations and overall design but also ensures that the analyst's interview responses were correctly interpreted.

Upon completing this phase, the medium-fidelity mockups together represent a non-functional concept of the final visualisation tool. This helps prepare for the implementation by making design-related decisions in advance, allowing the focus to be placed entirely upon implementation details.

4.4.3 Implementation

Following the completion of the mockups, they can be utilised as the base for a functional prototype. This prototype contains interactive visualisations divided over various pages as its frontend, presenting actual data acquired from Arkime in the backend. Git is used for version control to ensure that previous versions of the code can be restored if necessary. The code is stored in a GitHub repository, providing a secure, centralised backup and enabling easy access from different devices.

Using Arkime's Viewer API (versions 3.x to 5.x) [91], data added to Arkime can be accessed, and various additional functionalities can be executed. The API also includes built-in aggregations that analysts, not using Arkime, would otherwise need to carry out manually. The main functionality that will be used within the visualisation tool is the `/sessions` API, which returns all session data in JSON format, based on the applied parameters. This endpoint supports both `GET` and `POST` requests with the parameters included in the query or request body, respectively. Table 11 presents these parameters and their descriptions.
To access Arkime's web service or API, users are required to create an account with a username and password. Access control of the API is enforced through a mechanism called digest authentication, a secure authentication method that ensures the safe transmission and verification of user credentials. When a user attempts to access the API, Arkime issues a digest challenge, requiring the user to respond with an encoded combination of the username, password, and other digest-specific parameters [92]. This interaction ensures that user credentials are not transmitted in plain text.
It is important to note that the field names used in the Arkime API differ from those displayed in the web service. Instead of requiring the field names as shown in the web service, the API requires the field names used by Arkime's underlying Elasticsearch or OpenSearch database [91]. Additionally, the API documentation does not provide a list of the correct field names. The most effective way to identify the correct field names is by using browser developer tools to inspect API requests sent by the web service, which include the actual field names used by the API.

To request session data from Arkime and perform the necessary aggregations for the visualisations, the tool first requires a backend server. This server is created using the Go [93] programming language. As the number of sessions returned by the Arkime API can go up to 2,000,000, as seen in Table 11, a backend

Param	Type	Default	Description
`date`	number	1	Number of hours ago to begin the search from. `-1` indicates all available data.
`expression`	string	—	A URL-encoded search expression string.
`facets`	number	0	Set to `1` to include aggregation information for maps and timeline graphs.
`length`	number	100	Number of items to return starting from `start`, with a max of `2,000,000`.
`start`	number	0	Pagination start index.
`startTime`	number	—	Start of time window in seconds since Unix epoch if `date` is not defined.
`stopTime`	number	—	End of time window in seconds since Unix epoch if `date` is not defined.
`view`	string	—	Name of an Arkime view to apply before the expression.
`order`	string	—	Comma-separated list of fields to sort on. Optionally follow with `:asc` or `:desc` for ascending or descending sorting.
`fields`	string	See below	Comma-separated list of database fields to return.
`bounding`	string	`"last"`	Defines session time boundaries. Options: `first`, `last`, `both`, `either`, `database`.
`strictly`	boolean	false	If set to `true`, only sessions fully inside the time range are returned. Overrides `bounding` with `both`.

Table 11: Request parameters for the `/sessions` API endpoint with types and default values

language must be chosen that can efficiently handle aggregations on such large numbers of sessions. Golang programs are compiled ahead of time and compiled to native machine code, greatly reducing the memory overhead when compared to programming languages that do not, like Java or Python [94]. This allows it to efficiently handle a large number of sessions. Additionally, Golang offers built-in libraries for HTTP and JSON formatting [95], simplifying the use of the Arkime API.

Once session data can be correctly and efficiently fetched from the Arkime API and aggregated within the Golang backend, a frontend is required to display the data in various visualisations. This frontend is made through the use of a progressive JavaScript framework called Vue.js, which allows for changes in the session data to be directly reflected in the UI through its reactive nature [96]. Vue.js, by itself, does not offer type safety for the code. TypeScript [97] can be added to the project to enforce type safety in Vue.js code.
Creating standard components like buttons, cards, and tables can take up a lot of extra development time. Component libraries can be used to avoid spending time on the implementation of such components by offering a selection of pre-made ones. For this tool, PrimeVue [98] is used as a component library due to its wide selection of components, which are easy to give a custom theme to.
Using Vue.js with PrimeVue, an interactive frontend tool with tables and lists can be created, but in this context, other visualisations are required. Creating the necessary visualisations from scratch, while likely to result in more fine-tuned ones, takes a significant amount of development time and is therefore outside of the scope of this study. Hence, a visualisation library can be used that offers various visualisations

to choose from. The visualisations included in the tool are facilitated by a visualisation library called ApexCharts [99]. This library is chosen because of its wide selection of visualisations and customisability options. This should still allow for the required visualisations and interactions to be included within the tool.

By following the steps and using the technologies previously described, a functional prototype of the visualisation tool can be made. Using this prototype, tests can be conducted to figure out if it improves analyst efficiency. A final overview of the prototype structure is presented in Figure 1.

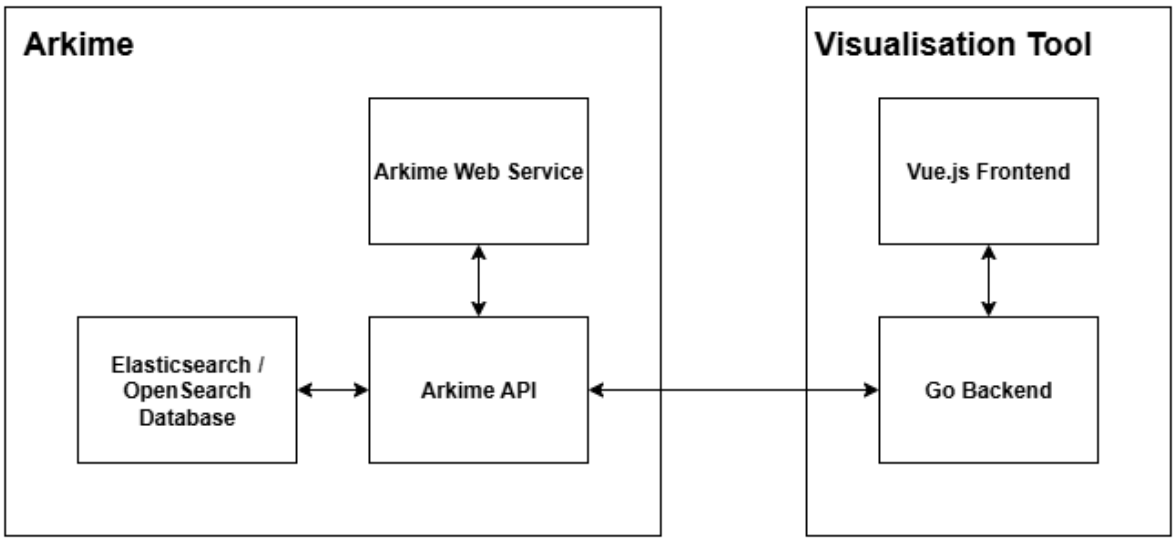

Figure 1: Overview of the prototype structure

4.5 Impact Analysis

After developing the functional prototype, the final step of the study's methodology evaluates its impact on network forensics analysis. This evaluation assesses the effectiveness, intuitiveness, and correctness of the tool, using predefined metrics.

This subsection begins by outlining the context and objectives of the impact analysis. It then provides a description of the evaluation procedure, followed by the criteria and metrics. Next, it presents the participants involved in the study, outlines the evaluation tasks along with bias prevention strategies, and concludes with the study's limitations.

4.5.1 Context

This evaluation compares the prototype described in Section 4.4 with Arkime, a tool frequently used by the analysts interviewed in this study. The comparison assesses whether the new visualisation tool improves the workflows of network forensics analysts, thereby determining whether it achieves the goal stated in Section 1.

4.5.2 Evaluation Procedure

The evaluation takes place as an in-person test lasting approximately one to two hours, during which participants complete 10 short tasks varying in type and length using both Arkime and the new tool separately. The tasks are intentionally designed to be of low complexity, ensuring the evaluation stays within the time limit. While this approach only partially covers the interviewed analysts' workflow, it still

provides a meaningful assessment of the tool's impact on short-term task performance, offering insight into whether the tool has a potentially positive, negative, or neutral impact on network forensics analysis.

As participants may lack prior experience with either tool, a 10-minute documentation review period is provided at the start of the evaluation. Since Arkime includes a documentation page within its web interface, a comparable document is created for the prototype to ensure fairness. The contents of this document can be found in Appendix F. A duration of 10 minutes is chosen because both the Arkime documentation and the prototype's comparable document take approximately five minutes each to scan through.

4.5.3 Criteria and Metrics

To evaluate the tool's effect on the current analysis process, the assessment compares three criteria across the two tools: effectiveness, intuitiveness, and correctness.

Effectiveness. Effectiveness refers to the extent to which the new tool improves the workflow of an experienced analyst. This is measured by comparing the time taken to complete each task individually and the total time for all tasks when using the new tool versus Arkime. The comparison provides insight into whether usage of the tool results in network forensics tasks being completed faster, at the same speed, or slower than with Arkime. A significant reduction in time would indicate higher effectiveness, whereas an increase would indicate lower effectiveness. To obtain a meaningful measure, this test is conducted with an expert network forensics analyst who has prior experience using Arkime, ensuring that the participant can operate both tools efficiently (see Section 4.5.4).

Intuitiveness. Intuitiveness refers to how easily a new user, with no prior experience with either Arkime or the prototype, can use the tool to complete analysis tasks. Like effectiveness, this is evaluated by comparing the time taken for each task and the total time for all tasks between the two tools. The aim is to determine whether the tool's interface and workflows are easy to grasp for new or less skilled analysts. A faster or comparable performance time would indicate higher intuitiveness, while a slower time would suggest a steeper learning curve. This measure is tested with a participant who has basic knowledge in network forensics and computer science but no prior experience with either tool (see Section 4.5.4).

Correctness. Correctness measures the extent to which the tool supports analysts in completing tasks without mistakes and producing correct results. Each task is designed with a single correct outcome, allowing results to be scored clearly and consistently. The number of tasks completed correctly is counted and compared between the two tools. In addition, the evaluator observes the participant's process to record mistakes made along the way, even if the final answer is correct. This dual focus on results and process provides a deeper understanding of whether accuracy is achieved through reliable methods or by chance. Correctness is assessed with both experienced and inexperienced participants for increased fairness.

To evaluate the correctness of the prototype, the correctness rate for each tool is calculated as:

$$\text{Correctness Rate} = \frac{\text{Total Tasks} - \text{Mistakes}}{10} \tag{1}$$

The relative correctness improvement of the prototype compared to Arkime is then computed as:

$$\text{Relative Improvement} = \frac{\text{Correctness Rate}_{\text{Prototype}} - \text{Correctness Rate}_{\text{Arkime}}}{\text{Correctness Rate}_{\text{Arkime}}} \tag{2}$$

Multiplying this value by 100 gives the percentage increase in correctness.

4.5.4 Participants

The evaluation is performed on two distinct participant types. The first type consists of experienced network forensics analysts who have prior experience using Arkime. The second type includes individuals with basic knowledge of computer science and network forensics, but no prior experience with either Arkime or the new tool.
Evaluating the first type allows measurement of the new tool's effectiveness compared to Arkime. Because these analysts are familiar with Arkime, they are likely efficient in locating information and performing tasks with it. However, their lack of experience with the new tool introduces a time overhead when using it. Consequently, while the evaluation can provide insights into the relative effectiveness of the new tool, it does not offer information about its intuitiveness.
To address the new tool's intuitiveness, the second participant type is evaluated. These individuals have foundational knowledge sufficient to understand the tools and context, but lack practical experience with either. Comparing their task performance times between the two tools provides a measure of intuitiveness. If one tool allows for faster task completion, this indicates it is easier for new or inexperienced analysts to use. However, since this type is inexperienced, their results do not reflect the tools' effectiveness for expert analysts.
The evaluation includes only two participants, one of each type, due to the limited availability of suitable candidates within a feasible timeframe. While this small sample size reduces the validity of the results, it provides initial insights into the tool's effectiveness and intuitiveness.

4.5.5 Tasks

The evaluation consists of 10 tasks varying in length and complexity. Each task is designed to assess specific analysis skills relevant to network forensics. The tasks are listed below, with certain elements presented in bold and separated by a slash (e.g., **HTTP / HTTPS**) to indicate alternative versions used in different task sets:

1. Determine the total number of sessions and packets in the dataset.
2. Identify the number of **HTTP / HTTPS** sessions, and find the **most / least** commonly used HTTP method.
3. Find an **IP / MAC** address involved in an abnormally high number of DNS sessions.
4. Given a list of **ports**, their intended **IP protocols**, and **services**, identify an anomaly.
5. Identify an IP address that uses the proxy-related header **Proxy-Connection / Via**.
6. Determine the top non-IP protocol for the top **4 / 5** most frequently occurring **MAC / IP** addresses.
7. An **IP** has been confirmed to belong to a C2 server. Locate it in the dataset and verify that the packet with timestamp **2/28/2023 02:35:01 / 2017/05/26 15:39:57** and size **420 / 48** bytes contains suspicious contents.
8. Determine the year in which the session with the **highest / second-highest** data volume (in bytes) occurred.
9. Calculate or find the percentage of all sessions that involve a confirmed proxy **IP**.
10. Construct and execute a query (in Arkime) that filters all **HTTP / HTTPS** traffic using TCP that also includes the **PSH / ACK** TCP flag and uses the **SOCKS / PPPoE** protocol.

Using the same tasks for both tools introduces potential bias. Specifically, participants performing the second tool's tasks might be faster due to familiarity gained during the first tool's evaluation, which could influence the results. To mitigate this bias, two strategies are employed:

First, the order of tasks is randomised between tools to reduce the likelihood that familiarity with the tasks for one tool unfairly advantages the participant when performing them with the other tool. To ease participants into the evaluation, task 1, the simplest task, is presented first for both tools. While the order of tasks varies between tools, it remains consistent across participants to maintain comparable evaluation conditions.
Second, two equivalent but distinct sets of tasks are created, with each set assigned to one tool. These sets are balanced in length and complexity to minimise the influence of the variation of tasks on the results, while differing sufficiently to prevent learning effects from repetition. The bolded elements, possibly with slashes, in the task list above denote aspects that vary between the two sets.

4.5.6 Limitations

Due to time constraints, the functional prototype is not fully implemented at the time of evaluation. Therefore, the evaluation focuses on the components with the most visualisations and functionality.

The **metadata page** is fully implemented, including all visualisations, user interactions, and the execution of queries built through these interactions.
The **roles page** is partially implemented. Since each role tab shares the same structure, evaluating a subset of roles is considered sufficient. Accordingly, only the two most important roles identified in interviews are implemented along with their associated visualisations and interactions. These roles are Proxy and C2.
The **main overview** is not implemented and is excluded from the evaluation.
Additionally, the query functionality, including its execution, integration with Arkime, and the formatted overview, is completely implemented.

This limitation affects the scope of the evaluation. Because only a subset of the tool's full functionality is available, the results cannot definitively assess the complete tool design outlined in this study. While the findings provide insight into the potential impact of the prototype on network forensics analysis, the actual impact of the fully implemented tool may differ substantially. Furthermore, the evaluation involves only two participants, which limits the ability to generalise the results. Thus, this evaluation serves as an initial indication of impact and a proof of concept.

A second limitation is the clear lack of participants. While the impact analysis still provides insights into the initial impact of the tool, a definitive answer can not be given using this evaluation. To provide a statistically significant and final evaluation, this impact analysis must be repeated with a larger number of participants for each type.

It is important to note that much of the new tool's functionality does not completely overlap with Arkime's features, since both include unique aspects. Rather, the two tools complement each other. As such, the evaluation focuses only on a small subset of the prototype's capabilities that directly align with comparable tasks in Arkime. This complementary relationship means that the tools are intended to be used together, rather than as direct replacements.

5 Results

This section presents an analysis of the findings obtained through the steps outlined in the methodology in Section 4. Each subsection tackles one step in detail, providing all information necessary to answer its corresponding research question in a later section.

5.1 Interviews

This subsection of the report describes findings from four interviews with data analysts. To protect the participants' privacy, we provide only an aggregated summary that does not separate the individual answers per interview and omits any content deemed irrelevant or explicitly withdrawn by the analysts. This approach complies with the requirements set by the Human Research and Ethics Committee (HREC).

The following analysis of the interviews is divided into three parts: (1) recurring answers given by multiple analysts, (2) answers that, although mentioned only once, were considered significant to the remainder of the study, and (3) descriptions of the important roles resulting from the second interview question. A full overview of all tasks can be found in Appendix B.

5.1.1 Overlap

Given that all analysts shared the same job description, some overlapping themes were expected between the interviews. Since our goal is to develop a tool that benefits as many analysts as possible rather than one tailored to the preferences of an individual, we prioritised insights that emerged across multiple interviews.

The most prominent overlapping theme concerns the ability to exclude sessions based on specific attributes of the dataset. This was mentioned multiple times in each interview, referring to different types of data. More specifically, there were mentions of excluding scanners, VPN and TOR traffic, sexually explicit traffic, SSH sessions, and generally any sessions for which the analysis has been completed. Every answer to the first interview referenced this theme, and a multitude of analysts gave filtering as their most frequent action. This overlap highlights the need for an efficient and intuitive filtering mechanism for our visualisation tool.

A further recurring theme was the uncovering and portrayal of the infrastructure of one or more adversaries. This is an iterative process with multiple steps needed for completion. Its importance was evident, as most analysts mentioned this in detail. The tool must therefore aid in the investigation of adversary infrastructures, while also providing a clear overview simultaneously. This should streamline an otherwise lengthy and complicated process.
The importance of infrastructure analysis is further reflected in the need to identify the victims within the dataset. Although identifying victims is part of uncovering the adversary infrastructure, most analysts mentioned this as a separate action to highlight its importance. Consequently, the infrastructure-related features of the tool must also aid in the identification of victims.

All analysts also expressed their wishes for a clear overview of various static data and aggregations. This overview could include various frequency analyses, attribute counts, session and packet counts, and more. Analysts mentioned creating this overview themselves, as they often need it as a starting point for the analysis. This was referred to by one analyst as a **metadata view**, a term we then adopted throughout the remainder of this report. By incorporating the **metadata view** into our tool, we eliminate the need for analysts to create it manually, significantly improving efficiency. In the following, we describe the in-depth requirements for the **metadata view**.

The first aspect most analysts emphasised regarding the **metadata view** was the inclusion of statistics about the dataset. They expressed their need to see the total number of packets, sessions, and duration of

all analysed PCAPs. This information helps them understand the nature of the data under analysis and gives them indications of abnormalities.
Additionally, the majority of analysts mentioned frequency analyses and counts as some of their most performed actions. Frequency analysis refers to a sorted list based on the frequency of a given attribute. In contrast, counts provide the number of occurrences, often without sorting. The interviews also revealed the attributes that the analysts most often perform these analyses on. More specifically, they perform frequency analyses and counts for IP addresses, MAC addresses, geolocations, HTTP methods, protocols, TCP flags, and SSH sessions. By showing attribute counts in a sorted format, these actions can be performed simultaneously. This eliminates the need to create two separate lists.
Besides a frequency analysis and count for HTTP methods, there is another common HTTP-related action mentioned by analysts. They need to distinguish HTTP from HTTPS traffic, as HTTP traffic is not encrypted. This allows them to examine the session content, potentially giving them valuable leads. Hence, the **metadata view** needs to include a way of discerning between HTTP and HTTPS traffic, allowing for one of the two to be filtered out as well.
When answering the question about their use of data enrichments, all analysts mentioned adding geolocations of IP addresses to their datasets. A geolocation shows the estimated physical location of an IP address on Earth. It typically includes a country, city, and coordinates. Although mostly accurate, they should not be seen as the absolute truth, as they can be altered by adversaries or simply inaccurate due to faulty sources. Nevertheless, displaying geolocations in the **metadata view** can help analysts identify suspicious patterns or anomalies.
A further group of metadata involves session-related attributes. The most prominent of this group were the session durations and volumes, as irregularities in either of these can help analysts find a starting point for their analysis. Some helpful session-related attributes require aggregation before they can be utilised. An example of this is the distribution of the number of sessions over weekdays and time of day. This can show the analyst patterns that do not exist in the raw dataset. Finally, as mentioned before, the analysts need VPN, TOR, and sexually explicit sessions to be filtered out. However, these should not be filtered out blindly, as they could be malicious traffic in disguise. Therefore, providing a clear overview of these sessions with easy filtering functionality makes it possible for analysts to keep only the sessions they need, giving them a smaller dataset at the start of the analysis.
Finally, some analysts mentioned looking into the ports appearing within the dataset. More specifically, they look for mismatches between ports and their expected protocols and service anomalies. Most ports have one or more intended transfer protocols and services they can be used with. However, there is no mechanism enforcing this rule. Thus, looking for ports that are used with a transfer protocol or service that does not match the intended ones gives the analyst indicators of malicious behaviour. Since this task is currently performed manually, integrating this into our **metadata view** could save the analysts a significant amount of time.

To carry out all tasks previously mentioned, analysts are required to create and execute queries within Arkime. However, switching between multiple tools to perform queries and view visualisations takes up extra time. Integrating Arkime's querying functionality directly into the visualisation tool can streamline the analysts' workflow and reduce the inefficiency caused by constant context switching.

5.1.2 Distinct but Significant

Not all answers from interview participants overlapped with those of others. Most of these non-overlapping answers were regarded as out of the scope of this study. However, some answers either expanded on answers given by other analysts or seemed important to take into account during the rest of the study.

One notable theme expands on the task of figuring out and portraying the adversary infrastructure. While it is important to know what elements are part of that infrastructure, it is equally important to distinguish the ones that are not. When the size of the dataset increases, the distinction between these two types

of traffic becomes harder to find. Furthermore, some infrastructures can be compromised by hackers. This leads to a mix of legitimate and malicious traffic within the same infrastructure. To detect malicious traffic, analysts examine the IP addresses that connect to suspicious ports, such as those with port-protocol mismatches. Consequently, form clusters that slowly reveal the adversary infrastructure. Once entities are confirmed as adversaries, their connections are investigated and re-clustered to identify additional related adversarial activities. Without a clear indication of these already identified adversaries, the detection of adversaries becomes more difficult and error-prone.

Notably, one analyst stressed that SSH traffic, in particular, calls for a more in-depth investigation. They state that many failed attempts at establishing a connection using SSH may indicate automated scanning. A clear distinction between failed and successful SSH login attempts is vital, as scanners and their traffic can often be excluded from the analysis. For successful login sessions, analysts examine the frequency, patterns, sources, and most prominently, the volume. An unusually high volume may indicate that the sessions contain extra data or files, which is not typical of standard SSH sessions. Such anomalies indicate possible malicious behaviour.

Although most analysts skipped the final interview question, one analyst offered a valuable insight. However, this was not regarding any analysis-related theme, but concerning the visualisations for our tool. The analyst emphasised the importance of keeping our visualisation techniques clear and purposeful. In some cases, a simple list or table conveys the information better than a complex chart. Moreover, consideration must be put into the colour schemes used in charts. Colours should be easy to tell apart, even for users with colour blindness. These insights on the clarity and purpose of the visualisations must be taken into account for the design of the tool to ensure its usability for all analysts.

Another analyst also answered the final interview question. However, rather than offering a tip or missing information, they told us about a feature they missed during the analysis. Currently, there is no way to get a summary of the data that provides immediate insights at a glance. This answer further underlines the importance of the **metadata view** mentioned in Subsection 5.1.1 as this would present multiple data types in a single overview.

One analyst mentioned that the analysis is not always performed on static PCAP files. In some cases, it must be performed on a continuous stream of data. Since Arkime supports such a data stream, our tool should also be capable of acquiring and displaying real-time data to ensure both types of analyses are compatible.
Finally, an analyst requested that we come up with tasks and aggregations on top of the tasks extracted from the interviews. Tasks introduced from an outside view might cover aspects that the analysts overlooked. Therefore, some potentially useful tasks will be created and added to the visualisation tool.

5.1.3 Role Descriptions

In addition to the themes mentioned in the previous two subsections, our consulting network forensics analyst stressed the importance of assigning roles to various entities within the network during analysis. One such role, mentioned in Subsection 5.1.1, is that of victims. While victims only have that one role, adversaries can have a variety of roles that impact how the analysis needs to be performed. The second interview question aims to identify these roles to create visualisations that can aid in their detection and assignment. As an answer to this question, the analysts named the following roles: Proxy, Command and Control (C2), Scanner, and Exfiltration. In the following, we define each role, outline the steps that must be taken to detect it, and explain how it affects the analysis.

Proxy The Proxy role refers to an intermediary server within a network that typically sits between clients and servers. It receives data or requests from one party and forwards them to their intended

destination. In the context of network forensics, the focus is primarily on malicious proxies. These proxies intercept traffic that would normally flow directly between entities, allowing them to observe or manipulate the data in transit.
A first sign that a server might be a proxy is its use of proxy-related protocols such as SOCKS, HTTP proxy, or IPsec. IPsec is not commonly used for general proxies, but for VPN connections. These are considered a special type of proxy. Typically, network traffic involving the HTTP proxy protocol contains one or more of the following proxy-specific HTTP headers:

- `Proxy-Authenticate`
- `Proxy-Authorization`
- `Via`
- `X-Forwarded-For`
- `X-Forwarded-Host`
- `X-Forwarded-Proto`
- `Forwarded`

The presence of these headers within an entity's traffic strongly suggests that it either is a proxy or is communicating through one. However, as these headers can be added manually by an attacker or by faulty server configuration, their presence does not give definitive proof that the server is a proxy. Therefore, further analysis is required for confirmation. To avoid detection and retain functionality, most malicious proxies relay all data they receive to the intended source. This characteristic allows for detection by analysing the input-output byte ratio of a server. If this ratio is close to 1, this means that it receives almost the same volume of data as it sends, indicating that it may be a proxy. By plotting the input and output volumes over time, an analyst can confirm their identity as a proxy. If the input and output match closely, possibly with an offset in time, this is a strong indication of a proxy. However, not all proxies have an input-output byte ratio of 1. Some proxies manipulate the data before relaying, causing the ratio to diverge from 1. Although the volumes may differ, the input and output traffic will display in a similar pattern.
Once a server is confirmed to be a proxy, it impacts the approach of the subsequent analysis. Because traffic passing through a proxy lacks the original destination IP in incoming packets and the original source IP in outgoing packets, it is impossible to correlate the input and output flows using just their attributes. By plotting the input and output volumes over time, the analyst can see which input peaks correlate to which output peaks. By then examining the corresponding timestamps, the analyst can form a complete picture of the traffic. Repeating this process allows the communication network of the proxy to gradually be revealed. If there are input-output pairs that should match but differ in volume, this might indicate tampering, providing the analyst with a further point of investigation.

C2 The C2 role refers to an adversary-controlled entity that sends commands to malware-infected systems. This allows the attacker to remotely control compromised systems through the C2 server, often stealing data in the process. To avoid detection, C2 servers often mimic legitimate traffic patterns. Infected systems periodically send short messages, known as beacons, to the C2 server. These beacons inform the C2 server that the infected system is active and ready to receive more commands.
One method of detecting C2 servers is by inspecting the traffic payload. If this payload can be decrypted or is sent in plaintext, the analyst can directly observe the sent commands. However, well-implemented malware uses strong encryption protocols, making this approach ineffective. In practice, however, encryption is often implemented poorly. If an adversary encrypts the traffic themselves and the encryption protocol is known to the analyst, often the encryption can be brute-forced.
In some cases, the decryption key is transmitted alongside the encrypted payload, allowing for easy decryption. Another detection method involves analysing the timing of beacons sent to the C2 server. Beacons are sent at regular intervals, resulting in a consistent beaconing frequency. By applying a Fast Fourier Transform (FFT), a signal processing technique that reveals repeating patterns in time-series data,

to the timing intervals between messages, the beaconing frequency will be displayed as a distinct peak, indicating that the server potentially is a C2. Often, when C2 servers try to stay undetected, there will be a lot of traffic mimicking legitimate traffic, causing noise in the FFT. This can significantly complicate finding the correct beaconing frequency. In this case, an analyst can make use of the fact that beacons are typically sent using consistent ports. Hence, by performing an FFT separately for various ports, noise can be filtered out.
Once a C2 server is identified, a deeper analysis of the network can follow. First, victims can be identified by looking at the source of the beacons, as each originates from an infected system. It can, however, not be assumed that the direct source of the beacon is a victim. In some cases, a proxy is put between the victim and the server, obfuscating the source of the beacons. Consequently, the process described for proxies in the previous paragraph must be followed to identify the victim. If the timestamps of the beacons are known to the analyst and they are in contact with a victim, they can correlate victim traffic with the beacons, revealing behaviour caused by C2 commands.

Scanner The Scanner role refers to an entity that probes a network by sending a large number of packets to various IP addresses to identify active hosts. It sends packets to a range of ports to identify which ones are open on each active host. Since each port is typically associated with a specific service, scanning ports can also reveal the services running on the host. Sometimes scanners perform what is known as vulnerability scans, where they probe victims for specific vulnerabilities. The goal is to determine whether the victims' configurations align with known vulnerabilities that can be exploited.
Unlike proxies or C2 servers, detecting scanners does not require aggregating traffic over time. Analysts detect scanners by identifying entities that, often after establishing an SSH connection, attempt to connect to numerous IP addresses across a wide and sometimes unusual range of ports. These connections often carry a payload, suggesting that they are scanning for vulnerabilities. If an entity responds, it suggests that a vulnerability has been found. However, most of the connections fail, giving the analyst another characteristic to look for. In special cases, the analyst will be able to see the payload of a connection. Scanners built using CVE detection tools may include the corresponding CVE number within the payload, making detection simple.
Once a scanner is identified, analysts typically exclude its traffic from further analysis. Scanners appear in many networks and often are not malicious. However, they cannot be excluded blindly. Analysts sometimes maintain internal lists containing the baselines for the typical number of connections a scanner initiates. If the observed number of connection attempts a scanner performs differs from what is found in that list, it cannot be excluded and needs further analysis.

Exfiltration The Exfiltration role refers to an entity that collects stolen data from compromised systems and later forwards it to the attacker or another destination. This transfer may occur either at regular intervals or be triggered manually by the attacker. An exfiltration entity can sometimes also serve other purposes. For instance, some C2 servers also simultaneously handle data exfiltration.
To detect exfiltration, analysts search for servers with a large number of incoming sessions over protocols such as FTP or SCP. Since these are file transfer protocols, this suggests that multiple systems are sending files to the server. This data will only be sent to the attacker in long intervals, so there should be much more incoming than outgoing traffic. When the data is sent to the attacker, that session should have a similar data volume as all incoming sessions before, starting after the previous exfiltration.
Identifying the exfiltration server may help analysts trace the attacker, especially as data is sometimes sent directly to the attacker's infrastructure. However, this is not always the case, so further investigation is needed before making that assumption. Additionally, the sources of the file transfer sessions are likely compromised systems and should be investigated as potential victims.

5.2 What-Why-How Analysis

This subsection applies the WWH analysis, introduced in Section 4.3, to the tasks extracted from the interviews in Section 5.1. To enhance clarity, the analysis is structured into three categories based on task type: (1) general tasks, (2) metadata-related tasks, and (3) role-related tasks. Each task group is first introduced with an overview of its corresponding WWH selections, followed by a detailed explanation. As noted in the interviews, the data to be analysed can be either static or dynamic, depending on the context. To simplify the analysis, we treat all data as static wherever possible. This avoids the need to repeat the WWH analysis for both static and dynamic variants of the same task, which would be beyond the scope of this study.
Additionally, analysts requested that we propose potential new tasks or aggregations that might be useful. These are indicated by highlighted rows in the tables and are further elaborated where relevant.

The reasoning for the visualisation choices presented in this section align with the data visualisation principles outlined in Section 4.3.5 as its foundation. These principles provide context for when each visualisation should and should not be utilised.

5.2.1 General Tasks

This group consists of tasks not linked to metadata or roles but still significantly important for analysis. Together, these tasks help analysts understand the infrastructure and its corresponding data. Table 12 and 13 list the nine tasks in this group.

Task	What	Why	How
Label adversaries	**Data Types:** Item	**Actions:** Annotate, Identify **Targets:** Features	Colour, badge, icon
Label victims	**Data Types:** Item	**Actions:** Annotate, Identify **Targets:** Features	Colour, badge, icon
Distinguish adversary infrastructure	**Data Types:** Items, Link	**Actions:** Browse, Summarise **Targets:** Features	Network graph, icon, colour
Distinguish victim elements	**Data Types:** Items, Link	**Actions:** Browse, Summarise **Targets:** Features	Network graph, icon, colour
Distinguish non-important elements	**Data Types:** Items, Link	**Actions:** Browse, Summarise **Targets:** Features	Network graph, icon, colour
Distinguish entity roles	**Data Types:** Items, Attribute **Attribute Type:** Categorical	**Actions:** Browse, Summarise **Targets:** Features	Network graph, icon, colour

Table 12: WWH analysis – General tasks (Part 1)

Task	What	Why	How
View open ports per IP	**Data Types:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Discover, Explore, Summarise **Targets:** Features, Outliers	Table, List
Perform Arkime query	**Data Types:** Items	**Actions:** Lookup **Targets:** Features	Text
View sessions	**Data Types:** Items, Attribute, Link **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Explore, Summarise **Targets:** Features	Table, network graph

Table 13: WWH analysis – General tasks (Part 2)

The first two tasks involve labelling adversaries and victims, using only the **Item** data type to represent potential adversary or victim entities. The analyst's goal is to **Annotate** these entities so they can be easily **Identified** as adversaries or victims based on their **Features**. Such annotations are best achieved through colours, badges, or icons, as these visual cues allow the analyst to quickly distinguish between different entity types at a glance.
Next, analysts must distinguish between four types of elements within the dataset. The first three tasks, concerning adversary, victim, and non-important elements, are identical in their WWH selection. These tasks involve **Items** and **Links**, representing entities and their connections to other entities. The fourth task, distinguishing entity roles, differs in its What selection by involving **Categorical Attributes** rather than links. For all four tasks, the goal is to **Browse** and **Summarise** the infrastructures and elements to gain a clearer understanding of their **Features**. Network graphs are the most suitable visualisation for representing entities and their connections. To clarify distinctions between different types or roles, colours and labels can be applied directly within the graph. Even though entity roles do not include links, they represent the same entities as the other three tasks, so a network graph with colours and labels remains the most effective visualisation.
Analysts also examine open ports per IP address. Here, IP addresses are represented as **Items**, referring to the entities associated with those IPs rather than the IP addresses themselves, while ports are represented as **Categorical Attributes**. The goal is to **Explore** a **Summary** of the open ports to **Discover** anomalies, which appear as outliers in the features of each entity. Because this information is textual and structured, tables or lists provide the clearest representation.
Since analysts frequently use Arkime for analysis, performing queries within the Arkime web app is another common task. These queries are represented as **Items**, and the goal is to **Lookup** information based on the dataset's **Features**. The most effective representation of these queries is text, entered via a text input field.
Finally, analysts examine the complete set of sessions, represented by both **Items** and **Links**, containing various **Attributes**, including **Categorical** and **Ordered Quantitative** data, with a **Sequential** ordering direction. The goal is to **Explore** and **Summarise** the sessions with a focus on all session **Features**. While tables can provide a structured overview of numerous session attributes, network graphs are essential for visualising the relationships between IPs, capturing connections that tables alone cannot show.

5.2.2 Metadata-related Tasks

24 metadata-related tasks were identified during the interviews. To improve readability, we divided them into three groups: overview metrics, frequency/count analysis, and miscellaneous tasks. A WWH analysis is presented for each of these groups.

Overview Metrics. To gain a better understanding of the data, analysts rely on a set of metrics that describe the dataset as a whole, which we refer to as **overview metrics**. These include four tasks: viewing totals for the number of packets, number of sessions, duration, and data size. These metrics serve two main purposes: first, to provide a general overview of the dataset, and second, to evaluate the impact of applying filters to specific attributes. The WWH analysis for these tasks is presented in Table 14

Task	What	Why	How
View total number of packets	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Summarise **Targets:** Dependency, Correlation	Text (ratio)
View total number of sessions	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Summarise **Targets:** Dependency, Correlation	Text (ratio)
View total duration	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Summarise **Targets:** Dependency, Correlation	Text (ratio)
View total data size	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Summarise **Targets:** Dependency, Correlation	Text (ratio)

Table 14: WWH analysis – Overview metrics

All four overview metrics tasks share the same WWH selection. Since these tasks operate on the entire dataset, sessions are treated as the **Items**, and the tasks differ only in the specific **Attributes** of the sessions they measure: packet count, session count, duration, and data size. All four attributes are **Ordered Quantitative** and follow a **Sequential** ordering direction. The goal of these tasks is to **Summarise** these values for the entire dataset to identify potential **Dependencies** or **Correlations** with other aspects of the data. Because each task produces a single aggregated value, the most appropriate way to represent them is using text. To enhance interpretability and highlight dependencies and correlations, both the total value and the filtered value can be displayed side by side as a ratio, optionally with the corresponding percentage.
The task of viewing total data size was included as a custom task because, like the other metrics, it varies significantly when filters are applied. Displaying this value in the overview allows analysts to gain insights that are otherwise difficult to discern from raw data alone.

Frequency Analysis / Count. To detect patterns and anomalies in the data, analysts often rely on frequency- or count-based analyses. These analyses involve counting occurrences and either displaying the results or ordering the data based on frequency. There are eight tasks in this group, each targeting attributes such as IP addresses, MAC addresses, SSH sessions, geolocations, HTTP methods, network protocols, TCP flags, and DNS hosts. The following WWH analysis covers all these frequency/count tasks.

Task	What	Why	How
Frequency analysis / count on IP	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table
Frequency analysis / count on MAC	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table
Frequency analysis / count on SSH sessions for IP	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table
Frequency analysis / count on geolocation	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table, bar chart
Frequency analysis / count on HTTP methods	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table, bar chart
Frequency analysis / count on protocols	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table, bar chart
Frequency analysis / count on TCP flags	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table, bar chart
Frequency analysis / count on DNS hosts	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Locate, Summarise **Targets:** Outliers, Distribution, Extremes	Table, bar chart

Table 15: WWH analysis – Frequency and count-based analysis

All eight tasks share the same What and Why selection. Each task involves **Sessions** as the **Items**, with a focus on a specific **Attribute** such as IP address, MAC address, protocol, geolocation, HTTP method, TCP flag, or DNS host. These attributes are all **Categorical**, and their associated counts are treated as **Ordered Quantitative** data with a **Sequential** ordering direction. The objective is to **Discover** the attribute value **Distribution**, **Locate** both the most and least frequent instances, and **Summarise** frequency patterns to identify **Outliers** and **Extremes**.

While the What and Why selections are identical across tasks, the How selection differs between the first three and the remaining six. Attributes with many unique values, such as IP and MAC addresses, are best represented using tables, which allow sorting and direct comparisons. Attributes with fewer discrete categories, such as protocols, geolocations, HTTP methods, TCP flags, and DNS hosts, benefit from bar charts in addition to tables, providing a more immediate visual sense of distribution.

The DNS hosts task is a custom task that enables analysts to identify unusual and malicious domains and detect potentially compromised hosts deviating from expected behaviour. This task follows the same principles as the other frequency analyses, ensuring consistent interpretation and visualisation.

Miscellaneous. Finally, there is the group of metadata-related tasks that do not fit either of the previous two categories. Tables 16 and 17 provide an overview of the WWH analysis of the 12 tasks in this group.

Task	What	Why	How
Distinguish HTTP from HTTPS	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Derive, Compare **Targets:** Distribution	Pie chart, bar chart
View session durations	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Summarise **Targets:** Outliers, Extremes, Distribution	Bucketed bar chart
View session volumes	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Summarise **Targets:** Outliers, Extremes, Distribution	Line chart
Session distribution over time of day per weekday	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Cyclic	**Actions:** Discover, Derive, Summarise **Targets:** Trends, Outliers, Distribution	Heatmap, polar chart
Identify VPN/TOR traffic	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Summarise **Targets:** Features	Table
Identify sexually explicit traffic	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Summarise **Targets:** Features	Table
Find port usage anomalies	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Discover, Annotate, Explore **Targets:** Features, Outliers	Table

Table 16: WWH Analysis – Miscellaneous (Part 1)

Task	What	Why	How
Find IPs with high destination entropy	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Diverging	**Actions:** Discover, Summarise **Targets:** Outliers, Extremes	Table, bar chart, line chart
Find IPs with low destination entropy	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Diverging	**Actions:** Discover, Summarise **Targets:** Outliers, Extremes	Table, bar chart, line chart
Find high volumes per IP	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Summarise **Targets:** Outliers, Extremes	Table, bar chart, line chart
Distinguish SSH from non-SSH traffic	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Discover, Compare **Targets:** Distribution	Pie chart, bar chart
Find DNS sessions with high volumes	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Summarise **Targets:** Outliers, Extremes	Table, bar chart

Table 17: WWH Analysis – Miscellaneous (Part 2)

The first miscellaneous task involves distinguishing between HTTP and HTTPS traffic. Here, the data consists of sessions (**Item**) and their corresponding protocols (**Attribute**), which are **Categorical**, as there is no inherent order. Although not immediately apparent, this task includes the **Derive** action, as HTTPS is not directly observed but must be derived from the presence of encryption within HTTP traffic. Next, the analyst **Compares** the frequency of HTTP versus HTTPS traffic to analyse its **Distribution**. Pie charts and bar charts are suitable visualisations for comparing categorical data with few unique values, making them appropriate here for the two protocol types.

Next, there are two tasks, viewing session durations and volumes, with identical WWH selections. The **Items** represent sessions, and the **Attributes** represent the durations and volumes, both **Ordered Quantitative** with a **Sequential** ordering direction. The goal is to **Summarise** these values to highlight **Outliers** and **Extremes**, helping the analyst in **Discovering** unusual activity. Line and bar charts both support summarising distributions and identifying outliers and extremes. Line charts are better suited for volume because volumes typically change smoothly between sessions, focusing on changes over time, while bucketed bar charts effectively group session durations, which may vary widely, into meaningful ranges such as 0 - 100 ms or $>$ 10 s. Overall, volumes are best represented continuously, while duration patterns are clearer when segmented.

The next miscellaneous metadata-related task involves analysing the distribution of sessions across different hours of the day, split by weekday. **Items** represent sessions, while **Attributes** represent the weekday and the hour of the day. These are **Ordered Quantitative** attributes, but their ordering direction is **Cyclic** because values repeat after completing a full day or week. This task aims to **Discover Trends** and **Outliers** in attacker behaviour, producing a **Summary** of the session **Distribution**. To enable this, the weekday and hour must first be **Derived** from timestamps. Heatmaps and polar charts are the most suitable visualisations for cyclic and temporal data, making patterns easier to spot.

Next, there are two tasks with identical selections: identifying VPN/TOR and sexually explicit traffic. The data consists of sessions as **Items** and their **Attributes**, specifically IP address and domain, used for identification. For both tasks, the relevant traffic needs to be identified so it can be verified and subsequently filtered out. To support this process, analysts require a **Summary** of the respective **Features**. The only effective way to represent this information clearly is through the use of a table, as this supports individual analysis of each session for filtering.

Another task focuses on detecting anomalies within the transport protocols and services used with

certain ports. Here, **Items** are sessions, and ports, transport protocols, and services are represented by **Categorical Attributes**. Analysts aim to **Discover** anomalies and **Annotate** them whenever they are confirmed. The analysts **Explore** the set of sessions to find **Features** that deviate from the norm and can be considered **Outliers**. Once again, this is best represented by a table to allow each session to be analysed individually and to ensure all necessary attributes are included in the visualisation.
Four custom tasks were included in addition to the previous miscellaneous tasks. The first two involve computing the entropy of destination IPs. Entropy (H) is calculated over the set of destination IPs (W) for a given source IP using the formula for Shannon Entropy as seen in Equation 3 [100]:

$$H(X) = -\sum_{i=1}^{N} p_i \log_2(p_i) \tag{3}$$

Here, N is the total number of destination IP addresses, or the length of W. The probability of a destination IP, p_i, is calculated as shown in Equation 4, where x_i is the number of times a unique destination IP appears in set W:

$$p_i = \frac{x_i}{N} \tag{4}$$

A higher H means that the IP address has many unique destination IPs, while a low H means that its communication is concentrated to a few destination IPs [101]. These entropy values are an **Ordered Quantitative Attribute** of IP addresses (**Item**). As the entropy can indicate anomalies for both significantly low or high values, the ordering direction is **Diverging**. The aim of displaying the highest and lowest entropy values is to **Discover** IPs that have either a spread-out or concentrated communication pattern. These **Outliers** and **Extremes** are then found in a **Summary** of the entropy values. These numerical values are best shown in a sortable table or a sorted bar chart. Line charts can also be used to show entropy variation over time.
The third custom task involves finding IP addresses with high data volumes. It shares the same WWH structure as the entropy task, except that the ordering direction is **Sequential**. A **Summary** that highlights high-volume sessions to **Discover** potential hotspots or suspicious activity is created. Tables and bar charts remain effective, providing both detailed inspection and quick comparison of volume magnitudes.
The fourth custom task is distinguishing SSH from non-SSH traffic. Analysts stressed the importance of analysing SSH traffic, but did not explicitly mention creating a distinction. As this is still likely to be an important feature, this custom task was added. Here, sessions are denoted as **Item** and the protocols as **Categorical Attributes**. This task is performed to **Discover** the **Distribution** of SSH and non-SSH traffic by **Comparing** the two fractions. Pie charts emphasise proportion, while bar charts show absolute counts, making it easier to detect any imbalance or unusual SSH usage patterns.
The final custom task involves identifying DNS sessions with high volumes. Here, the **Items** are DNS sessions, and the **Attribute** is the session volume, which is **Ordered Quantitative** with a **Sequential** ordering direction. The objective is to **Discover** and **Summarise** these values to highlight **Outliers** and **Extremes** that may indicate suspicious or abnormal activity. As with other high-volume detection tasks, tables allow detailed examination of individual sessions, while bar charts provide an overview and quick visual comparison. Together, these visualisations ensure analysts can efficiently detect unusual DNS traffic patterns for further investigation.

5.2.3 Role-related Tasks

The role-related tasks identified in the interviews are divided into four groups based on their respective roles. To ensure clarity of the WWH analysis, we present it separately for each role.

Proxy. Four tasks related to proxy-involved analyses were identified. An overview of these tasks and their corresponding WWH selections is presented in Table 18.

Task	What	Why	How
Detect proxy-related protocols per IP.	**Data Types:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Identify **Targets:** Features	Bar chart, list
Count proxy-related HTTP headers per IP	**Data Types:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Discover, Identify **Targets:** Features, Outliers	Bar chart, list with counts
Check if the input-output byte ratio is near one per IP.	**Data Types:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Diverging	**Actions:** Identify, Derive **Targets:** Extremes	Scatter plot with identity line, table
Compare input and output data volumes over time per IP	**Data Types:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Compare, Locate **Targets:** Correlation, Similarity, Dependency	Line chart, area chart

Table 18: WWH analysis - Proxy-related tasks

The first task involves detecting proxy-related protocols associated with entities, represented by their IP addresses (**Items**). The protocols are considered **Attributes** and are **Categorical**, as they represent named categories without inherent ordering. The analyst aims to **Identify** which protocols are used by specific IP addresses, focusing on these protocols as **Features**. A bar chart is suitable for visualising a small set of categorical attributes because it allows the analyst to quickly see which protocols are most frequently associated with different IPs, highlighting patterns of protocol usage. For larger datasets or when the distribution itself is not the focus, a textual list allows direct inspection of all IP-protocol associations, supporting detailed analysis without requiring aggregation.
The second task is similar, involving counting proxy-related HTTP headers per IP address. In addition to identifying the protocols, the analyst seeks to **Discover** the number of headers to find **Outliers**, specifically entities that use an unusually high number of headers. High counts may indicate that an IP is acting as a proxy. Bar charts can highlight these outliers visually by showing the relative number of headers per IP, while annotated lists present precise counts for each IP, enabling direct inspection of extreme values.
The third task assesses whether the input-output byte ratio for each IP address approaches one. The IPs are treated as **Items**, and the byte ratio is an **Attribute** that is **Ordered Quantitative**. The ordering is **Diverging**, as values may deviate either above or below 1.0. To perform this task, the analyst must first **Derive** the ratio from raw input and output volumes, then **Identify** IPs with ratios close to one, which is an **Extreme** in this context. A scatter plot with an identity line (y=x) allows the analyst to quickly see which IPs deviate from the ideal ratio and in which direction. A table complements this by providing exact numerical ratios, supporting filtering and detailed investigation of individual IPs.
The fourth task involves comparing the volumes of input and output data over time to identify typical correlations of proxy activity. The **Items** remain IP addresses, and the **Attributes** are Unix epoch timestamps and corresponding data volumes, both **Ordered Quantitative** with a **Sequential** ordering. The goal is to **Discover** patterns over time, **Compare** input versus output volumes, and **Locate** specific periods of interest. By analysing these patterns, the analyst can detect **Correlation**, **Similarity**, and **Dependency** between input and output streams, like synchronised spikes or dips in volume that suggest that an IP is acting as a proxy. Line charts clearly show time-related trends and allow direct comparison between input and output streams, revealing correlated patterns. Area charts can improve the visualisation by highlighting the magnitude of differences over time, making periods of unusual activity easier to spot.

C2. This group consists of four tasks focusing on the detection of C2 entities. An overview of these tasks and their WWH analysis is presented in Table 19.

Task	What	Why	How
Inspect traffic payloads for plaintext or decryptable content.	**Data Types:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Lookup, Identify **Targets:** Features	Table, text
Determine if encryption keys were transmitted with the payload.	**Data Types:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Locate, Identify **Targets:** Features	Table, text
Apply FFT to session intervals to identify communication frequency patterns.	**Data Types:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Derive, Locate **Targets:** Outliers	Line chart, spectrogram
Show distribution of frequency counts	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Summarise **Targets:** Distribution, Trends	Line chart, histogram

Table 19: WWH analysis - C2-related tasks

The first task concerns checking whether payloads are decryptable or already in plaintext. Here, the **Items** are individual packets sent by a specific IP address, and the **Attribute** is the packet payload, which is **Categorical** since it does not follow any order. The analyst's goal is to **Identify** the packets with plaintext or decryptable payloads, and since their locations within the payload are known, the analyst also performs a **Lookup**. The targets of this task are the payloads, which are **Features**. Because the analyst is only concerned with the content of each packet rather than trends or summaries, a simple table or text list is sufficient. This allows for detailed inspection of each payload and enables the analyst to confirm plaintext or decryptable data efficiently.

The second task is conceptually similar but differs in that the location of the encryption key is unknown. Therefore, the action is **Locate** rather than lookup. The analyst must **Locate** the relevant content before further analysis. Despite this change in action, the underlying data and goal remain the same, so the choice of visualisation remains a table or text. This ensures that the analyst can systematically review packet content while examining where the relevant information is located.

For the third task, analysts examine the FFT of session intervals, represented by the **Attribute**, which is **Ordered Quantitative** in the **Sequential** direction. The goal is to **Discover** a hidden frequency within the **Derived** intervals. Because the exact location of the frequency within the FFT is unknown, the analyst must also **Locate** it. Frequencies of interest appear as **Outliers** in magnitude after performing the FFT. A line chart effectively visualises these outliers and supports two quantitatively ordered axes (frequency and magnitude), making it easier to spot significant peaks. Additionally, since FFT maps time series data to the frequency domain, a spectrogram is another suitable visualisation, as it displays frequency intensities over time and highlights patterns that may not appear in a static FFT plot.

The final, custom task added for the C2 role is to show the distribution of frequency counts across the dataset to create a general overview. IP addresses act as **Items**, and frequencies act as **Attributes**, which are **Ordered Quantitative** and **Sequential**. The goal is to **Discover** and **Summarise** how many items

occur with each frequency. Once summarised, the analyst can examine the **Distribution** and **Trends** of frequency counts to identify patterns. A line chart or histogram is most appropriate to visualise distributions and trends. Adding frequency counts on the y-axis makes the distribution more explicit and allows the analyst to see which counts are common, rare, or indicative of suspicious behaviour. This combination of visualisation and summary allows for overview-level insights.

Scanner. Among the four role groups, the Scanner role has the highest number of tasks associated with it, as shown in Table 20, which also presents the corresponding WWH analysis.

Task	What	Why	How
Check for SSH connection at start	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Identify, Lookup **Targets:** Features	Table
Count the number of connection attempts to different IPs	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Locate, Summarise **Targets:** Outliers, Extremes	Table
Count the number of connection attempts to different ports	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Locate, Summarise **Targets:** Outliers, Extremes	Table, bar chart
Inspect the traffic payload volume	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Lookup **Targets:** Outliers, Extremes	Table, line chart
Inspect traffic payload content	**Data Type:** Item, Attribute **Attribute Type:** Categorical	**Actions:** Discover, Lookup **Targets:** Features	Table, text
Check the number of failed connections	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Derive, Locate, Compare **Targets:** Outliers, Extremes	Table, pie chart, bar chart
Check if the number of connections fits the baseline	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Derive, Locate, Compare **Targets:** Outliers, Extremes	Baseline chart
Find detection patterns in the number of unique IPs, unique ports, and failed connections	**Data Type:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Summarise **Targets:** Trends	Parallel coordinates plot, bubble chart

Table 20: WWH Analysis of Scanner-Related Tasks

The first task involves checking for the presence of SSH connections. The **Item** here is an individual connection, and the **Attribute** is its protocol type, which is **Categorical**. Since the analyst knows the target SSH session is at the start of an entity's activity, they perform a **Lookup** and then **Identify** whether

SSH is present. The task focuses on **Features** of the connection, namely protocol types. A table column provides a clear textual indication of the presence of a starting SSH session. This is sufficient because the analyst only needs to confirm SSH existence rather than summarise patterns or trends.
The second and third tasks aim to count connection attempts to different IPs and ports, respectively. **Item** represents an attempt, and the **Attribute** is either the target IP or port, which are both **Categorical**, while their counts are **Ordered Quantitative**. The analyst aims to **Summarise** overall activity and **Locate** peaks or irregularities, which might indicate scanning behaviour. **Outliers** or **Extremes** such as sudden bursts of attempts to many IPs or ports are the targets. Tables are effective for showing counts directly, and bar charts work well for port counts, illustrating distribution clearly. For IP counts, a bar chart is less suitable due to the high number of possible values, which would make it unreadable.
The fourth task inspects the volume of traffic payloads. Here, **Item** represents packets, and the **Attribute** is payload size, which is **Ordered Quantitative** in a **Sequential** direction. The analyst performs a **Lookup** to check each volume, and focuses on detecting **Outliers** or **Extremes**, such as sessions with unusually large payloads. A table provides exact values per packet, while line charts can illustrate fluctuations or sudden spikes over time. This combination allows both precise inspection and trend analysis of payload sizes.
The fifth task examines payload content to possibly find scanner indicators. This is a **Categorical Attribute** of a session **Item**, as contents such as CVE numbers or encoded strings do not follow an inherent order. The analyst uses both **Lookup** and **Discover** actions, as they might be searching for known indicators or encounter previously unknown and suspicious content. Since the focus is on the **Features** of the payload, tables or text are most effective for displaying exact payload content details, allowing for careful inspection.
The sixth task concerns failed connections. Each **Item** is a connection attempt, and the **Attribute** is a numeric count, which is **Ordered Quantitative** and **Sequential**. The analyst **Derives** the number of failures per IP, then **Compares** across entities and **Locates** anomalies. The targets are again **Outliers** and **Extremes**, such as unusually high failure rates. Tables are useful for exact counts, while bar or pie charts can visualise comparisons between failed and successful attempts, highlighting differences between entities.
The final interview task in this role group involves comparing connection frequency per unit of time to a baseline. This requires first **Deriving** the connection frequency. As with previous tasks, the data is **Ordered Quantitative** with a **Sequential** nature. The analyst's goal is to **Compare** frequencies with the baseline to **Locate** ones that differ. **Outliers** and **Extremes** are again the key targets. Baseline charts effectively display deviations over time, allowing the analyst to detect anomalies relative to an expected baseline.
Additionally, one custom task was added. As scanner behaviour is largely based on the number of unique IPs connected to, the number of unique ports used, and the number of failed connections, an overview can reveal patterns that were not visible before. Here **Items** are sessions, while the **Attribute** represents the number of unique IPs, unique ports, and failed connections. Since these are counts, they are **Ordered Quantitative** in the **Sequential** direction. Using these visualisations, analysts can **Discover Trends** within a **Summary** of the three attributes. A parallel coordinates plot or bubble chart is well-suited to simultaneously represent multiple sequentially ordered numeric attributes, highlighting patterns and correlations between attributes.

Exfiltration. The final role, Exfiltration, comprises four tasks. Their corresponding WWH analysis is presented in Table 21.

Task	What	Why	How
Count incoming sessions using file transfer protocols.	**Data Types:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Locate, Summarise **Targets:** Outliers, Extremes	Table, grouped bar chart
Count outgoing sessions using file transfer protocols.	**Data Types:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Locate, Summarise **Targets:** Outliers, Extremes	Table, grouped bar chart
Compare incoming data volume to subsequent outgoing data volume.	**Data Types:** Item, Attribute **Attribute Type:** Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Compare, Locate **Targets:** Correlation, Similarity, Dependency	Line chart, histogram
Find IPs with high incoming and low outgoing file transfers of large volume	**Data Type:** Item, Attribute **Attribute Type:** Categorical, Ordered Quantitative **Ordering Direction:** Sequential	**Actions:** Discover, Identify **Targets:** Trends	Parallel coordinates plot, bubble chart

Table 21: WWH analysis of exfiltration-related tasks

The first two tasks involve counting sessions that use file transfer protocols, split by incoming and outgoing direction. Each **Item** is a session, and the **Attributes** include the protocol type (**Categorical**) and the counts (**Ordered Quantitative**). As the counts increase from a lower bound of zero, they follow a **Sequential** ordering. The analyst wants to **Locate** peaks or anomalies, and **Summarise** the protocol usage. The targets are potential **Outliers** or suspicious **Extremes**. Grouped bar charts are well-suited for comparing session counts between multiple protocols, making anomalies easier to spot. Tables can complement this by providing precise numeric totals per IP, enabling more detailed inspection of individual cases.

The third task focuses on comparing incoming data volume with subsequent outgoing sessions to detect potential data exfiltration patterns. The **Items** are sessions, and the **Attributes** (data volume and timestamps) are **Ordered Quantitative** and **Sequential**. The goal is to **Discover** hidden patterns, **Compare** incoming and outgoing data volumes, and **Locate** potential exfiltration sessions. This task targets **Correlation**, **Similarity**, and **Dependency**, for which line charts can effectively display both incoming and outgoing volumes over time, highlighting spikes and relationships. Histograms can also reveal similar distributions and patterns.

To detect exfiltration patterns, analysts examine the number of incoming and outgoing file transfers, particularly those involving high data volume. The **Items** in this task are sessions, while the number of incoming and outgoing file transfers, and volume are included as **Ordered Quantitative Attributes** ordered in the **Sequential** direction. Additionally, the protocol must be checked for file transfer protocols. These are included as **Categorical** attributes. Using a visualisation for this task, analysts **Discover Trends** within the attributes and **Identify** exfiltration. A parallel coordinates plot is ideal for visualising relationships across multiple continuous variables, while a bubble chart offers an alternative that can encode volume and counts simultaneously for more intuitive visual scanning.

5.3 Prototyping

This subsection outlines the development process of the visualisation tool through the three prototyping phases described in Section 4.4: Concept Design, Mockups, and Implementation. Each phase builds upon the previous one, gradually transforming the list of task-visualisation pairs into a functional prototype of the visualisation tool.

5.3.1 Concept Design

The concept design phase focuses on defining the structure and layout of the visualisation tool based on the identified visualisation pairs. The initial design choices regarding page division and visualisation placement establish the foundation for the subsequent mockup and implementation phases.

Before determining the tool's layout and visualisation placement, the task-visualisation pairs must be categorised and distributed across different pages. For clarity, the WWH analysis in Section 5.2 divides the tasks into three groups: general tasks, metadata-related tasks, and role-related tasks. These groups are based on the tasks' analytical purpose, which also provides a logical distribution across pages. Consequently, the corresponding pages are titled **Main Overview**, **Metadata**, and **Roles**.
While the Main Overview page includes only nine visualisations, the Metadata and Roles pages contain twenty and twenty-four visualisations, respectively, making a single page for each impractical. To maintain clarity and usability, these pages are further organised into categorised tabs. These tabs are designed to guide users through the data by grouping visualisations with similar data types or analytical purposes together, helping users navigate the tool and understand its content more easily. All tasks discussed in this subsection, along with their corresponding IDs, can be found in Appendix B.

We begin with the Metadata page, which includes four overview metric tasks identified by the IDs `MT-OMOx`. These tasks examine how filtering affects the dataset. Since filters can be applied in any tab, these tasks must remain visible, regardless of which tab is active. As a result, the four tasks are displayed on the Metadata page, but are positioned outside the tabs.
The remaining metadata-related tasks can be divided into three categories according to their associated data types: addresses, protocols, and sessions. The addresses and protocols tabs include visualisations for seven tasks, whereas the sessions tab contains six. A complete overview of the tasks in each tab is provided in Appendix C.

Role-related tasks are similarly organised into appropriate tabs. This organisation is simpler than that of the metadata-related tasks, since they were previously grouped into four categories based on their corresponding roles (see Section 5.2.3). These role-based categories naturally determine how tasks are distributed across tabs, since they reflect their distinct analytical purposes. As a result, the Roles page contains four tabs: Proxy, C2, Scanner, and Exfiltration. Each tab presents visualisations tailored to the specific analytical objectives associated with that role. Together, the pages and tabs form an interconnected structure in which each element is accessible from any other page. For an overview of the pages and tabs, consult Figure 2.

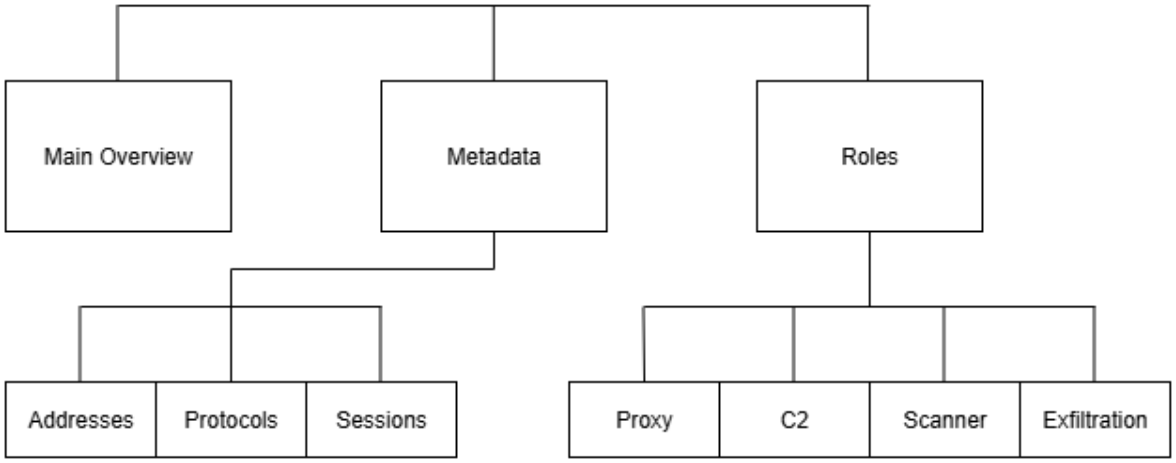

Figure 2: Structure of pages, tabs, and connections

Before finalising each page's layout, a final visualisation must be selected for tasks that are paired with multiple visualisation options. In some cases, integrating all available visualisation options provides additional and useful context that would otherwise be missed. These tasks are not discussed here, since no choice is required between options.

The first task requiring a choice between visualisation options is `GT07`. This task involves examining the open ports of a specified IP. The information can be visualised as either a simple list or a table. While a list offers the most concise representation, a table provides additional context within its columns, such as the expected transfer protocol and associated service. Since this added context is likely to support analysts in their analysis, a table was selected for this task.

Next, a group of six tasks requires selecting between two visualisation options. These tasks are identified by the IDs `MT-FC04` through `MT-MS12` and involve frequency and count analyses of various categorical attributes. One option is a sortable table that includes a column for attribute counts. Alternatively, a sorted bar chart can be used, as these attributes typically have a limited number of unique values, unlike identifiers such as IP or MAC addresses. Since bar charts provide a more intuitive way of comparing attribute values than tables, all six tasks are visualised using bar charts.

Another group with two visualisation options includes tasks `MT-MS01` and `MT-MS11`. These tasks handle the distinction between two types of attributes: HTTP vs HTTPS, and SSH vs non-SSH. This comparison can effectively be visualised using either a pie chart or a bar chart. Since each comparison involves only two values, a pie chart offers a more intuitive visual representation than a bar chart.

Tasks `MT-MS08`, `MT-MS09`, and `MT-MS10` involve identifying extreme and extreme-adjacent values for a specified attribute. These tasks can be visualised using a table, bar chart, or line chart. Line charts provide additional temporal context by showing how values vary over time, highlighting the distribution. However, because the objective is to identify overall extremes rather than trends over time, this additional context is unnecessary. A sortable table with a count column would support the identification of target values, but a bar chart allows for direct visual comparison, offering greater analytical clarity. Therefore, a bar chart was selected as the most effective visualisation.

Task `RT-PR04` involves comparing the input and output data volumes of a single IP address over time. This task can be visualised using either a line chart or an area chart, which both depict changes in values over time using similar line-based representations. Although area charts also emphasise the magnitude of values by shading below the line, this is unnecessary for the current task, where analysts are focused solely on identifying similarities in trends. Therefore, a line chart was selected as the most appropriate visualisation.

Task `RT-C203` identifies frequency patterns within session intervals by applying an FFT. The resulting frequency data can be visualised using either a line chart or a spectrogram. While spectrograms offer clearer distinctions between frequencies over time, they may be less intuitive to interpret for some users.

Consequently, a line chart was selected for its simplicity and broader interpretability.
Although tasks RT-C204 and RT-EX03 serve widely different objectives, they share the same visualisation options. RT-C204 compares two numerical values over time, whereas RT-EX03 identifies all interval frequencies used and their corresponding counts. Both tasks can be visualised using either a line chart or a histogram. However, since neither task benefits from the binning of values, a line chart offers a more effective visual representation than a histogram.
Task RT-SC03 involves counting the number of connection attempts to different ports. This can be visualised using either a table or a bar chart. While a bar chart allows for quick comparisons of port frequencies, a sortable table provides exact values and supports precise identification of the most and least targeted ports. Since exact values are more important than visualising the overall distribution, a table was selected as the more appropriate visualisation.
Task RT-SC04 focuses on inspecting traffic payload volumes. The effective visualisation options are a table and a line chart. Although line charts effectively highlight fluctuations over time, the goal in this task is to examine individual payload volumes and identify outliers and anomalies. A table allows for a direct lookup and precise values. Therefore, a table was selected as the most appropriate visualisation.
Task RT-SC06 involves examining the number of failed connections. This task offers the option of using a table, pie chart, or bar chart. While pie and bar charts support a clear visual comparison between failed and successful connections, a table presents the exact numerical values more clearly. Since the page already contains several visualisations and there is no requirement to display successful connections, a table is the preferred option for its clarity and minimal size.
Finally, tasks RT-SC08 and RT-EX04 both aim to detect patterns across three numerical attributes. These can be visualised using either a parallel coordinates chart or a bubble chart. Task RT-SC08 involves the number of unique IPs, unique ports, and failed connections. Since these attributes are independent of each other, a parallel coordinates chart is most suitable, as it allows the detection of trends within multiple unrelated attributes. RT-EX04 concerns the number of incoming and outgoing file transfers, along with their corresponding volumes. In this case, session volume acts as an attribute of the incoming and outgoing transfer counts. Therefore, a bubble chart is more effective, as it can represent volume using bubble size while positioning reflects the transfer counts.

With a single visualisation assigned to each task previously described, the layout of pages and tabs can now be sketched on paper and digitally represented using wireframes, where abstract shapes represent the individual visual elements. An example of such a wireframe is shown in Figure 3. Each component is depicted using simple boxes, labels, or visualisation placeholders, without any design details. Only the position and identity of the components are indicated. For clarity, the task ID corresponding to the visualised task is included within each placeholder. This provides a clear overview of which components should appear in each section of the page. The full set of wireframes can be found in Appendix D.

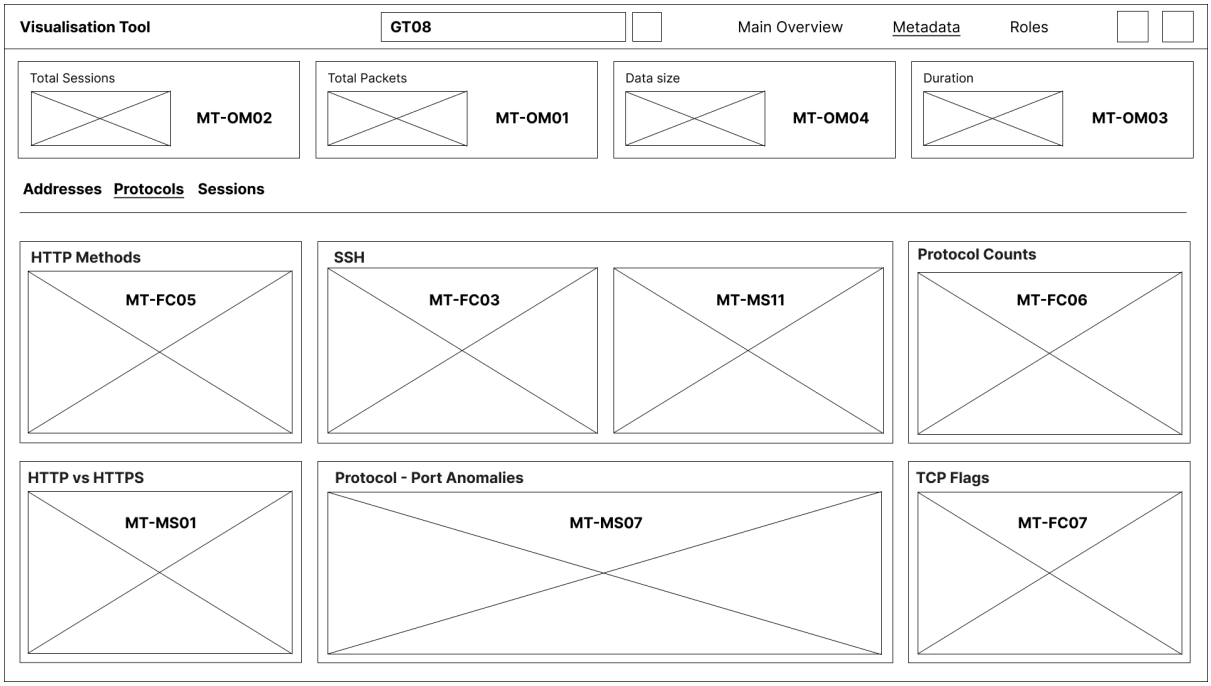

Figure 3: Wireframe example

5.3.2 Mockups

The mockup phase builds upon the sketches and wireframes resulting from the concept design phase. Before the creation of the mockups, a colour-blind-safe palette must be chosen for use within the visualisations. Paul Tol, a Dutch instrument scientist, proposed various such colour palettes for his colour-blind colleagues [102]. He proposes multiple colour palettes, categorised based on the data being either qualitative, diverging, or sequential. Since denoting order with colour is not a feature of most chosen visualisations, the majority will be coloured using the qualitative palette. Only task `MT-MS04` requires a sequential palette.
The qualitative palette that has been chosen is called the *Vibrant* colour scheme. These are bright versions of blue, red, and adjacent colours. Because of red-green colour-blindness, either true red or green can not be included in this scheme. For this context, red was chosen as the more important inclusion over green. Thus, the colour scheme includes teal instead of green. The full colour scheme is found in Figure 4.

Figure 4: Vibrant colour scheme [102]

The sequential palette that was chosen is called the *Iridescent* colour scheme. This palette was chosen because out of three options, it is the most similar to the *Vibrant* colour scheme. The *Iridescent* colour scheme is presented in Figure 5

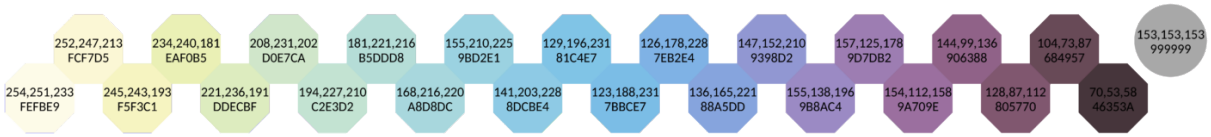

Figure 5: Iridescent colour scheme [102]

Now that a colour scheme has been selected for the visualisations, the mockups can be created. To keep this section concise and clear to read, we discuss only the most representative mockup for each page. The main overview page is discussed in full, while only one representative tab is discussed for both the metadata and roles pages. The remaining mockups, which follow similar reasoning for their design choices, are provided in Appendix E.

Menu Bar. A consistent component across all pages is the menu bar, located at the top of the screen. This bar contains several key functionalities of the tool. Besides a title, it also contains buttons to switch pages, so any page can be accessed from any other page. Furthermore, it contains the query functionality of task `GT08`. The choice of putting this in the menu bar was made so that the query can be viewed and executed from any of the pages. This way, analysts can see their query update from any interactions within the tool. Finally, the menu bar contains two buttons on the far right. The rightmost button refetches the data from Arkime. Since the data from Arkime may change over time while the visualisations assume static data, a refresh button is included to retrieve any newly added sessions. The leftmost button opens a settings dialog. Here, users fill in their Arkime username and password. This is necessary for the tool to communicate with the Arkime API. The placement of these functionalities across all pages ensures consistent access and reduces context-switching for analysts. The mockup of the menu bar is presented in Figure 6.

Figure 6: Menu bar mockup

Main Overview. One of the largest components of the main overview page is the network graph, which is a visualisation option for five out of eight tasks. This visualisation also includes colours, badges, and icons to indicate whether elements represent adversaries, victims, roles, or are unimportant. Specifically, red denotes adversaries, blue indicates victims, orange represents potential roles, and grey is used for neutral or unimportant elements. These colour choices were made based on common associations: red typically signals danger or threats, orange suggests caution or warnings, blue is often used to represent trustworthy or protected entities, and grey implies absence or irrelevance. If a session involves traffic between two differently coloured nodes, each half of the session arrow will have the respective colour of the source and destination nodes. When a role is confirmed, the corresponding node displays icons denoting the first letters of the assigned roles. This way, analysts can recognise all roles they have confirmed without requiring them to examine the attributes.
Given that two nodes can be involved in a large number of sessions, the graph can become cluttered. To prevent this, only one arrow is initially shown per session. Clicking this arrow expands it to reveal multiple individual sessions as separate arrows. Selecting any of these arrows opens a pop-up that provides detailed session information, including the session ID, start and end times, duration, and both the source and destination IP addresses, ports, packet counts, total bytes, and data bytes. A mockup of this session overview is presented in Figure 8.
Clicking on a node opens a different pop-up. This pop-up presents the node's IP address, a colour-coded badge, and a list of confirmed, potential, and absent roles as defined on the roles page. Additionally, it includes a table of the node's open ports, along with the associated expected transfer protocols and services.

To help analysts manage clutter and focus on relevant information, the mockup also includes a collapsible legend with actions in the top right of the network graph. This legend shows users what each colour and letter means, since this might not be immediately clear to new users. Analysts can also filter out unclassified traffic to allow for a focused analysis on more important or suspicious activity. This prevents analysts from becoming distracted by unimportant elements. Additionally, users can adjust the number of entities shown in the network graph, ensuring the visualisation remains usable even for large datasets.
To add to the visual representation in the network graph, a collapsible drawer on the right side of the screen provides a detailed table of all sessions. This table includes important session data such as the start and end times, source IP and port, as well as destination IP and port. It serves as a raw, textual counterpart to the network graph, allowing users to access structured data directly.
For added clarity, summary statistics are displayed above the table, showing the number of identified adversaries, potential roles, and the total number of sessions. This enables users to quickly assess the overall state of the data without having to interpret the full visualisation.

This approach, combining both visual and tabular data, offers support to both exploratory and precise analytical tasks. A mockup of the main overview page is shown in Figure 7, presenting the layout and visual elements described previously.

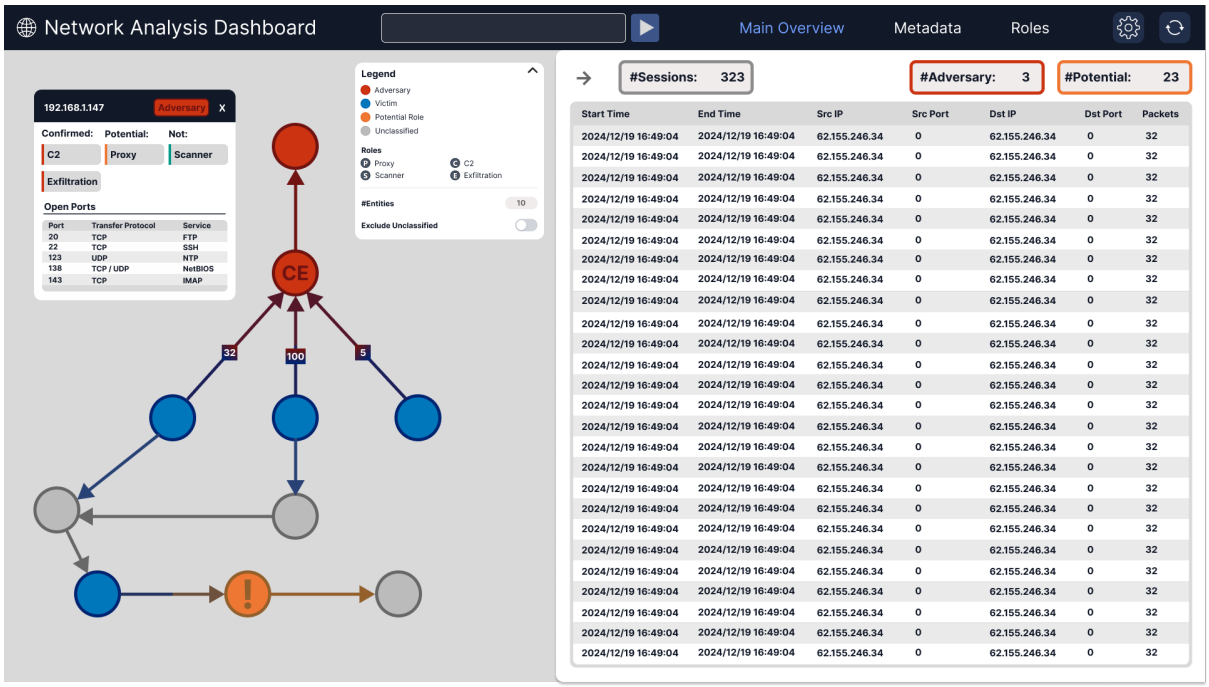

Figure 7: Page Mockup - Main Overview

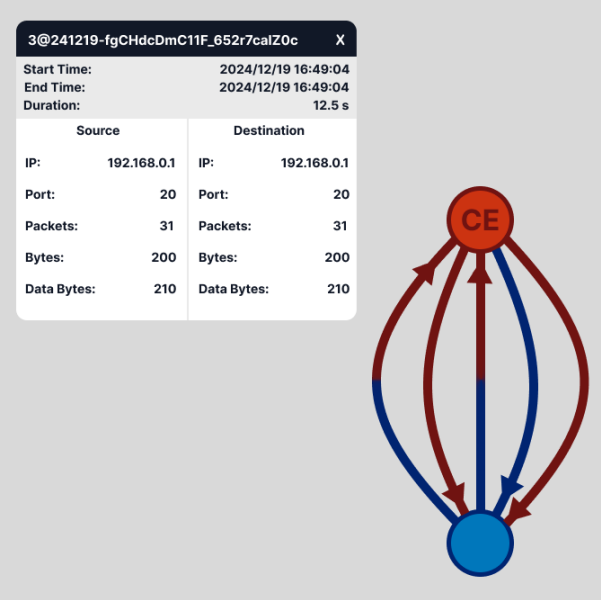

Figure 8: Page Mockup - Detailed session overview

Metadata. As discussed during the concept design phase, four overview metrics are displayed outside the main tab structure. Each metric is presented within its own card positioned at the top of the page. This provides real-time feedback to the analyst whenever interactions are made within a tab. To maintain visual clarity and prevent overflow, the units in the two rightmost cards dynamically update based on the displayed values.
Underneath these overview metrics are three selectable tabs. Selecting a tab changes the set of visualisations displayed below to match the selected category. Each visualisation is contained within a white card, separating it from others and maintaining a clean layout. Visualisations related to the same or a similar data type may be grouped within a single card to highlight their relationship. While most visualisations fit within the bounds of their cards, some require more space. In such cases, the visualisation is made scrollable to keep the layout consistent without sacrificing readability. Each card includes a title in the top-left corner and may also contain various interactive components in the top-right corner.
The primary colour used for visualisations is blue, chosen for its calming visual effect, which makes prolonged analysis more comfortable. Red is used as a secondary colour due to its strong contrast with blue, making important elements such as anomalies or comparisons stand out clearly.

Figure 9 presents the mockup for the protocols tab of the metadata page. This design follows the layout defined within the wireframe shown in Figure 3. The specific visualisation choices were determined during the concept design phase and are supported by the findings in Section 5.2. While many design decisions have already been established, some require additional explanations.
For instance, the visualisation showing the most frequent protocols includes an option to include or exclude transfer protocols. These have a low number of possible values but appear frequently, which might skew the results if not filtered out. By allowing analysts to exclude them, the visualisation is ensured to be useful, regardless of context. Similar action-based customisations appear in other visualisations as well, giving analysts more flexibility and control.
Another such design choice is seen in the protocol-port anomalies visualisation, where anomalies are highlighted in red to immediately draw attention. Only the anomalous columns contain red text, ensuring that the visual emphasis is both targeted and effective.
Furthermore, the orientations of the charts also require deliberation. HTTP methods and protocol counts are shown using horizontal bar charts to accommodate a potentially large number of categories, making them scrollable if needed. Scrolling is only applied if the chart overflows vertically, since horizontal

scrolling is not as intuitive for most people. In contrast, TCP flag data, which is limited to only seven possible values, is visualised using a vertical bar chart, which fits within the card without requiring scrolling.

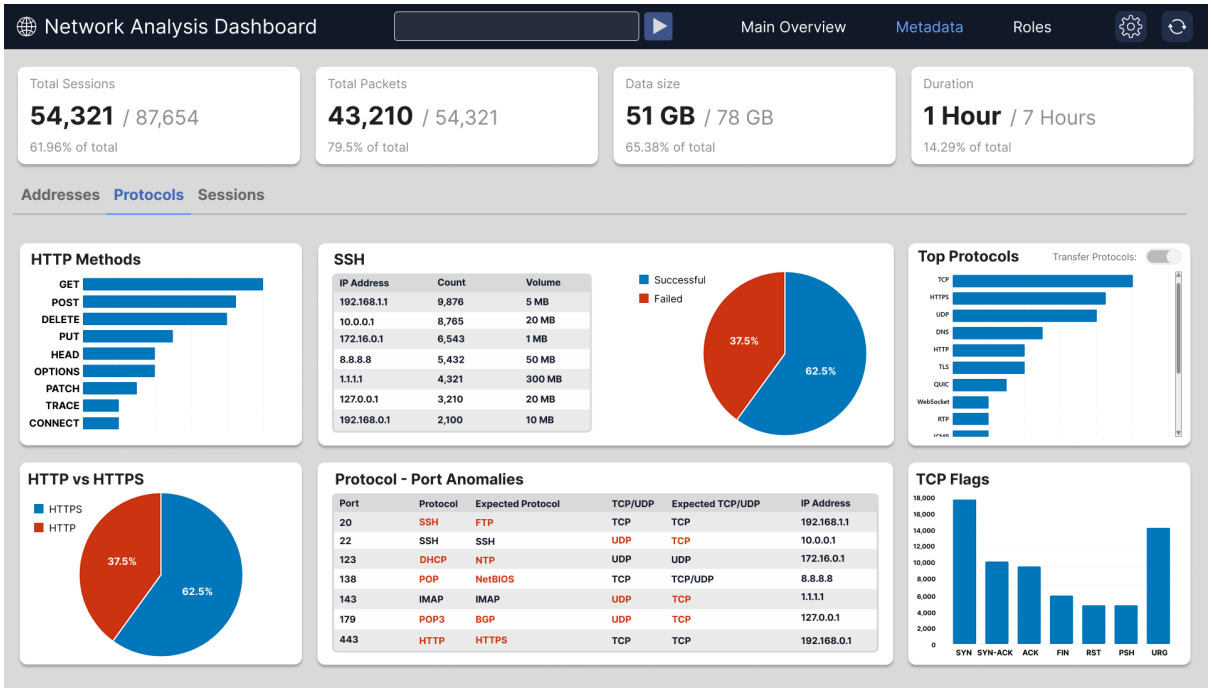

Figure 9: Page Mockup - Metadata

Roles. To support both broad and focused analyses, two separate views were created: one for the full dataset and one for a selected IP address. This dual-view design allows analysts to gain an overall impression of the detected roles in the dataset, as well as to conduct targeted analyses of individual IPs. As a result, the roles page required the highest number of mockups during the design phase.
A consistent element across both views is located in the left half of the screen, which contains a table with each row representing a single IP address. This table includes various attributes and aggregations that help analysts determine whether an IP fits with the role associated with the currently selected tab. Some rows contain a button that opens a pop-up containing more information that does not fit within one column. An example of this is the *header* column of the proxy table. This button opens a list of proxy-related headers and counts. The final column of each row contains a selectable badge that allows analysts to indicate the detection status of the IP. For instance, within the proxy role tab, the available badge options are: *none*, *potential proxy*, *proxy*, and *not proxy*. These statuses are colour-coded to match the colours used in the main overview for visual consistency. Potential roles are automatically detected using predefined rules derived from the interview findings in Section 5.1. When relevant, parameters influencing this automatic detection are displayed above the table, allowing analysts to customise them. The right side of the screen displays either an overview of the complete dataset or visualisations targeted towards an IP, depending on whether an IP address is selected within the table. When no IP is selected, the overview visualisation is shown. This provides analysts with a quick understanding of the role detection status across the entire dataset. For example, in the proxy role tab, a scatter plot is displayed with total incoming bytes on the x-axis and total outgoing bytes on the y-axis. The identity line indicates equal input-output ratios, and proximity to this line suggests potential proxy behaviour. A slider control above the table lets analysts adjust the ratio threshold for detecting potential proxies. Data points in the scatter plot are coloured according to the same scheme as the table badges, simplifying interpretability. Above the visualisation, a summary of the number of confirmed and potential roles is displayed. A mockup of the roles page with the detection overview is shown in Figure 10.

When an IP address is selected from the table, the right panel switches to a detailed view containing two visualisations that help with determining the IP's role. Taking the proxy role as an example, the first visualisation shows input and output bytes over time, helping analysts determine whether they follow similar patterns, which is a potential proxy indicator. Since plotting the full timeline would result in an overwhelming number of data points, the number of sessions displayed is customisable. This approach was preferred over selecting a time window, as analysts often do not know in advance where in time to focus their investigation. An offset parameter is also provided to allow examinations of later portions of the timeline. Below this, a protocol distribution chart shows the relative usage of proxy-relevant protocols for the selected IP, providing further evidence for role classification. A mockup of the roles page with an IP address selected is shown in Figure 11.

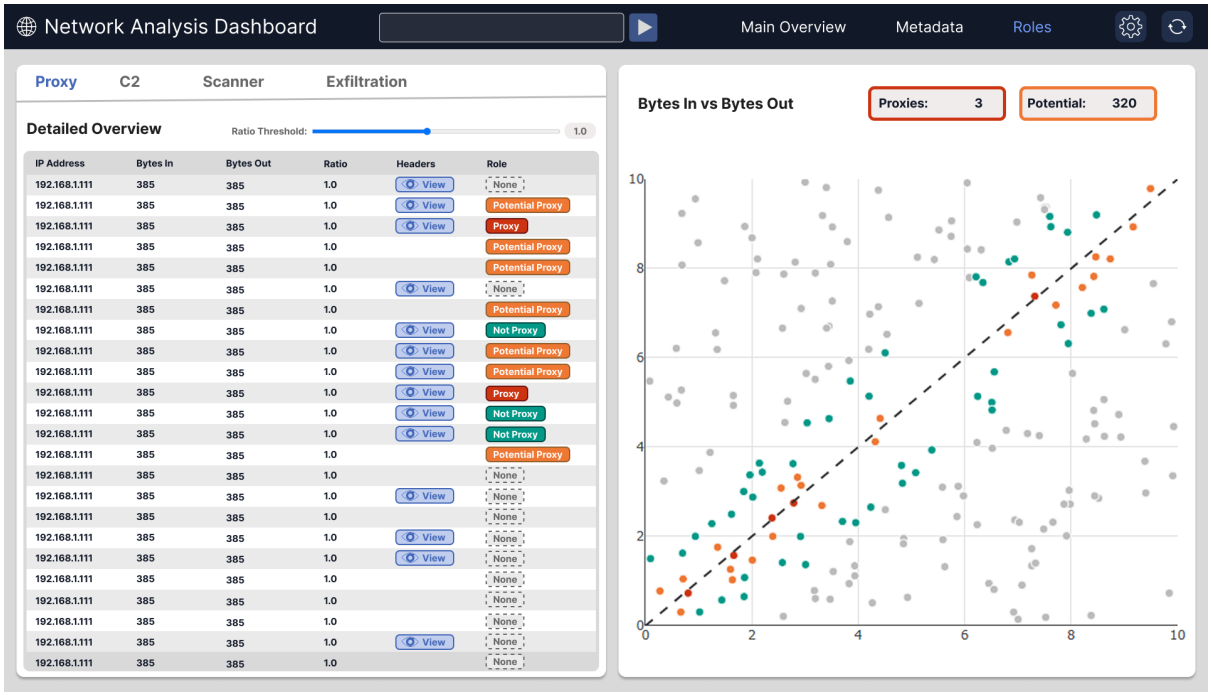

Figure 10: Page Mockup - Roles complete overview

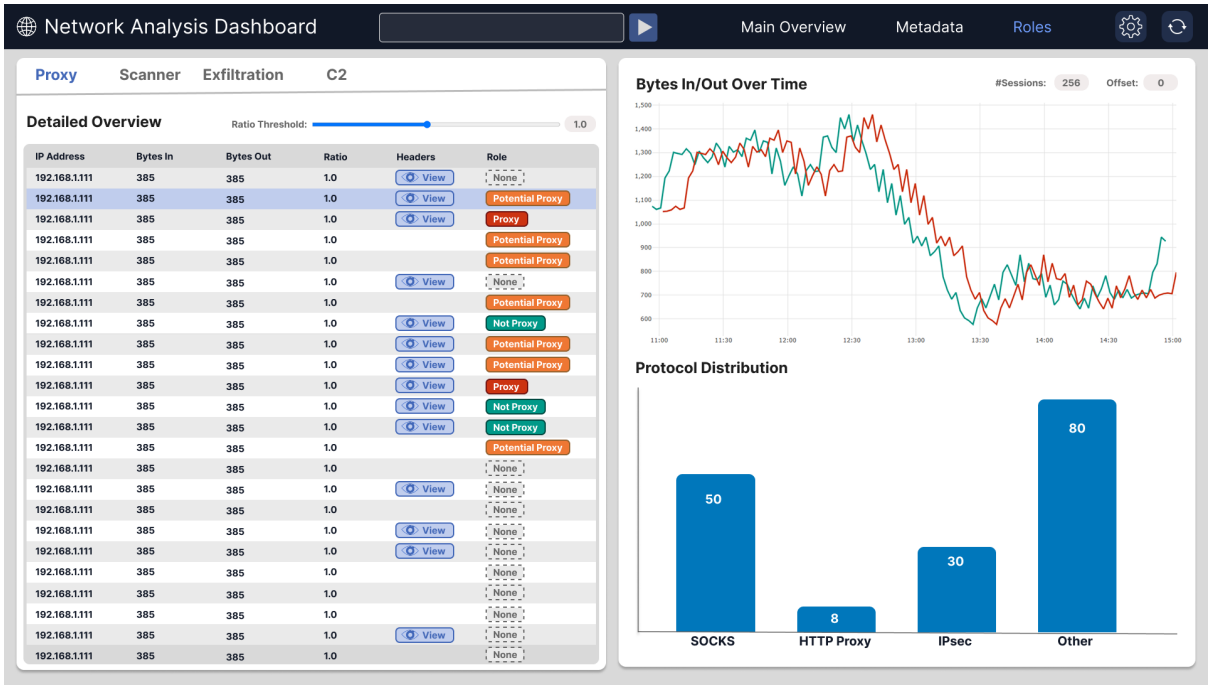

Figure 11: Page Mockup - Roles with selected IP

Feedback. The mockups presented in this report are finalised versions, but they were not completed in a single iteration. After the initial mockups were created, they were reviewed by the consulting analyst to ensure that all visualisations fulfilled their intended purpose. Additionally, custom tasks that were proposed outside of the interview results were evaluated for their usefulness to analysts. To demonstrate this feedback process, one specific example is discussed in detail below.

Figure 12 shows the mockups for the addresses tab of the metadata page, both before and after the analyst's feedback. In the original version, several custom task visualisations were included. However, these were removed after it became clear that they did not provide sufficiently useful insights in the context of this study. In their place, the analyst recommended introducing a frequency analysis focused on traffic volume per IP address, which was deemed more relevant. The freed space was then filled with other custom tasks that offered more useful information.

Moreover, two visualisations related to VPN and TOR usage were relocated from the sessions page to the addresses page. Initially presented at the session level, these visualisations were found to be more relevant when aggregated at the address level.

Finally, during the feedback session, the need for a globally accessible query functionality was raised. This feature was considered important enough to be incorporated across all pages instead of only one.

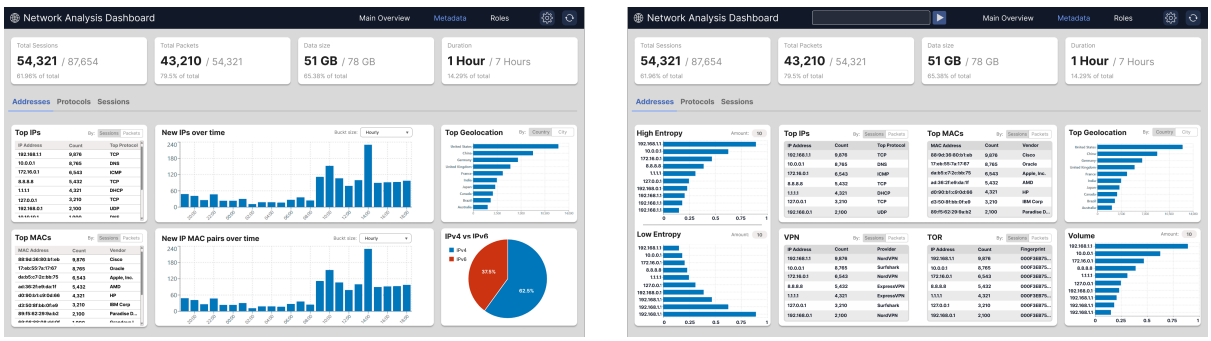

Figure 12: Mockup before feedback (left) and after feedback (right)

5.3.3 Implementation

During the implementation phase, the mockups described in Section 5.3.2 were translated into a functional prototype, bringing a close to the prototyping stage. This functional prototype comprises both frontend and backend code that is connected to Arkime using its API. While the objective was to implement all functionalities described in the previous sections, several components remain unimplemented due to technical and time constraints. In particular, most of the Main Overview page and parts of the Roles page remain unimplemented. In addition to the visualisations, each page and tab also requires the implementation of specific user interactions linked to the visualisations. The final prototype still misses some interactions for implemented pages and tabs. These interactions are absent either due to time constraints or because the Arkime query language does not support certain inputs. An example of this is weekdays. Although there is a heatmap that visualises session counts for weekdays and hours of the day, implementing filtering on these values requires a different approach since Arkime queries do not support weekdays or hours of the day. While theoretically possible, this functionality falls outside the scope of this study and was therefore not implemented. The functional prototype's code can be found on GitHub in its respective repository [103].

The completion percentages for each page and tab are summarised in Table 22, covering both visualisations and their corresponding interactions. These percentages were calculated by counting the number of tasks from the WWH analysis (see Section 5.2) that have been implemented. Each of these tasks has an associated interaction, and by counting the implemented ones, a second percentage is calculated. Interaction progress cannot exceed visualisation progress since each visualisation has exactly one interaction.

Page	Tab	Visualisations Progress	Interactions Progress
Main Overview	**Total**	33%	11%
Metadata	Addresses	100%	100%
	Protocols	100%	100%
	Sessions	100%	100%
	Other	100%	100%
	Total	100%	100%
Roles	Proxy	100%	80%
	C2	100%	70%
	Scanner	0%	0%
	Exfiltration	0%	0%
	Total	50%	38%

Table 22: Pages and tabs with task and interaction progress

Backend. The backend was implemented in Go, with dedicated API endpoints for retrieving the data required by each visualisation. This design allows each visualisation to load independently, thereby reducing initial load times. Data is retrieved from Arkime via POST requests to its API, with filters primarily applied using dynamically constructed Arkime queries. These queries are generated in the frontend and sent to the relevant endpoint, which retrieves only the subset of data matching the specified criteria.
Automatic role detection is implemented for the two available role pages: Proxy and C2. This functionality

relies on predefined rules derived from the interviews described in Section 5.1. Proxy sessions are identified based on their input–output byte ratio. If this ratio falls within a customisable tolerance of 1.0, the session is classified as a Potential Proxy. C2 sessions are identified by applying an FFT to the session intervals and counting the resulting frequency components. If at least one frequency component is detected, the session is classified as a Potential C2

Frontend. The frontend was implemented using Vue.js with Vite to optimise compile times, ApexCharts for visualisations, and PrimeVue components. These technologies collectively reduced development time by a significant margin.

One functionality that was added during implementation, but was not originally included in the mockups, is displayed on the right side of each tab on the roles page when no IP is selected. In the mockups, the design only included counts for confirmed and potential roles. However, it became clear that including a count for entities that had been ruled out for a role was equally important. Without this additional information, the overview of role statuses for the full dataset would fail to provide a clear and complete representation of the data.

Users can also write and execute queries via the query input bar at the top of each page. Queries are then sent to the Arkime API through the backend to filter data. Writing queries manually for session data can be complex and prone to errors. To address this, the frontend dynamically constructs queries based on user interactions with visualisations. For example, clicking a row in the Top IPs table appends the following expression to the query input field: `(ip.src == [IP] || ip.dst == [IP])`, where `[IP]` denotes the selected address. Clicking the same row again removes the expression. This interaction-based approach allows users to build complex queries from predefined formats without manual typing, which reduces the likelihood of human errors.

Three main challenges were encountered in implementing this functionality. First, users still require access to Arkime for features not added to the prototype. To address this, a dedicated button was implemented to execute the current query directly in Arkime, which removes the need to manually copy and paste between tabs. Second, queries can quickly become complex enough to be difficult to read, interpret, and modify. To mitigate this, a structured query overview was introduced. A button in the query input field opens a pop-up displaying the query components in a structured list with the option to remove individual elements. This view also groups complex conditions under simpler labels. For example, the expression `((protocols == http || protocols == http2) && protocols != tls)` is shown simply as HTTP since that is what the query describes. An example of the structured overview is found in Figure 13. Finally, certain Arkime connection parameters, specifically the username, password, and base URL, are currently stored in a `.env` file. For ease of use, these should be incorporated into a settings page as part of future development.

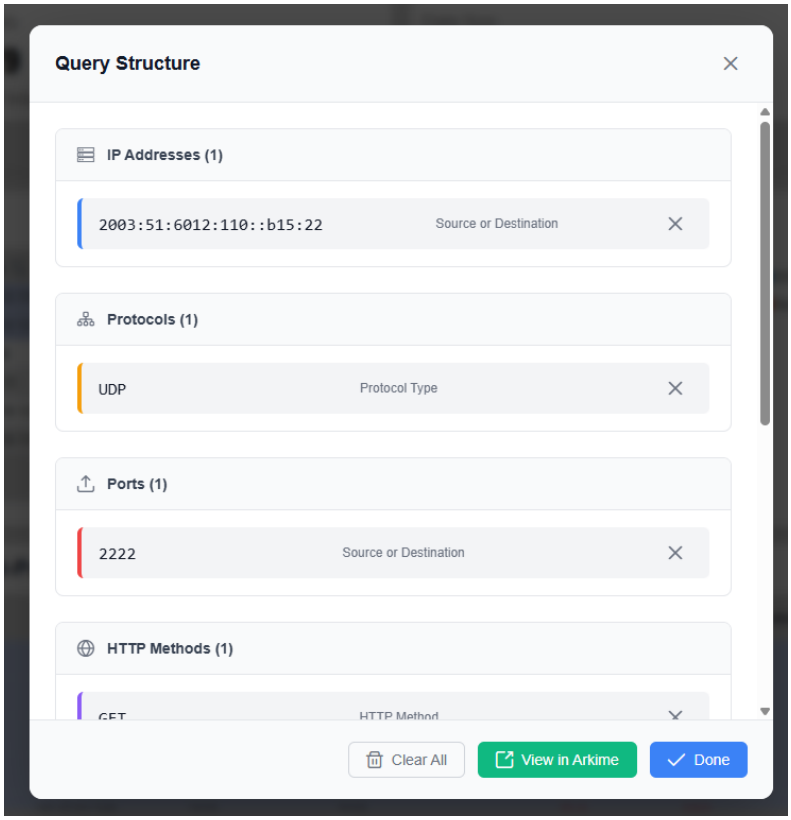

Figure 13: Implemented query structure pop-up

Limitations. During the implementation of the functional prototype, several limitations were encountered. Some resulted from methodological choices, some from unimplemented features, and some were unavoidable within the context of this study. In the following, these unsolved limitations are described along with a potential solution.

The first, and most important limitation encountered during implementation is scaling. As the number of sessions increases, visualisations become slower and may even fail to load entirely. Several mitigation strategies were applied to deal with frontend bottlenecks, such as limiting and customising the number of data points displayed at once. Aggregating data is another approach that reduces the number of points while keeping the visualisation meaningful. For example, session durations are grouped into buckets to reduce the total number of points. While these mitigation strategies have been applied for most implemented visualisations, there are still some that would benefit from them as well.
While Go provides functionalities for backend optimisation, these have not yet been fully implemented. Without these optimisations, the backend can become a bottleneck for larger datasets. Techniques such as using goroutines to execute tasks in parallel and preallocating data structures can significantly improve performance. Choosing the appropriate data types for each task can also reduce execution times and memory usage. These strategies would allow the backend to handle larger datasets more efficiently than it currently does.

Another big limitation, similar to the previous one, is that pages containing multiple visualisations may take a long time to load. The chosen visualisation library, ApexCharts, has a large selection and is highly customisable, but its visualisations slow down with larger data inputs. Furthermore, if one visualisation takes longer to render, it blocks others from loading. This means that if a visualisation with longer loading times is rendered first, all other visualisations also remain unusable during that time. There are two approaches that can mitigate this problem. First, a custom asynchronous loading order can be defined to prioritise faster visualisations, allowing users to interact with them while slower ones finish loading.

Second, batched rendering can be used to load visualisations in groups, giving the browser breathing room between renders. This keeps the frontend responsive while all visualisations are rendered.

Some functionality depends on external APIs, such as IP geolocation services, when one is not provided by Arkime. These APIs introduce their own limitations into the tool and must therefore be used sparingly. Free versions of these APIs often limit the number of requests per timeframe, restricting the use of related visualisations. For example, the API used in the functional prototype has a limit of 45 requests per minute. One solution is to simply use the paid version of the API if available. Alternatively, results can be cached on the first render, allowing unlimited uses within the timeframe. However, since data from the API can change after the first render, this approach may result in outdated data.

Finally, there is one limitation regarding Arkime. The Arkime API allows for a maximum of 2,000,000 sessions to be fetched at once. However, some analyses may involve an amount of sessions that exceeds this limit. A possible solution is to fetch sessions in batches and store them until all data is available. While this introduces overhead due to multiple requests, it allows for complete datasets to be visualised.

5.4 Impact Analysis

This subsection presents the results of the final step in the study's methodology as described in Section 4.5. After completing the functional prototype, an impact analysis was conducted to assess whether the prototype had a positive impact, a negative impact, or no impact on the network forensics analysis workflow. First, the results for the experienced participant are presented, followed by those for the inexperienced participant. Next, the results are interpreted in terms of the tool's effectiveness, intuitiveness, and correctness, with a final evaluation discussion completing this subsection. It is important to note that tasks are denoted by task numbers coinciding with the task numbers found in Section 4.5.5.

5.4.1 Experienced Participant

The experienced participant, who was already familiar with Arkime, completed 10 tasks for each tool separately. The completion times, total time, and number of incorrectly performed tasks were recorded in Table 23. Overall, the prototype allowed for faster task completion, with a total time of 11 minutes and 38.64 seconds compared to 18 minutes and 52.65 seconds for Arkime. This resulted in a total task completion time reduction of 7 minutes and 14.01 seconds, corresponding to a 1.62× improvement. Furthermore, while three tasks were performed incorrectly with Arkime, the prototype resulted in zero mistakes during the analysis.

Task completion times varied between tasks, with six tasks completed faster and four completed slower using the prototype. Usage of the prototype resulted in substantial speed-ups for Task 1 (×24.15 faster), Task 3 (×4.13 faster), and Task 6 (×2.86 faster). Task 2 (×1.41 faster), Task 9 (×1.87 faster), and Task 10 (×1.51 faster) also showed improvements, which, although smaller than the first three, still show a significant improvement. Task 1 achieved the largest improvement, with a time difference of 3 minutes and 48.24 seconds, corresponding to a 24.15× speed-up over Arkime. In contrast, Arkime outperformed the prototype in Task 4 (0.57×), Task 5 (0.35×), Task 7 (0.87×), and Task 8 (0.10×). Task 5 showed the greatest absolute slowdown, with a difference of 2 minutes and 9.02 seconds, while Task 8 had the greatest relative slowdown at 0.10× the completion time of Arkime.

Finally, the analyst made three mistakes out of 10 tasks using Arkime, resulting in a correctness rate of 70%. The prototype allowed the analyst to complete all tasks without any errors, achieving a correctness rate of 100%. This represented a relative improvement in correctness of approximately 43%, calculated as $\frac{1.0-0.7}{0.7} \approx 0.43$.

	Arkime		Prototype		Time Diff (Speed-up)
Task	**Time***	**Mistake**	**Time***	**Mistake**	
1	03:58.10	N	00:09.86	N	-03:48.24 (×24.15)
2	03:06.57	Y	02:12.03	N	-00:54.54 (×1.41)
3	03:55.77	N	00:57.03	N	-02:58.74 (×4.13)
4	00:32.70	Y	00:57.10	N	+00:24.40 (×0.57)
5	01:10.99	N	03:20.01	N	+02:09.02 (×0.35)
6	01:49.08	Y	00:38.16	N	-01:10.92 (×2.86)
7	01:44.79	N	02:00.16	N	+00:15.37 (×0.87)
8	00:09.51	N	01:34.31	N	+01:24.80 (×0.10)
9	00:47.74	N	00:25.53	N	-00:22.21 (×1.87)
10	01:37.40	N	01:04.45	N	-00:32.95 (×1.51)
Total	**18:52.65**	3/10	**11:38.64**	0/10	**-07:14.01** (×1.62)

Table 23: Evaluation results of the experienced participant (with time differences and speed-ups)
*Time format is mm:ss.msms (minutes:seconds.milliseconds).

5.4.2 Inexperienced Participant

For the inexperienced participant, who had no prior experience with either tool, the differences were more pronounced, as shown in Table 24. The total completion time decreased from 25 minutes and 36 seconds when using Arkime to 5 minutes 33.62 seconds when using the prototype, corresponding to a time decrease of 20 minutes and 2.38 seconds, a 4.61× speed-up. Additionally, the number of mistakes dropped from two when using Arkime to zero when using the prototype.

Most tasks were completed substantially faster using the prototype. For example, Task 1 was 27.00× faster, Task 2 showed a 14.47× improvement, and Task 5 was 16.18× faster. Only Task 4 was slightly slower when using the prototype, with a time difference of 20.78 seconds and a 0.83× performance relative to Arkime.

When using Arkime, the participant performed two tasks incorrectly, resulting in a correctness rate of 80%. Using the prototype, however, resulted in zero mistakes and a correctness rate of 100%. This represented a relative improvement in correctness of 25%, calculated as $\frac{1.0-0.8}{0.8} = 0.25$.

	Arkime		Prototype		Time Diff (Speed-up)
Task	**Time***	**Mistake**	**Time***	**Mistake**	
1	03:48.14	N	00:08.46	N	-03:39.68 (×27.00)
2	06:41.28	Y	00:27.61	N	-06:13.67 (×14.47)
3	01:05.98	N	00:30.53	N	-00:35.45 (×2.15)
4	01:42.74	N	02:03.52	N	+00:20.78 (×0.83)
5	02:03.13	N	00:07.62	N	-01:55.51 (×16.18)
6	03:15.70	N	00:18.39	N	-02:57.31 (×10.63)
7	01:47.56	N	00:18.70	N	-01:28.86 (×5.72)
8	00:46.35	N	00:29.19	N	-00:17.16 (×1.59)
9	02:05.80	Y	00:26.79	N	-01:39.01 (×4.70)
10	02:18.82	N	00:22.62	N	-01:56.20 (×6.13)
Total	**25:36.00**	2/10	**05:33.62**	0/10	**-20:02.38** (×4.61)

Table 24: Evaluation results of the inexperienced participant (with time differences and speed-ups)
*Time format is mm:ss.msms (minutes:seconds.milliseconds).

5.4.3 Interpretation

The impact analysis evaluated the prototype's performance using three criteria: effectiveness, intuitiveness, and correctness. The prototype was evaluated based on these three criteria using results from both experienced and inexperienced participants. The following describes how the prototype performed for each criterion.

Effectiveness. The experienced participant's performance demonstrated that the prototype significantly improved effectiveness, as described in Section 4.5.3. The total task completion time decreased by 7 minutes and 14.01 seconds, representing a 1.62× improvement compared to Arkime. Most tasks were completed notably faster, especially Task 1, which was 24.15× quicker. This shows that the prototype streamlines complex workflows for users already familiar with network forensics tools and especially Arkime. Although a few tasks were completed more slowly when using the prototype, the overall reduction in completion time and the majority of tasks being completed faster demonstrate enhanced effectiveness.

Intuitiveness. For the inexperienced participant, the prototype proved highly intuitive compared to Arkime. The total completion time dropped from 25 minutes 36 seconds to 5 minutes 33.62 seconds, which is a 4.61× improvement. Nearly all individual tasks showed significant speed-ups, with several exceeding 10× faster performance than when using Arkime. This performance gain for participants with no prior experience suggests that the prototype's design has a lower learning curve than Arkime and is more intuitive for new users.

Correctness. Both participants achieved perfect correctness rates when using the prototype, completing all tasks without mistakes. In contrast, when using Arkime, the experienced participant made three mistakes (70% correctness rate) and the inexperienced participant made two mistakes (80% correctness rate). The prototype's ability to eliminate mistakes indicated that it improved both the accuracy and reliability of network forensics analysis.

5.4.4 Evaluation

The combined results from both participants demonstrate that the prototype had a positive impact on the network forensics analysis workflow across all three evaluation criteria. Its effectiveness was shown through significant reductions in overall and individual task completion times for the experienced participant, indicating that the tool can enhance productivity for analysts already familiar with Arkime or other tools. The intuitiveness of the prototype was supported by the inexperienced participant's substantial improvements in both total and individual task completion times, suggesting that it lowers the barrier to entry for network forensics analysis and reduces the learning curve for new users. The elimination of mistakes by both participants further indicates that the prototype can increase the accuracy and reliability of network forensics analyses, supporting its correctness.

While some tasks took longer to complete when using the prototype (see Table 23 & 24), the overall performance improvements and error reduction still indicate a net benefit. Observations made during the evaluation suggest that slower task completion times were often due to the participant being unable to quickly locate a necessary component. This suggests that the prototype's navigation structure and the placement of components could be optimised to improve element discoverability and ease of access. Improving these aspects would likely further boost both effectiveness and intuitiveness by enabling users to locate and utilise visualisations more efficiently, reducing frustration and minimising time spent searching for components during complex workflows.

6 Discussion

This section interprets the findings of the study and places them within the broader context of network forensics and data visualisation. The discussion begins with an overview of the main findings, explores their implications, reflects on assumptions, and then addresses the study's limitations and validity. It concludes with directions for future research, ultimately providing a structured overview of the study's contributions and their significance.

6.1 Key Findings

The following key findings correspond to separate steps of the methodology, each providing an answer to one of four sub-questions. Together, they address the main question of this study:

How can network data visualisations be designed to support forensic analysts in overcoming challenges and improving the efficiency of their workflows?

First, the key tasks and challenges identified through the interviews will be discussed, followed by their corresponding visualisations acquired through the WWH framework. Next, the integration of these visualisations into a functional prototype is described, ending with an impact analysis to assess its effect on network forensics analyses

6.1.1 Tasks and Challenges

The first sub-question explores the tasks and challenges network forensics analysts commonly encounter during their investigations. To answer this question, interviews were conducted with experienced analysts, providing detailed insights into the tasks and challenges that benefit most from visualisations. Several recurring themes emerged across the interviews. Analysts highlighted the importance of filtering, working with metadata and aggregations, and obtaining an overview of the infrastructure. They also noted role-related challenges, particularly in detection and follow-up analyses. Although some insights were unique to individual analysts, they highlight important but less common requirements that should be considered in the design of the visualisation tool. Summarising these findings directly answers the sub-question and highlights which analyst needs can be effectively supported through visualisation. The following provides an overview of the tasks identified through the interviews and concludes by addressing the corresponding sub-question. Additional details, including the interview questions and the complete list of tasks, are provided in Appendix A and B.

Recurring Themes. To keep the scope of the study manageable, a focus was put on recurring themes across the interviews, since these represent the most common challenges analysts face. Because these tasks and challenges were mentioned by the majority of analysts, they were considered more important than those mentioned by only a few. This overlap revealed three main categories of tasks.
The first category concerns filtering functionalities which the visualisation tool must provide, since filtering was described as the most frequent action by most analysts. They highlighted the need to filter out sessions such as those from VPNs or TOR nodes, previously analysed traffic, SSH sessions, and other irrelevant or already analysed traffic. Such filtering was named as the most frequent action by the majority of analysts. A related and also frequent action is creating and executing queries in Arkime to retrieve specific subsets of the dataset. These queries allow for filters to be easily applied based on various session attributes.
The second category involves the analysis of infrastructures. The most prominent task in this group is to iteratively reveal adversary infrastructures and to clearly distinguish them from other elements. Once adversary infrastructures are distinguishable, victim identification becomes easier. Analysts must then also clearly distinguish victims from other elements. This process produces a clear overview of malicious

entities and traffic, which supports further analysis. The visualisation tool must therefore include features that support uncovering and distinguishing various types of infrastructures.
Finally, there was an emphasis on tasks related to metadata. Most analysts stressed the need for an overview of various metadata and metadata-related aggregations of the dataset. Currently, analysts must manually locate or compute most of these values. They expressed a need for a clear metadata overview as a starting point for their analysis. This would help analysts identify promising starting points and select relevant filters. Important metadata-related tasks include displaying basic statistics such as total packets, sessions, and duration, frequency analyses of IPs, MAC addresses, geolocations, HTTP methods, protocols, TCP flags, and SSH sessions, distinctions of elements like HTTP vs HTTPS and SSH vs non-SSH, geolocation details like country, city, and coordinates for anomaly detection, session attributes including duration, volume, and time-related patterns, and port anomalies where observed ports do not match expected protocols or services.

Distinct but Significant. Although the focus was put on recurring themes, some analysts mentioned unique but important tasks, challenges, or information. Even if not shared by others, these insights heavily influenced the design of the visualisation tool.
One such task resembles the infrastructure-related tasks discussed earlier. In addition to adversary and victim elements, analysts must also clearly distinguish elements that are confirmed to be neither. This is especially important in large datasets, where unclassified elements can clutter the analysis. Therefore, the tool should allow analysts to distinguish and, if needed, remove unimportant elements.
Analysts also offered practical advice for the design process. One emphasised that visualisations should remain simple and with purpose. Unnecessary complexity in visualisations can obscure rather than clarify data. Furthermore, colour choices within visualisations should both represent information effectively and remain accessible to colour-blind users.
One analyst noted that datasets under analysis are not always static PCAP files and that, in some cases, analysts must work with continuous data streams. Although this had little impact on the design, a choice was made to treat all data as static to keep the goals feasible within the given timeframe.
Finally, one analyst suggested that we think of custom tasks and aggregations that might benefit from visual support. This outside perspective could reveal tasks or challenges that analysts themselves might overlook, and could therefore reveal valuable insights.

Roles. Network forensics analysts often assign specific roles to entities during an analysis. Each role has distinct characteristics and requires specific detection and analysis approaches. Understanding these roles helps guide visualisation design, ensuring that patterns, anomalies, and relationships are visualised clearly. The majority of analysts highlighted four roles as most important to their analysis: Proxy, C2, Scanner, and Exfiltration.

Proxy

- **Definition:** A proxy is an intermediary server that relays and potentially manipulates network traffic between two endpoints.
- **Detection:** Analysts examine proxy-related headers (e.g., `X-Forwarded-For`, `Via`, `Forwarded`) to detect proxies. They also compare incoming and outgoing traffic volumes over time, where a close correlation suggests proxy activity.
- **Analysis:** Since proxies obscure the original source and destination IP addresses, analysts must correlate input and output traffic peaks to uncover hidden traffic flows. Detecting volume mismatches or anomalies in proxy behaviour may further reveal malicious activity.

C2

- **Definition:** A C2 server is an adversary-controlled entity used to send instructions to malware-infected systems.

- **Detection:** Analysts locate plaintext or decryptable payloads to identify C2 activity. They also apply FFT analysis to session intervals, allowing for the detection of beaconing.
- **Analysis:** Identifying victims through beacon sources provides insight into compromised systems, while correlating beacon timing and payloads to victim behaviour can reveal commands. Since proxies can obscure direct relationships with a C2 server, analysts must take this into account when reconstructing traffic flows.

Scanner

- **Definition:** A scanner is an entity that probes networks to discover active hosts and services, often to find weaknesses to exploit.
- **Detection:** Scanning activity is detected by identifying entities that initiate connections to a large number of IP addresses and ports. In some cases, payloads explicitly indicate vulnerability scanning behaviour, occasionally including CVE numbers.
- **Analysis:** As scanning behaviour is typically unimportant, scanners are often excluded from analysis unless their behaviour deviates from the known baseline. Excluding scanners reduces clutter in visualisations and improves the clarity of adversary infrastructure representations.

Exfiltration

- **Definition:** An exfiltration server is an entity that receives stolen data from compromised systems and sends it to the attacker.
- **Detection:** Exfiltration is identified by observing unusually high volumes of incoming sessions, commonly over file transfer protocols such as FTP or SCP. Analysts also examine outgoing traffic to find sessions with even larger volumes that happen less frequently than the incoming traffic.
- **Analysis:** Investigating the sources of incoming traffic helps identify compromised entities, while analysing traffic volumes indicates when and how data exfiltration occurs. Outgoing traffic may provide clues about the attacker, but this cannot be assumed with certainty, as proxies may obscure the true destination.

Sub-question Summary. The interview analysis provides a clear answer to the sub-question concerning the key tasks and challenges encountered by network forensics analysts. Recurring tasks include filtering sessions, constructing queries, uncovering and distinguishing adversary and victim infrastructures, and aggregating metadata to support investigations. Additionally, analysts reported challenges such as distinguishing unimportant elements and handling continuous or non-static data streams. Role-specific analyses further highlighted tasks related to the detection and interpretation of proxies, C2 servers, scanners, and exfiltration entities. Together, these findings define the requirements that a network forensics visualisation tool must address and directly reflect the practical tasks and challenges analysts face during analyses.

6.1.2 Suitable Visual Representations

The second sub-question aims to link the tasks identified during the interviews to their most suitable visualisation strategies. To answer this question, the WWH framework outlined by Tamara Munzner [66] was utilised. This framework describes each task using predefined categories for both the involved data, the objectives of the task, and points of interest. These abstract descriptions of the tasks can then be used as a foundation for picking the most suitable visualisations. By applying the framework to all tasks, an answer is provided to the second sub-question. During this analysis, three groups of tasks were identified: general tasks, metadata-related tasks, and role-related tasks. An overview of all tasks provided in Appendix B.

General Tasks. One group of tasks was given the name General Tasks due to their generic nature. These tasks involve the general analysis process and do not serve a purpose as specific as the other groups. However, this does not make them less important to the design process. Especially due to their generic nature, linking these tasks to suitable visualisations is essential to the creation of a visualisation tool that can support a general network forensics analysis.
The first two tasks in the group concern labelling adversaries and victims. Due to their identical What and Why selection, their most fitting visualisations are also the same. From these tasks, the result is that labelling such elements is best done through the use of colour, badges, or icons.
Next, there are four tasks related to distinguishing various elements: adversary infrastructure, victim elements, non-important elements, and entity roles. The first three again have identical What and Why selections, while the last also includes a categorical attribute representing the role. Still, all four tasks can be best visualised using a network graph with colours and icons to make distinguishing easier.
Two tasks concern more detailed views of attributes. One concerns viewing open ports for each IP. This is best visualised using a table or textual list. The second involves viewing sessions and their corresponding attributes in detail. These can be visualised using a table to make attributes easier to examine, but since sessions represent links, they can also effectively be visualised inside a network graph as edges.
Finally, one task involves executing queries, which is best visualised using text. More specifically, since the complete query needs to be clearly visible and editable, a text input field is the best visualisation.

Metadata-related Tasks. The next group of tasks involves various types of metadata and aggregations of metadata. These tasks result from the analysts' wishes to have a starting point for their analysis that gives an overview of various metadata visualisations. This group can be further divided into three sub-groups concerning overview metrics, frequency and count analyses, and the remainder of tasks called miscellaneous.
The first sub-group contains tasks that concern summary statistics of the dataset. More specifically, to view the number of packets, the number of sessions, the duration, and the data size. These tasks all have the same What and Why selection, resulting in the same visualisation. These tasks are best visualised using simple text, but also benefit from being represented as a ratio of filtered vs total for extra context.
The next sub-group contains tasks that involve the analyst performing a frequency or count analysis on a specific attribute. This group involves eight tasks with identical What and Why selections, but their suitable visualisations differ. While all can be effectively visualised using a table, only five can also be visualised using a bar chart. The reason for this is that these tasks involve attributes that either have a small set of possible values or that are less likely to have many unique values in one dataset. This is not the case for IP and MAC addresses, which will likely have many unique values, and can thus only be visualised well with a table.
Miscellaneous metadata tasks exhibit more diverse objectives that do not fit the previous two groups, including identifying IPs with high or low destination entropies, examining session distributions over hours of day per weekdays, identifying specific traffic types like from VPNs, from TOR nodes, sexually explicit, or SSH, distinguishing sessions based on their protocol, and volume related analyses. Tasks involving ordered quantitative attributes often benefit from line charts or bar charts, with similar tasks involving cyclic data being most effectively represented via heatmaps or polar charts. Categorical attributes are best summarised with tables or pie charts to support direct comparison. Notably, the majority of tasks can also effectively be visualised using tables as an alternative or addition.

Role-related Tasks. Role-related tasks are the result of the interview question asking about roles assigned during analysis. These tasks have the purpose of detecting specific roles and guiding further analysis. Since role assignment is an important part of the analysis, supporting these tasks is essential for the design of the tool. The roles identified during the interviews are Proxy, C2, Scanner, and Exfiltration. Proxy-related tasks focus on detecting proxy activity per IP through protocols, HTTP headers, input-output ratios, and data volumes over time. Categorical data, such as protocols and headers, are best visualised with bar charts or lists to highlight patterns and outliers. Quantitative attributes, like byte ratios

or time-related data volumes, are effectively visualised using scatter plots with identity lines, line charts, or area charts, supporting the identification of patterns, correlations, and trends.
C2 tasks focus on detecting malicious activity through packet payloads, encryption key presence, session frequency, and frequency distributions. Categorical attributes, like payload content or encryption keys, are best examined using tables or text lists for detailed inspection. Ordered quantitative attributes, such as FFT of session intervals or frequency counts, are effectively visualised with line charts, histograms, or spectrograms for the first, supporting identification of outliers, trends, and patterns over time.
Scanner tasks involve examining connection attempts, traffic payloads, and failed connections to identify scanning behaviour. Categorical attributes, such as protocol type or payload content, are best inspected using tables or text lists for detailed analysis per item. Ordered quantitative attributes, including connection counts, payload volumes, and frequency relative to baselines, are visualised with tables, bar charts, line charts, pie charts, or baseline charts to detect outliers, extremes, and deviations. The custom task summarises multiple metrics regarding unique IPs, ports, and failed connections using parallel coordinates or bubble charts, allowing analysts to discover trends and correlations throughout the complete dataset efficiently.
Exfiltration tasks focus on detecting anomalous file transfers by monitoring incoming and outgoing file transfer sessions. Categorical attributes, such as protocol type, are best inspected using tables or grouped bar charts to locate outliers and summarise protocol usage. Ordered quantitative attributes, including session counts and data volumes, are effectively visualised with line charts or histograms to highlight outliers, correlations, and trends over time. For the analysis of multiple metrics, such as IPs with high incoming but low outgoing transfers, parallel coordinates plots or bubble charts allow analysts to discover trends and identify suspicious activity efficiently.

Sub-question Summary. The WWH analysis addresses the sub-question by identifying which types of visual representations best support different tasks and challenges in network forensics analysis. It describes that general tasks, such as labelling entities, distinguishing elements, examining detailed attributes, and executing queries, are supported by techniques that clarify complex data and enable quick interaction and distinction. Metadata-related tasks, including overview metrics, frequency, and miscellaneous analyses, are aided by visualisations that support pattern recognition, analysis of trends, and comparisons. Role-related tasks, covering proxies, C2 activity, scanners, and exfiltration, require representations that help detect anomalies and guide deeper analysis. Together, these insights define the visualisation requirements for a network forensics tool, ensuring it effectively supports analysts' tasks and addresses common challenges.

6.1.3 Integration into Prototype

The third sub-question is aimed at integrating the identified task-visualisation pairs into the design of a prototype tool. The answer to this sub-question is acquired through the development of a functional prototype in three phases: concept design, mockups, and implementation. Through this process, task-visualisation pairs are first transformed into rough sketches, mockups, and finally a functional prototype. To ensure the usefulness of the tool, this process is performed iteratively based on feedback given by the consulting analyst. This way, the tool can best achieve its goal of supporting network forensics analysis.

Concept Design. During the concept design phase, visualisations were grouped into three pages: Main Overview, Metadata, and Roles. These pages contain visualisations according to their corresponding objectives. The Main overview page only contains eight general tasks, which fit on one page. However, the Metadata and Roles pages contain 20 and 24 visualisations respectively, which are too many to be effectively displayed within one page. Therefore, these pages must further be divided into tabs. The Metadata page contains visualisations representing overview metrics, which must always be visible when the page is open to have the desired effect. Therefore, these visualisations will be visible above the tabs, not changing when the tabs change. The remaining visualisations are divided according to the data

types they deal with. More specifically, they are divided into three groups: Addresses, Protocols, and Sessions. A detailed overview of the tasks visualised in each tab is provided in Appendix C. The Roles page has a clearer and intuitive division into tabs. Since there are four roles, with each containing its own visualisations, these can also be used as tabs. Therefore, this page is divided into four tabs, with one for each role.
Once the page division was made, sketches were created to define the layout of each page. These sketches contain abstract representations of visualisations and components to define their positioning and the overall layout of each page and task. These abstract components contain task IDs to denote which task is visualised in which component. This provides a clear structure of the visualisation tool that makes the following two phases of the prototyping process easier. These sketches are then turned into wireframes so they also have a digital representation that can easily be shared. The wireframes are found in Appendix D.

Mockups. Before creating the mockups based on the sketches, a colour palette must be chosen that will best help the visualisations represent their respective data. For accessibility purposes, this must be a colour-blind-safe palette. Two of Paul Tol's [102] colour palettes were picked to represent both qualitative and sequential data. These palettes are called the Vibrant and Iridescent palettes, with the latter only used for task `MT-MS04` since it requires ordered data. From the Vibrant palette, blue is chosen as the primary colour, and red as the secondary for contrast.
Using these colour palettes, the mockups were created. These mockups can be found in Appendix E. First, the Main Overview page received a mockup showing a large network graph with various colours representing adversary, victim, and role-related elements. By clicking on various network graph components, a pop-up is opened containing more detailed information about the attributes of the clicked element, like open ports for IP addresses or timestamps for sessions. Besides the network graph, this page also contains a collapsible side drawer with a table. This table displays all, possibly filtered, sessions and their most important attributes. This way, there is both a visualised and textual overview of the dataset, allowing for both exploratory and precise analyses. Above the table, summary statistics are shown to provide a quick overview of the state of the analysis regarding adversaries and roles.
Next, the mockups for the Metadata page tabs were created. These tabs all have the dynamically updated overview metrics visible above the tabs as a fraction with a percentage representation below. This allows for an immediate and quick overview of the filtering status. Each tab has one corresponding mockup consisting of various visualisations contained within a card to allow for a clean and structured look. Furthermore, visualisations with similar or identical data types are grouped within one card to make their connection even clearer. These cards also contain a title to identify the visualisations, along with, potentially, components in the top right called actions, that can change how a visualisation functions. These actions allow for more customisability by the users, making visualisations more likely to be useful.
Finally, the roles page consists of two main components that change dynamically based on the role and IP selection status. Left, there is always a table containing IP addresses with columns relevant to the detection and analysis of the specific selected role. This table also contains badges to denote the detection status of that IP for the selected role. This badge is manually updated during analysis. To further help in detecting roles, potential roles are automatically detected based on various metrics. Some of these metrics can be slightly customised, like the threshold tolerance for proxy detection. The right initially contains a visualisation depicting the detection metrics throughout the complete or filtered dataset, along with counts for potential roles, confirmed roles, and ruled-out roles above it. If an IP is selected within the left table, the right component changes to show two visualisations useful for deeper analysis of a specific IP. These visualisations aid analysts in deciding if that IP has the selected role. All elements on this page are colour-coded in the same way that elements in the Main Overview are colour-coded. This creates a more consistent and intuitive design.
To enhance usability, all pages must be interconnected so that users can move between them seamlessly. Therefore, a menu bar containing tabs for each page has been added, so any page can be accessed from any other page. This menu bar also contains a text input field to facilitate query creation and execution on any page. While this was not one of the original design choices, feedback from the consulting analyst

showed that this functionality must be available during any analysis step.
Overall, using the sketches and wireframes, mockups were created to be analyst-friendly by following three design principles. First, the mockups should be accessible for many users through the use of colour-blind palettes and clear visual indicators for information. Next, consistency is ensured through a shared menu bar, uniform and static layouts for all pages, and overlapping colour choices per page. Finally, through feedback with the analyst, usability is ensured since impractical or useless elements are removed in favour of more useful ones.

Implementation. The final phase of the prototyping process is the implementation of a functional prototype based on the designs outlined by the mockups. This prototype consists of three main components: a frontend that users interact with, a backend that forms the link between the frontend and Arkime and performs the underlying calculations and aggregations, and the Arkime API that can be used to fetch session data from Arkime. While not all components and pages of the functional prototype ended up being completed, enough have been implemented to allow for answering the sub-question.
The frontend design has some small changes when compared to the mockups. These changes concern the query functionality seen in the menu bar. First, during implementation, the discovery was made that queries can quickly become too large and complex to still be easy to interpret. Having to look deeply into a query to understand what it does would introduce an overhead into the analysis. Therefore, a structured overview of the query was added to allow for quick understanding of a query's functionality. This query can then be executed directly in Arkime using a new browser tab, minimising context switching through manual changes in tabs. This way, it still feels like the natural analysis flow. Besides this functionality, a need for a place to enter required Arkime-specific information was discovered. Information such as the user's Arkime password, username, and base URL is necessary for the tool to function as intended, but is currently provided using a `.env` file.
The backend is designed as a server with one endpoint for each visualisation. This way, visualisations are loaded independently, reducing the time before visualisations can be used. This backend also requests data from the Arkime API and subsequently sends this data to the frontend to be visualised. Using the queries generated in the frontend, this expression can be passed to the API to function identically to the queries in the Arkime web-app. This way, data can easily be filtered before passing it to the frontend. Automatic role detection is also handled in the backend based on various metrics like input-output byte ratio for proxies.

Sub-question Summary. By following the three prototyping phases, an answer to the third sub-question is acquired. The functional prototype created through these steps shows how the task-visualisation pairs can be transformed into a prototype tool. This prototype contains three pages with various tabs containing the necessary visualisations and is made to be useful for as many analysts as possible for real-world analyses. By following these steps, any group of task-visualisation pairs can be transformed into a prototype tool, providing an answer to the third sub-question.

6.1.4 Impact on Analysis

The final sub-question evaluates the impact of the functional prototype on the network forensics analysis workflow by making two types of participants perform a set of tasks while being observed and timed. The individual task completion times, total time, and number of tasks with mistakes tracked, to give an answer according to three criteria: effectiveness, intuitiveness, and correctness. These metrics allow for both the overall impact on the analysis and individual impact on tasks to be assessed. Together, they provide a comprehensive view of the prototype's practical value in supporting forensic analysts.

Effectiveness. For the experienced participant, the total task completion time decreased by 7 minutes and 14.01 seconds, corresponding to a 1.62× speed-up compared to Arkime. Six tasks were completed faster using the prototype, while four tasks were slower, indicating areas for potential optimisation. Task 1

showed the largest improvement, with a 24.15× speed-up, and Tasks 3 and 6 also demonstrated significant improvements (×4.13 and ×2.86, respectively). Although some tasks were slower, the majority of tasks were completed faster, and the total time was significantly reduced, demonstrating that the prototype is more effective overall than Arkime.

Intuitiveness. For the inexperienced participant, the total completion time decreased by 20 minutes and 2.38 seconds, corresponding to a 4.61× speed-up. Only one task (Task 4) was slower using the prototype, while all other tasks showed improvements. Several tasks demonstrated speed-ups of more than 10×, including Task 1 (×27.00), Task 2 (×14.47), Task 5 (×16.18), and Task 6 (×10.63). These improvements suggest that the prototype reduces the learning curve for new users and provides a more intuitive workflow compared to Arkime.

Correctness. Both participants achieved 100% correctness when using the prototype, completing all tasks without mistakes. In contrast, the experienced participant made three mistakes with Arkime (70% correctness), and the inexperienced participant made two mistakes (80% correctness). This corresponds to a relative improvement of approximately 43% and 25%, respectively. The elimination of errors demonstrates that the prototype helps users avoid mistakes during analysis and increases the reliability and accuracy of their results, proving its correctness.

Overall Impact. The prototype demonstrated a positive impact on the network forensics analysis workflow across all three evaluation criteria: effectiveness, intuitiveness, and correctness. For both experienced and inexperienced participants, individual task completion times were mostly reduced, total completion time was substantially reduced, and all mistakes were eliminated. While a few tasks were completed more slowly with the prototype, these cases were limited and often related to difficulties in quickly finding necessary components. This suggests that with further optimisation of navigation and component placement, the prototype's benefits could be even greater. Overall, the results indicate that the prototype improves productivity, lowers the learning curve, and enhances the reliability of network forensics analysis.

Task-Specific Observations. The task-specific results highlight how the prototype's design particularly benefits certain tasks. For example, Task 1 showed dramatic improvements for both participants, with ×24.15 for the experienced participant and ×27.00 for the inexperienced participant, indicating that the prototype streamlines an otherwise time-consuming process. Similarly, Tasks 2, 5, and 6 showed significant improvements, suggesting strong support for similar tasks. In contrast, Tasks 4, 5, 7, and 8 for the experienced participant, and Task 4 for the inexperienced participant, were slower with the prototype. These slowdowns appear linked to challenges in navigation and the discoverability of interface elements rather than to functional shortcomings. Further improvement might show that these tasks can also be supported through visualisations.

Sub-question Summary. The evaluation analysis provides a clear answer to the sub-question concerning the impact of the functional prototype on the network forensics analysis workflow. The prototype was shown to improve effectiveness by reducing overall and most individual task completion times, intuitiveness by enabling inexperienced users to complete tasks with significant speed-ups, and correctness by eliminating mistakes entirely. Challenges identified were mainly linked to navigation and discoverability rather than functional shortcomings, suggesting that further optimisation could result in even greater benefits. Collectively, these findings demonstrate that the prototype enhances efficiency, lowers the learning curve, and increases reliability, thereby directly supporting analysts in their investigative processes.

6.2 Implications

Beyond the specific results presented in this study, it is important to consider what these findings mean in a broader context. Research does not give a complete picture on its own, and the value of any study lies both in its direct outcomes and in how it informs future practice and theory. By reflecting on the implications, we can better understand how the developed tool and the insights gained contribute to the wider field of network forensics and related domains. These implications affect both the practical and theoretical domains, each of which should be taken into account when interpreting this study's significance.

6.2.1 Practical Implications

The practical implications of this study go further than immediate results, highlighting how the findings and the developed tool can be utilised in various real-world contexts. These implications show the impact of the study's findings on not only network forensics analysis but also on other related fields and practices.

The functional prototype proved to be much more intuitive than Arkime, which can often be difficult for new analysts to completely understand [104]. The new tool greatly reduced this learning curve. As a result, the training process for network forensics analysts can be shortened because they can grasp its features more easily. With this support, new analysts are able to reach the stage of conducting real analyses more quickly after becoming familiar with the required tools.
In addition to helping new analysts complete onboarding and training more quickly, this finding also makes it easier for individuals who might otherwise feel unqualified or intimidated to become analysts. Currently, the complexity of many analysis tools discourages some people from pursuing this career or prevents those without certain technical skills from becoming network forensics analysts [105]. More intuitive and beginner-friendly tools lower this barrier, enabling such individuals to perform analyses that would previously have been harder. This development could ultimately increase the number of professionals entering the field of network forensics analysis.
The prototype also proved to be effective in speeding up the analysis workflow for experienced analysts and in reducing mistakes for both experienced and inexperienced users. This improvement has the potential to enhance cybersecurity, as fewer errors during analyses allow more cyber threats to be identified and mitigated. Faster analyses also create the opportunity to conduct additional analyses within the same timeframe, further increasing the number of threats that can be addressed. Overall, such tools have the potential to strengthen cybersecurity and reduce the number of cyber threats.
The ability to perform more analyses also has broader implications, particularly for companies that employ network forensics analysts. When more analyses can be completed, company revenue may increase, as many firms charge for individual investigations [106]. However, some companies prioritise cost savings over increasing revenue [107]. In such cases, they may reduce staff, since the same number of analyses can now be handled with fewer analysts. While these outcomes may seem opposite, they are two sides of the same coin, reflecting different organisational priorities in response to increased efficiency.
This study found that information is more easily conveyed and understood through the use of visualisations. While this finding applies primarily to analysts, it is also likely that their clients will benefit from the same. As a result, after completing analyses, analysts can communicate their findings more effectively to clients by incorporating the visualisation tool into their explanations.
Finally, for analysts who already use Arkime, this tool does not fully replace it. Both Arkime and the new visualisation tool offer unique functionalities that the other lacks. For these users, the most effective approach is to use both tools together. This combination is likely to result in more thorough and accurate analyses than when using Arkime alone.

6.2.2 Theoretical Implications

The theoretical implications of this study go beyond specific findings, shedding light on how the results contribute to broader concepts and understanding. These implications demonstrate the study's impact on the development of theories in network forensics, while also addressing related domains and advancing the knowledge that forms the foundation for analytical and methodological approaches.

Most notably, this study provides a step-by-step description of how to create and evaluate the functional prototype of a visualisation tool. This contributes to theory by offering a structured methodological framework for translating domain-specific requirements into effective visualisations and tools. Additionally, based on the group of experts interviewed, the purpose of such tools can vary widely, highlighting that tool development must be seen as context-dependent but with a process that is generalisable across domains. For example, the same process could be applied in healthcare, where visualising patient data can support decision-making and diagnoses [108]. By interviewing healthcare experts, this process will result in a domain-specific visualisation tool.
Current research shows that visualisations can help analysts reduce cognitive load, decrease error rates, and generally support their analyses [11]. This study demonstrates a significant improvement in these areas within network forensics analysis when a visualisation tool is used. Therefore, the findings further reinforce this claim and provide additional evidence supporting it.
Although some literature discusses the tasks and challenges faced by network forensics analysts [8], few studies are based on interviews with expert analysts. This study provides a structured overview based on such interviews, offering a clear picture of the analysis process in a field that is often little understood by those without training. As a result, network forensics analysis becomes a better understood domain, and the study contributes valuable knowledge about its practices.
While visualisations can significantly reduce cognitive load and help with pattern recognition for analysts [11], those designed without consideration for experts' hunches may unintentionally eliminate valuable subjective insights [109]. In recognition of this, this study emphasises the preservation of subjectivity by including interactive visualisations that act as indicative guides rather than definitive answers. This approach ensures that analysts' knowledge and hunches remain part of the analysis, allowing for the prototype to support analysts rather than performing the analysis for them, further emphasising the importance of highlighting analysts' hunches.

6.3 Assumptions

During this study, certain assumptions have been made to define the scope of the analysis. These assumptions are categorised into contextual and methodological, forming both the foundation of the research and the approach taken. Contextual assumptions establish the conditions under which the study is conducted. Methodological assumptions outline the procedure and its technical constraints. Addressing these assumptions is essential for understanding the limitations of the study and the interpretation of its findings.

6.3.1 Contextual Assumptions

The study relies on a number of contextual assumptions that describe the conditions under which the research is conducted. These assumptions provide a framework for interpreting results and understanding the limits of the analysis.

Although not immediately apparent, one assumption is that the interviewed analysts possess a baseline knowledge of network forensics and cyber threats. While it is reasonable to expect that they have the necessary expertise given their professional roles, this assumption is acknowledged due to the anonymity of both the analysts and their employing organisation.

Furthermore, it is assumed that the interviews with the analysts resulted in a list of tasks and challenges representative of typical network forensics analysis workflows. The analysts' insights are considered to reflect common practices and challenges, thereby providing a clear overview of standard procedures in network forensics analysis.
During the impact analysis, a specific set of tasks is used to evaluate the new tool in comparison to Arkime. The assumption here is that these tasks accurately reflect real-world analyses, offering a meaningful measure of the tool's impact on network forensics analysis workflows.
Finally, the evaluation of the tools using these tasks requires access to network data. Without realistic data, the tasks cannot generate meaningful results, rendering the evaluation invalid. Therefore, the study relies on network data that is assumed to reflect real-world network traffic.

6.3.2 Methodological Assumptions

The research is based on methodological assumptions that define the procedure of this study. These assumptions help clarify the scope of the study and ensure consistency in the application of the chosen methods.

The assumption with the greatest impact on this study is that Arkime is frequently used by analysts and can serve as the basis for session data. The tool developed in this study uses the Arkime API to fetch session data, utilises the Arkime query language for queries, and allows queries to be executed directly within Arkime. This assumption forms the base for the design and functionality of the tool, ensuring its relevance to real-world network forensics workflows.
Network forensics data can be either static or dynamic. For this study, it was assumed that the required network data is static. Handling both static and dynamic data would require performing the WWH analysis twice, potentially resulting in a tool twice as large, which was considered beyond the scope of this research. Therefore, the study is performed under the assumption of static data.
During the design of the visualisation tool, it was assumed that users' screen sizes are sufficiently large. Specifically, wireframes and mockups were created for a screen resolution of 1920×1080. While the designs can be adapted for smaller or larger screens, significantly smaller screens may limit the effectiveness of some visualisations in conveying information.

6.4 Limitations

While the findings of this study provide valuable insights for both network forensics and related fields, it is important to acknowledge certain limitations that may have affected the findings. Recognising these helps highlight aspects that should be interpreted with caution, and identifies opportunities for future research to build on this study.

One constraint arose from the need for HREC approval before conducting interviews with expert analysts. While such approval is essential to ensure ethical practice and participant privacy, the process was time-intensive, requiring the preparation and creation of various documents and forms before submission. Even after submission, approval took approximately four months, which reduced the time available for data collection, prototype development, and evaluation.
Certain technical choices made early in development introduced challenges later in the process. For example, the use of PrimeVue initially sped up the implementation of tables, but later affected flexibility when the design changed. Similarly, while Vue.js provided an accessible framework, its reliance on props and emit handlers introduced complexity that slowed development and affected maintainability. These trade-offs highlight the importance of evaluating technologies before implementation.
Scalability also presented a challenge. The prototype's visualisations, built with ApexCharts, experienced limitations when handling larger datasets, causing large loading times and sometimes failure to load at all. Although strategies exist to enhance scalability, only a smaller subset was implemented in the current

version. Additionally, some pages, such as the Main Overview page, were not developed into a functional prototype and therefore were not tested for scalability.
Performance constraints were also observed. Visualisations currently load sequentially, which can result in long waiting times when larger datasets are analysed. If the visualisation with the longest loading time happens to load first, all others must wait, even if they would otherwise load and become usable much faster. Additionally, when visualisations reload unnecessarily after tab switches, breaking up the flow of the analysis, usability suffers.
Another limitation concerns the evolving nature of cyber threats. As these threats change over time, certain functionalities may lose relevance or become obsolete. Although the prototype offers features useful for current network forensics analysis, ongoing adaptation and extensibility will be necessary to maintain long-term relevance. Designing tools with future updates in mind allows for smoother adaptation to new threats.
The reliance on external APIs also introduced their corresponding limitations. The prototype depends primarily on Arkime for session data and on a geolocation API. Arkime restricts data retrieval to two million sessions per request, while the geolocation API imposes daily request limits. Such dependencies can restrict the scale and flexibility of analysis.
To ensure that the evaluation could be conducted, priority was given to the Metadata page and partial completion of the Roles page, while the implementation for the Main Overview page was considered outside the study's scope and remained a mock-up. Although this meant that not all functionalities could be evaluated, prioritisation ensured that the most interactive and representative components were evaluated. This allows for the tool to be evaluated in its most active and realistic state.
Finally, in addition to the fact that only a subset of the prototype was evaluated, the impact analysis has further limitations. The evaluation can only be conducted for tasks that are supported by both Arkime and the prototype, since a meaningful comparison requires an overlap. As a result, functionalities that exist exclusively in the prototype are not considered within the analysis, even if they could potentially provide significant benefits. The same applies to functionalities that are available in Arkime but are not included in the prototype. Therefore, the evaluation does not reflect the complete set of functionalities of either system. Instead, it focuses on the shared functionality, which makes the analysis reliable for the overlapping areas but prevents assessment of the overall potential of both systems. The evaluation method was selected specifically to allow for a comparison of overlapping functionalities, and alternative methods were not explored within the scope of this study. This introduces an additional limitation, as employing multiple evaluation approaches could enable a more comprehensive impact analysis, whereas the current study is restricted to only a single method.

6.5 Threats to Validity

Assessing the validity of this study is important to understanding the reliability of the findings. Highlighting potential sources of bias allows for interpretations of the results with appropriate context and confidence. This discussion provides transparency and clarifies how the findings should be interpreted.

The main threat to the validity of this study lies in the impact analysis. Three factors affect the validity of the evaluation. First, the number of participants is very limited. Due to time and resource constraints, only one willing participant of each type could be found, resulting in the evaluation being conducted only twice. With such a small sample, it is difficult to draw definitive conclusions about the prototype's impact, as results might vary with a larger participant group. Second, the evaluation consisted of only ten brief tasks. Although these tasks were designed to reflect realistic network forensics analysis tasks, the limited number of tasks restricts the findings to shorter analyses rather than complete workflows. Furthermore, the small number of tasks, combined with their independent design, limits the impact analysis to short-term effects. As a result, long-term effects are not adequately evaluated. Finally, the evaluation was conducted using an incomplete version of the prototype. Consequently, the results only represent a subset of the prototype's functionality, and the findings could differ if the full prototype were

tested. Overall, the findings of the impact analysis should not be interpreted as definitive or academically valid, but rather as indicative of potential results.
Some threats to the validity of the study also arise with the WWH framework. While it does give a structured overview that can be used to eliminate the majority of options for tasks, it does not provide one single answer. An attempt was made to base the final visualisation choice on literature as much as possible, but choosing visualisations is, by nature, slightly subjective in nature. Furthermore, choosing entries for the WWH framework can also introduce slight subjectivity, since there is no completely objective rule that decides what What and Why selections fit a task. This means that while the WWH framework allows for an academic approach to picking visualisations for tasks, it does introduce a slight subjectivity into the study.
The interviews conducted for this study also pose a potential threat to validity. It cannot be guaranteed that the interviews completely cover the network forensics analysis workflow. Although strategies were implemented to maximise its coverage and capture common tasks, it is still possible that certain relevant points were overlooked. If important tasks were not identified during the interviews, the prototype might lack important functionalities, potentially impacting its effectiveness and the validity of the design.

6.5.1 Validity Threats from Assumptions

The study is built upon assumptions that are used as a foundation for research. While necessary, these assumptions can affect the validity. Therefore, when discussing threats to the validity of the study, it is important to acknowledge which assumptions also pose a threat. The assumptions outlined in Section 6.3 are analysed below to determine which may pose a threat to the validity of this study.

- **Analyst expertise:** The study assumes that the interviewed analysts have baseline knowledge of network forensics and cyber threats. If this assumption is incorrect, the tasks and challenges identified may not represent real-world practices, potentially biasing the prototype design and evaluation.
- **Representativeness of tasks:** It is assumed that the tasks collected from interviews reflect typical network forensics workflows. If they are not, the design of the tool may not be based on reality, affecting its validity.
- **Task set for impact analysis:** The evaluation assumes that the selected tasks mirror real-world analyses. If the tasks do not reflect the diversity or complexity of real workflows, the measured impact of the prototype may be misleading.
- **Network data realism:** The study relies on network data assumed to reflect real-world traffic. Data that is not representative could affect the accuracy of tool performance evaluations, impacting the validity of the results.
- **Arkime usage:** The prototype is designed assuming that Arkime is frequently used by analysts. If Arkime adoption is lower than expected, the relevance and usefulness of the tool could be affected. Since analysts often use a broad range of tools, this also hurts the generalisability of the results.
- **Static data:** The study assumes static network data, while real-world networks may be dynamic. This assumption reduces the tool's generalisability, as tool performance might differ for continuous streams of data.
- **Screen size:** The visualisation tool was designed for 1920×1080 screens. Users with smaller screens may experience reduced usability, affecting the validity of the design.

6.6 Future Work

Identifying directions for future research helps extend the impact of this study beyond its findings. By outlining areas that remain unaddressed or could benefit from deeper research, this section calls for further studies to refine, validate, and expand on the current results, encouraging continued progress in network forensics analysis.

First, future extensions of this study should put emphasis on the interviews, as they form the foundation of the design. The outcomes of these interviews directly guide the development of the prototype. Consequently, conducting additional interviews with a broader range of questions, potentially informed by the findings of this study, could result in a larger and fine-tuned set of tasks. This expanded set of tasks could, in turn, reveal important functionalities that were not incorporated into the current design due to the current limited number of interviews conducted. Therefore, subsequent research should begin by carefully evaluating and, if necessary, iteratively refining the interview process.

As discussed in Section 6.5, further work is required for the evaluation to provide definitive and valid results. Currently, the evaluation was conducted with only two participants, ten short tasks, and an incomplete prototype. These limitations affect the validity of the findings. To properly assess the prototype's impact on network forensics analysis, future evaluations should be conducted with a significantly larger number of participants, while ensuring that there is an equal number of participants for each type. In addition, these evaluations should include a broader and more diverse set of tasks that vary in both length and complexity. Furthermore, a fully implemented version of the prototype must be used to ensure that the evaluation reflects the full and intended design. Together, these improvements would allow for a more reliable and valid assessment of the prototype's effect on real-world analysis workflows. Overall, when conducting the impact analysis in future research, the study should ideally involve a substantially larger participant group, incorporate a more comprehensive set of tasks representing a complete analysis workflow, and be performed using a fully implemented version of the prototype.

Besides improving the current impact analysis method, additional evaluation methods could also provide valuable insights. Currently, only the overlap between Arkime and the prototype is evaluated, which leaves unique functionalities unaddressed. Other evaluation methods more targeted towards the assessment of individual functionalities could be utilised to find the broader impact of the prototype, aside from the overlap. This could include individual usability tests, performance evaluations, scenario-based analysis, and many other methods. Such extensions of the impact analysis, in addition to improving the current method, would allow for a more valid and definitive assessment.

Furthermore, the impact analysis has been conducted only in comparison with Arkime. This assumption affects the validity of the evaluation results, as analysts may use tools other than Arkime. To ensure that the impact analysis reflects network forensics analysis as a whole, future research could perform the impact analysis using additional tools. This approach would provide better insight into whether the prototype genuinely enhances the general analysis workflow or if its benefits are limited to workflows that involve Arkime. Evaluating the prototype against a wider range of tools could also reveal areas for improvement that are not currently addressed.

During the impact analysis, the processes by which participants completed tasks were carefully observed. Although this was initially intended for identifying mistakes, it also revealed valuable insights into potential areas for improvement. For some tasks, participants required more time to complete their work when using the prototype compared to Arkime. While it may appear at first that this was caused by unfitting visualisations, observation during the evaluation indicated that most difficulties stemmed from participants being unable to locate specific components within the prototype. This finding suggests that the tool's navigation structure and component placement may need to be improved. Enhancing these aspects, rather than focusing on improving visualisation design, could lead to significant improvements in efficiency. Continuations of this study that address these issues may therefore show even greater benefits for network forensics analysis workflows.

A component left unimplemented in this study is an Arkime information input. Currently, the prototype relies on a `.env` file to provide this information. While functional, this solution is not user-friendly and requires technical knowledge that may act as a barrier for some users. A more intuitive and accessible approach would be to design a frontend component through which this information could be entered directly. Future development could repeat the process described in Section 4 to implement such a feature, thereby improving usability and reducing entry barriers for potential users.

Accessibility was incorporated into the design process through the use of colour-blind safe palettes. However, many additional accessibility features have not been implemented. Future iterations of the

prototype could be extended to address a wider range of user needs. For example, the interface could include special fonts for users with dyslexia or visual impairments, and provide full keyboard control for users unable to operate a mouse. Expanding accessibility in this way would ensure that the tool can be effectively used by the largest possible group of analysts, improving its impact and inclusivity.

To allow for the prototype to be finished within a feasible timeframe, choices had to be made about what elements to exclude from the design. One such element was roles. Currently, the prototype includes only the roles mentioned by the majority of analysts, but interviews also revealed less common roles that may still be important, such as the Attack Box role or a role responsible for managing other malicious entities. Furthermore, with additional interviews, more important roles could surface. Future research should expand the role set and link these roles to their corresponding tasks and visualisations. Doing so would allow the tool to better reflect the full set of functionalities necessary for network forensics analysis.

The prototype's frontend and backend were created with functionality as its main priority, with less emphasis on optimisation or scalability. Although the design takes these factors into account, the current implementation struggles with performance when used with larger datasets. While this limitation did not affect the evaluation results or the design process itself, it hinders use in real-world settings where scalability and efficiency are more important. Section 5.3.3 already outlines potential solutions to these challenges. Future work should focus on implementing these solutions to improve performance and ensure that the tool can meet the demands of real analyses.

During the prototyping phase described in Section 4.4, feedback was provided iteratively by a consulting analyst to ensure that the prototype was usable in a real-world setting. This feedback was collected on two occasions and incorporated into the design. While this approach was sufficient for the current scope, future work could extend it by increasing both the number of feedback sessions and the number of professionals providing the feedback. Incorporating such expansions in subsequent studies would further align the design with real-world use cases, enhancing its practical usefulness.

Finally, as discussed in Section 6.4, the dynamic and rapidly evolving nature of cyber threats presents an ongoing challenge. Functionalities that are effective today may become outdated or obsolete for new threats. To ensure the relevance and effectiveness of the tool, ongoing development is essential. Future research should therefore prioritise research on new cyber threats and iteratively update the tool with new features and visualisations. This continuous process will help maintain the tool's support in the detection, analysis, and mitigation of emerging threats.

7 Conclusion

This study presents an overview of the development of a functional prototype aimed at supporting network forensics analysis through fine-tuned, task-oriented visualisations, together with an evaluation of its impact on the workflow of analysts. Through interviews with expert network forensics analysts, the key tasks and challenges encountered during analysis were identified. In total, 53 tasks were identified and organised into three categories. Using Tamara Munzner's description of the WWH framework [66], these tasks were then mapped to appropriate visualisation techniques. The visualisations were then distributed across three pages, two of which also contained several tabs. These pages were first sketched to define their layout, after which mockups were created for each page and its tabs to refine details, and finally, the mockups were implemented into a functional prototype. Although not all pages were fully implemented, the impact analysis was conducted on the available functionalities. This analysis demonstrated improvements in effectiveness, intuitiveness, and correctness when comparing the functional prototype to Arkime. Ultimately, the results indicate that such fine-tuned visualisation tools can positively influence the workflow of network forensics analysts.

Besides these direct findings, the study also has broader practical implications. The prototype was shown to reduce the learning curve for new analysts, allowing for faster onboarding and lowering entry barriers into the field of network forensics analysis. This has the potential not only to speed up training but also to attract new professionals who might have been discouraged by the high complexity of current tools. For experienced analysts, the prototype improved speed and reduced errors, supporting more efficient and accurate analyses. These improvements also have an impact on an organisational level, where they can increase revenue, improve cybersecurity, impact staffing choices, and even improve how analysts convey information to clients. Importantly, the tool complements rather than replaces Arkime, and the two tools together can provide an even further enhanced workflow.

The study also discusses theoretical implications, contributing more to knowledge within network forensics and data visualisation. It demonstrates how expert-informed tasks can be translated into functional designs, resulting in a process that is context-dependent yet generalisable across domains such as network forensics and healthcare. Additionally, the findings also support the notion that visualisations reduce cognitive load and error rates, while also emphasising the importance of preserving expert subjectivity through interactive features that guide rather than dictate analysis. By using expert insights as a foundation for prototype development, the study contributes to a clearer understanding of network forensics analysis and offers a methodological foundation for future research.

Despite its contributions, several limitations should be acknowledged. The requirement for HREC approval delayed data collection, prototype development, and evaluation by several months. Technical decisions, including the use of PrimeVue and Vue.js, introduced challenges during later stages of implementation. Scalability issues arose with ApexCharts when handling large datasets, resulting in slower load times, and some pages were not or partly implemented, limiting the scope of the impact analysis. Unoptimised code further affected usability, while reliance on external APIs constrained the flexibility and scale of the prototype. Additionally, the evolving nature of cyber threats means certain functionalities may become outdated over time. Time constraints also necessitated prioritising the implementation and evaluation of specific pages, leaving others as mock-ups. Therefore, the assessment focused on the most interactive and representative components. Recognising these limitations is important for the interpretation of the results and highlights opportunities for future research to improve and expand on the prototype.

Finally, the study addresses areas that benefit from continued research and work. A complete implementation of the functional prototype, along with larger evaluations with more participants, is necessary to provide a more definitive answer regarding the prototype's impact on network forensics analysis. Improvements regarding navigation, support of dynamic data streams, and improved scalability and performance would increase its usefulness in real-world contexts. Increasing the number of covered roles, incorporating more accessibility features, and ensuring the tool keeps up with evolving cyber threats, form possible directions that future studies could use to expand upon this study. By including these extensions,

future research can build on the prototype and contribute to the improvement of network forensics tools. Overall, this study demonstrates that carefully designed, task-oriented network data visualisations can meaningfully support network forensics analysts by addressing their specific challenges, reducing cognitive load, and ultimately improving both the speed and accuracy of their analyses. While limitations in the implementation and evaluation highlight areas for improvement, the findings provide a clear outline for future research to refine, improve, or design visualisation tools.

References

[1] J. Jones, "Cyber crime," in *Encyclopedia of Cryptography, Security and Privacy*, Springer, 2025.

[2] J. Glowniak, "History, structure, and function of the internet," *Seminars in Nuclear Medicine*, vol. 28, no. 2, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001299898800032.

[3] R. Rojas, *How charles babbage invented the computer*, 2023. arXiv: 2311.04371 [cs.AR]. [Online]. Available: https://arxiv.org/abs/2311.04371.

[4] FBI, *Cybercrime — FBI*, https://www.fbi.gov/investigate/cyber.

[5] G. Palmer, "A road map for digital forensic research," in *Report from the First Digital Forensic Research Workshop (DFRWS 2001)*, Utica, New York, Aug. 2001.

[6] Dimension Market Research. "Network forensics market: Comprehensive industry landscape and strategic outlook." (Mar. 2025), [Online]. Available: https://dimensionmarketresearch.com/report/network-forensics-market/.

[7] Verified Market Research. "Apac digital forensics market valuation – 2024-2031." (Jun. 2025), [Online]. Available: https://www.verifiedmarketresearch.com/product/apac-digital-forensics-market/.

[8] E. S. Pilli, R. Joshi, and R. Niyogi, "Network forensic frameworks: Survey and research challenges," *Digital Investigation*, vol. 7, no. 1, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287610000113.

[9] H. Mohammed, N. Clarke, and F. Li, "An automated approach for digital forensic analysis of heterogeneous big data," *Journal of Digital Forensics, Security and Law*, vol. 11, no. 2, 2016.

[10] L. Niandong, S. Tian, and T. Wang, "Network forensics based on fuzzy logic and expert system," vol. 32, Nov. 2009.

[11] J. J. DiCarlo, D. Zoccolan, and N. C. Rust, "How does the brain solve visual object recognition?" *Neuron*, vol. 73, no. 3, Feb. 2012.

[12] Syracuse University. "What is data visualization? benefits, types & best practices." (Mar. 2025), [Online]. Available: https://ischool.syracuse.edu/what-is-data-visualization/.

[13] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*, 4th. Morgan Kaufmann, 2007.

[14] R. Mohanakrishnan. "What is a computer network? definition, objectives, components, types, and best practices." (Dec. 2021), [Online]. Available: https://www.spiceworks.com/tech/networking/articles/what-is-a-computer-network/.

[15] Living Internet, *Arpanet – the first internet*, Jan. 2000. [Online]. Available: http://www.livinginternet.com/i/ii_arpanet.htm.

[16] B. M. Leiner, V. G. Cerf, D. D. Clark, *et al.*, *A Brief History of the Internet*. Internet Society, 1997.

[17] Cloudflare, *What is the OSI model?* https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/.

[18] G. Bora, S. Bora, S. Singh, and S. M. Arsalan, "Osi reference model: An overview," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 7, no. 4, 2014.

[19] M. M. Alani, "Osi model," *Guide to OSI and TCP/IP Models*, 2014.

[20] D. Rafter. (Jun. 2024), [Online]. Available: https://us.norton.com/blog/privacy/what-is-an-ip-address.

[21] P. Wu, Y. Cui, J. Wu, J. Liu, and C. Metz, "Transition from ipv4 to ipv6: A state-of-the-art survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, 2013.

[22] E. I. of Technology. (Apr. 2024), [Online]. Available: https://www.eit.edu.au/resources/difference-between-mac-address-and-ip-address/.

[23] J. English. (Aug. 2024), [Online]. Available: https://www.techtarget.com/whatis/video/MAC-address-vs-IP-address-explained.

[24] Cloudflare, *What is a packet? — network packet definition*, https://www.cloudflare.com/learning/network-layer/what-is-a-packet/.

[25] The MITRE Corporation. "Artifact details — mitre d3fendtm: Network session." (2025), [Online]. Available: https://d3fend.mitre.org/dao/artifact/d3f:NetworkSession/.

[26] Netseccloud. "The lifecycle of a tcp session: From start to finish." (Sep. 2024), [Online]. Available: https://netseccloud.com/the-lifecycle-of-a-tcp-session-from-start-to-finish.

[27] R. T. Fielding, J. Gettys, J. C. Mogul, *et al.* "Hypertext transfer protocol – http/1.1: Semantics and content." (1999), [Online]. Available: https://www.rfc-editor.org/rfc/rfc2616.

[28] J. Reschke. "Hypertext transfer protocol (http/1.1): Semantics and content." (2014), [Online]. Available: https://www.rfc-editor.org/rfc/rfc7231.

[29] Mozilla Developer Network, *Http request methods*, https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods.

[30] Mozilla Developer Network, *Http response status codes*, https://developer.mozilla.org/en-US/docs/Web/HTTP/Status.

[31] Google Security Team, *Https encryption on the web*, https://transparencyreport.google.com/https/overview.

[32] Mozilla. "Http headers - http — mdn." (2025), [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers.

[33] The Postman Team. "What are http headers? — postman blog." (Jul. 2023), [Online]. Available: https://blog.postman.com/what-are-http-headers/.

[34] J. Park, "Vpn: Privacy and anonymity for all," *Geo. L. Tech. Rev.*, vol. 2, 2017.

[35] CDW, *What is a VPN?* https://www.cdw.com/content/cdw/en/glossary/vpn.html, [Online]. Accessed: 2025-09-04.

[36] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous connections and onion routing," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, May 1998. [Online]. Available: https://www.nrl.navy.mil/itd/chacs/sites/www.nrl.navy.mil.itd.chacs/files/pdfs/98-Anonymous-Connections-and-Onion-Routing.pdf.

[37] S. Goldschlag Reed, "Onion routing for anonymous and private internet connections," *Communications of the ACM*, vol. 42, no. 2, Feb. 1999. [Online]. Available: https://www.nrl.navy.mil/itd/chacs/sites/www.nrl.navy.mil.itd.chacs/files/pdfs/99-Onion-Routing-for-Anonymous-and-Private-Internet-Connections.pdf.

[38] G. Arora, "Improving the performance and security of tor's onion services," *Proceedings on Privacy Enhancing Technologies*, vol. 2025, no. 1, 2025.

[39] J. Postel. "Transmission control protocol." (1981), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc793.

[40] J. Postel. "User datagram protocol." (1980), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc768.

[41] J. Postel and J. Reynolds. "File transfer protocol (ftp)." (1985), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc959.

[42] P. Ford-Hutchinson. "Securing ftp with tls." (2005), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc4217.

[43] T. Ylonen and C. Lonvick. "The secure shell (ssh) protocol architecture." (2006), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc4251.

[44] P. Mockapetris. "Domain names - concepts and facilities." (1987), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1034.

[45] P. Mockapetris. "Domain names - implementation and specification." (1987), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1035.

[46] J. Postel. "Internet control message protocol." (1981), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc792.

[47] J. Postel. "Simple mail transfer protocol." (1982), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc821.

[48] P. Hoffman. "Smtp service extension for secure smtp over tls." (2002), [Online]. Available: https://datatracker.ietf.org/doc/html/rfc3207.

[49] S. Malik, "Network security principles and practices," 2003.

[50] S. Gordon and R. Ford, "On the definition and classification of cybercrime," *Journal in computer virology*, vol. 2, no. 1, 2006.

[51] S. Das and T. Nayak, "Impact of cybercrime: Issues and challenges," *International journal of engineering sciences & Emerging technologies*, vol. 6, no. 2, 2013.

[52] T. Caldwell, "Ethical hackers: Putting on the white hat," *Network Security*, vol. 2011, no. 7, 2011.

[53] Arkime, *Arkime*, https://github.com/arkime/arkime.

[54] CISA, *Wireshark — cisa*, https://www.cisa.gov/resources-tools/services/wireshark.

[55] U. Lamping and E. Warnicke, "Wireshark user's guide," *Interface*, vol. 4, no. 6, 2004.

[56] N. Hamidli, "Introduction to ui/ux design: Key concepts and principles," *Preuzeto*, vol. 28, no. 3, 2023.

[57] T. Green and K. Brandon, *UX Design with Figma: User-Centered Interface Design and Prototyping with Figma*. Springer, 2024.

[58] S. Qureshi, J. Li, F. Akhtar, S. Tunio, Z. H. Khand, and A. Wajahat, "Analysis of challenges in modern network forensic framework," *Security and Communication Networks*, vol. 2021, no. 1, 2021.

[59] G. McCarthy Emme. "Iris+ thematic taxonomy," Global Impact Investing Network (GIIN). (May 2021), [Online]. Available: https://iris.thegiin.org.

[60] D. Joy, F. Li, N. Clarke, and S. Furnell, "A user-oriented network forensic analyser: The design of a high-level protocol analyser," 2014.

[61] M. O'Connor and W. Walley, "A framework for computer-based data analysis and visualisation by pattern recognition," in *Modelling Community Structure in Freshwater Ecosystems*, Springer, 2005.

[62] R. Savvides, A. Henelius, E. Oikarinen, and K. Puolamäki, "Significance of patterns in data visualisations," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019.

[63] M. Sturdee, L. Thornton, B. Wimalasiri, and S. Patil, "A visual exploration of cybersecurity concepts," New York, NY, USA: Association for Computing Machinery, 2021.

[64] Cambridge University Press, *Aggregation. cambridge advanced learner's dictionary & thesaurus*, https://dictionary.cambridge.org/dictionary/english/aggregation.

[65] GeeksforGeeks. “What is data enrichment?” (May 2024), [Online]. Available: https://www.geeksforgeeks.org/data-engineering/what-is-data-enrichment/.

[66] T. Munzner, *Visualization Analysis and Design* (A K Peters Visualization Series). Boca Raton, FL: CRC Press, 2014. [Online]. Available: https://www.crcpress.com/Visualization-Analysis-and-Design/Munzner/p/book/9781466508910.

[67] A. Chen. “What is a pie chart?” (Jul. 2025), [Online]. Available: https://www.explo.co/blog/pie-chart.

[68] M. Yi, *A complete guide to pie charts*, https://www.atlassian.com/data/charts/pie-chart-complete-guide.

[69] A. F. Langley. “Why we should avoid using pie/donut charts?” (Apr. 2024), [Online]. Available: https://coreanalitica.com/why-we-should-avoid-using-pie-donut-charts.

[70] M. Zivkovic. “Histogram vs. bar chart - which one to use and when?” (Jan. 2024), [Online]. Available: https://www.luzmo.com/blog/histogram-vs-bar-chart.

[71] N. Desbarats. “Have i resolved the pie chart debate?” (Dec. 2023), [Online]. Available: https://nightingaledvs.com/have-i-resolved-the-pie-chart-debate/.

[72] Inforiver, *A comprehensive guide to bar charts: When to use them and how to design them*, https://inforiver.com/insights/guide-to-bar-charts-when-use-them-how-design/.

[73] G. S. “Stacked vs. grouped bar charts in blazor: Which is better for data visualization?” (Apr. 2025), [Online]. Available: https://www.syncfusion.com/blogs/post/stacked-vs-grouped-bar-charts-in-blazor-which-is-better-for-data-visualization.

[74] M. Yi, *A complete guide to line charts*, https://www.atlassian.com/resources/data-visualization/line-charts.

[75] Lewis. “What is a line plot and how does it work.” (Aug. 2025), [Online]. Available: https://www.fanruan.com/en/blog/what-is-a-line-plot.

[76] Inforiver. “Introduction to variance chart in power bi.” (May 2025), [Online]. Available: https://www.inforiver.com/blog/introduction-to-variance-chart-in-power-bi.

[77] JMP Statistical Discovery LLC, *Scatter plot*, https://www.jmp.com/en/statistics-knowledge-portal/exploratory-data-analysis/scatter-plot.

[78] Lewis. “Scatter plot examples and applications explained.” (Aug. 2025), [Online]. Available: https://www.fanruan.com/en/blog/scatter-plot-examples.

[79] MESCIUS USA, Inc., *Scatter and bubble charts*, https://developer.mescius.com/activereportsnet/docs/report-authors/report-controls/report-controls-page-rdl-report/chart-page-rdl/plots/scatter-bubble-dvcharts.

[80] T. Garrett. “Parallel coordinates plots for multivariate data analysis,” DEV3LOPCOM, LLC. (May 2025), [Online]. Available: https://www.dev3lop.com/parallel-coordinates-plots.

[81] insightsoftware. “When and why to use heat maps.” (Jun. 2022), [Online]. Available: https://insightsoftware.com/blog/when-and-why-to-use-heat-maps/.

[82] Editverse, *Network graphs: Visualizing relationships and connections in research data*, https://editverse.com/network-graphs-visualizing-relationships-and-connections-in-research-data.

[83] L. C. Muth. “What to consider when choosing colors for data visualization.” (May 2018), [Online]. Available: https://blog.datawrapper.de/colors-in-data-visualization/.

[84] C. Stokes, A. Arunkumar, M. A. Hearst, and L. Padilla, “An analysis of text functions in information visualization,” *IEEE Transactions on Visualization and Computer Graphics*, Jul. 2025. [Online]. Available: https://arxiv.org/pdf/2507.12334v1.

[85] J. Liew, *Badge ui*, https://mobbin.com/glossary/badge.

[86] D. J. Slutsky, "The use of tables," *Journal of Wrist Surgery*, vol. 3, Nov. 2014.

[87] C. A. Services. "How do i present data in an academic article?" (Aug. 2020), [Online]. Available: https://www.cwauthors.com/article/Presenting-data-tables-figures-in-academic-article.

[88] Figma, *What is figma?* https://help.figma.com/hc/en-us/articles/14563969806359-What-is-Figma.

[89] ISC2. "Women in cybersecurity: Women in the profession." (Apr. 2024), [Online]. Available: https://www.isc2.org/Insights/2024/04/Women-in-Cybersecurity-Report-Women-in-the-Profession.

[90] P. Pesko. "Why men are colorblind more often than women." (Sep. 2020), [Online]. Available: https://pilestone.com/blogs/news/why-men-are-colorblind-more-often-than-women.

[91] Arkime, *Viewer v3.x – v5.x api*, https://arkime.com/apiv3.

[92] P. Siriwardena, *Advanced API Security: OAuth 2.0 and Beyond*, Second. San Jose, CA, USA: Apress, 2020.

[93] Golang, *Build simple, secure, scalable systems with go*, https://go.dev/.

[94] Golang, *Frequently asked questions (faq)*, https://go.dev/doc/faq.

[95] Golang, *Standard library*, https://pkg.go.dev/std.

[96] Vue, *Vue.js: Reactivity in depth*, https://vuejs.org/guide/extras/reactivity-in-depth.

[97] TypeScript, *Javascript with syntax for types*, https://www.typescriptlang.org/.

[98] PrimeVue, *Vue ui component library*, https://primevue.org/.

[99] ApexCharts. "Open source javascript charts for your website." (Mar. 2020), [Online]. Available: https://apexcharts.com/.

[100] D. T. T. Mai, N. T. Dat, P. M. Bao, C. Q. Truong, and N. T. Tung, "Ddos attacks detection using dynamic entropy in software-defined network practical environment," *International Journal of Computer Networks & Communications (IJCNC)*, vol. 15, no. 3, May 2023.

[101] S. Raghavan, B. Golden, and E. Wasil, Eds., *Telecommunications Modeling, Policy, and Technology* (Operations Research/Computer Science Interfaces Series). New York, NY, USA: Springer, 2008, vol. 44.

[102] P. Tol. "Introduction to colour schemes." (Aug. 2021), [Online]. Available: https://sronpersonalpages.nl/~pault/.

[103] S. Duveen, *Network Forensics Visualisation Tool*, https://github.com/SDuveen/Network-Forensics-Visualisation-Tool, 2025.

[104] NetWitness. "Network forensic tools: The key to network forensics." (Jun. 2023), [Online]. Available: https://www.netwitness.com.

[105] K. Kelley. "Career exploration: Is data analytics hard?" (Sep. 2024), [Online]. Available: https://pg-p.ctme.caltech.edu/blog/data-analytics/is-data-analytics-hard.

[106] F. Data. "Paying for a private digital forensics lab." (2025), [Online]. Available: https://www.flashbackdata.com/paying-for-a-private-digital-forensics-lab/.

[107] J. Maverick. "Is it more important for a company to lower costs or increase revenue?" (2024), [Online]. Available: https://www.investopedia.com/ask/answers/122214/company-it-more-important-lower-costs-or-increase-revenue.asp.

[108] M. Gan. “How data visualization is transforming healthcare decision making,” Soy Marta Gan Blog. (Mar. 2025), [Online]. Available: https://soymartagan.com/en/revisit-consent-button.

[109] H. Lin, D. Akbaba, M. Meyer, and A. Lex, “Data hunches: Incorporating personal knowledge into visualizations,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 1, 2023.

A Interview Questions

Main Question

Could you walk me through a typical case from start to finish? Explain each step clearly and why you performed that step.

Workflow Questions

Q1: What roles do you typically assign to a host within the network when determining its function? How do you decide what role to assign?

Q2: What are the three most frequent actions you perform during the analysis process?

Q3: What actions take the most time to perform during your analysis?

Q4: What parts of the analysis process do you think would benefit the most from change?

Data Questions

Q1: What levels of aggregation do you most commonly use?

Q2: What kinds of data enrichment do you use during your analysis?

Q3: Generally, how large are the datasets you analyse, approximately?

Q4: When analysing large datasets, which aspects do you believe should be visualised in real time versus included in a final report?

Visualisation Questions

Q1: Do you use visualisations during the analysis? If so, what types of visualisation do you use and how do they support your analysis?

Q2: Do you use any existing visualisation tools during the analysis? If so, which ones do you use and how do they help with your analysis process?

Wrap-up Questions

Q1: Is there anything important that we have not discussed that you think would be valuable for this interview?

B Analyst Task Overview

Items marked with a larger and differently coloured bullet indicate custom tasks that were added in addition to the interview tasks.

General Tasks

- `GT01` - Label adversaries
- `GT02` - Label victims
- `GT03` - Distinguish adversary infrastructure
- `GT04` - Distinguish victim elements
- `GT05` - Distinguish non-important elements
- `GT06` - Distinguish entity roles
- `GT07` - View open ports per IP
- `GT08` - Perform Arkime query
- `GT09` - View sessions

Metadata-related Tasks

Overview Metrics

- `MT-OM01` - View total number of packets
- `MT-OM02` - View total number of sessions
- `MT-OM03` - View total duration
- `MT-OM04` - View total data size

Frequency Analysis & Count

- `MT-FC01` - Frequency analysis / count on IP
- `MT-FC02` - Frequency analysis / count on MAC
- `MT-FC03` - Frequency analysis / count on SSH sessions for IP
- `MT-FC04` - Frequency analysis / count on geolocation
- `MT-FC05` - Frequency analysis / count on HTTP methods
- `MT-FC06` - Frequency analysis / count on protocols
- `MT-FC07` - Frequency analysis / count on TCP flags
- `MT-FC08` - Frequency analysis / count on DNS hosts

Miscellaneous

- `MT-MS01` - Distinguish HTTP from HTTPS
- `MT-MS02` - View session durations
- `MT-MS03` - View session volumes
- `MT-MS04` - Session distribution over time of day per weekday
- `MT-MS05` - Identify VPN/TOR traffic
- `MT-MS06` - Identify sexually explicit traffic
- `MT-MS07` - Find port usage anomalies
- `MT-MS08` - Find IPs with high destination entropy
- `MT-MS09` - Find IPs with low destination entropy
- `MT-MS10` - Find high volumes per IP
- `MT-MS11` - Distinguish SSH from non-SSH traffic
- `MT-MS12` - Find DNS sessions with high volumes

Role-related Tasks

Proxy

- RT-PR01 - Detect proxy-related protocols
- RT-PR02 - Count proxy-related HTTP headers
- RT-PR03 - Check if the input-output byte ratio is near one
- RT-PR04 - Compare input and output data volumes over time

C2

- RT-C201 - Inspect traffic payloads for plaintext or decryptable content
- RT-C202 - Determine if encryption keys were transmitted with the payload
- RT-C203 - Apply FFT to session intervals to identify communication frequency patterns
- RT-C204 - Show distribution of frequency counts

Scanner

- RT-SC01 - Check for SSH connection at start
- RT-SC02 - Count the number of connection attempts to different IPs
- RT-SC03 - Count the number of connection attempts to different ports
- RT-SC04 - Inspect the traffic payload volume
- RT-SC05 - Inspect traffic payload content
- RT-SC06 - Check the number of failed connections
- RT-SC07 - Check if the number of connections fits the baseline
- RT-SC08 - Find detection patterns in the number of unique IPs, unique ports, and failed connections

Exfiltration

- RT-EX01 - Count incoming sessions using file transfer protocols
- RT-EX02 - Count outgoing sessions using file transfer protocols
- RT-EX03 - Compare incoming data volume to subsequent outgoing data volume
- RT-EX04 - Find IPs with high incoming and low outgoing file transfers of large volume

C Metadata page tabs

Addresses

- `MT-FC01` - Frequency analysis / count on IP
- `MT-FC02` - Frequency analysis / count on MAC
- `MT-FC04` - Frequency analysis / count on geolocation
- `MT-MS05` - Identify VPN/TOR traffic
- `MT-MS08` - Find IPs with high destination entropy
- `MT-MS09` - Find IPs with low destination entropy
- `MT-MS10` - Find high volumes per IP

Protocols

- `MT-FC03` - Frequency analysis / count on SSH sessions for IP
- `MT-FC05` - Frequency analysis / count on HTTP methods
- `MT-FC06` - Frequency analysis / count on protocols
- `MT-FC07` - Frequency analysis / count on TCP flags
- `MT-MS01` - Distinguish HTTP from HTTPS
- `MT-MS07` - Find port usage anomalies
- `MT-MS11` - Distinguish SSH from non-SSH traffic

Sessions

- `MT-FC08` - Frequency analysis / count on DNS hosts
- `MT-MS02` - View session durations
- `MT-MS03` - View session volumes
- `MT-MS04` - Session distribution over time of day per weekday
- `MT-MS06` - Identify sexually explicit traffic
- `MT-MS12` - Find DNS sessions with high volumes

D Wireframes

Main Overview

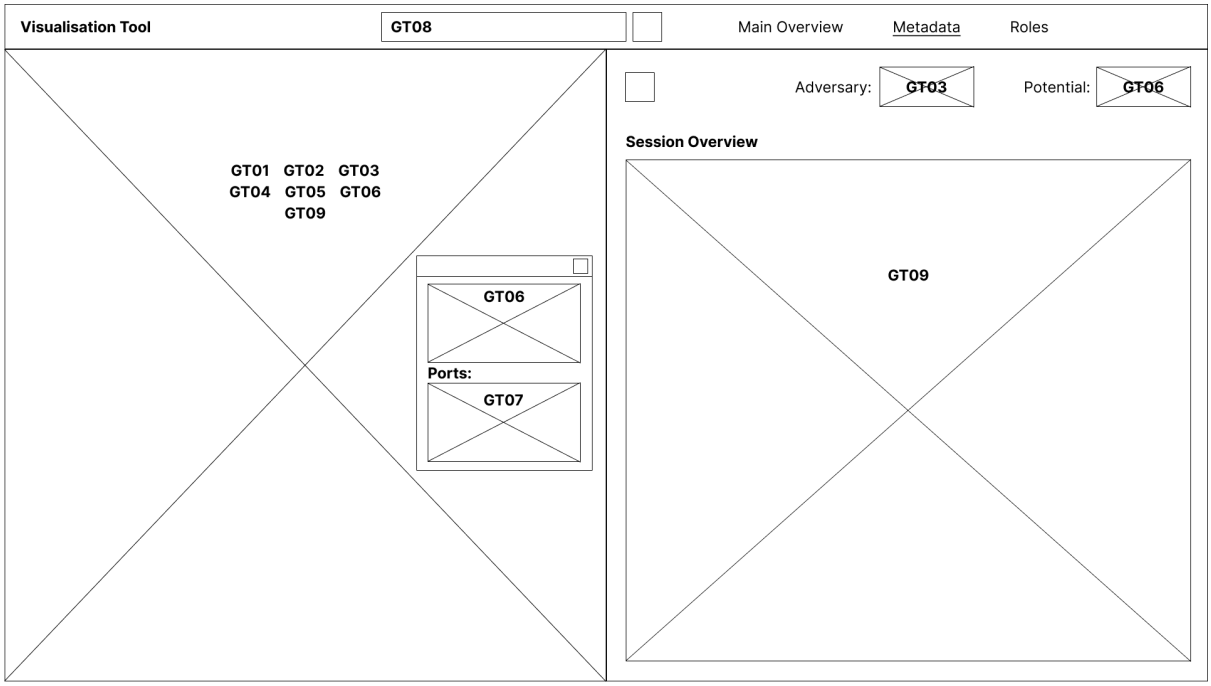

Figure 14: Main Overview page wireframe

Metadata

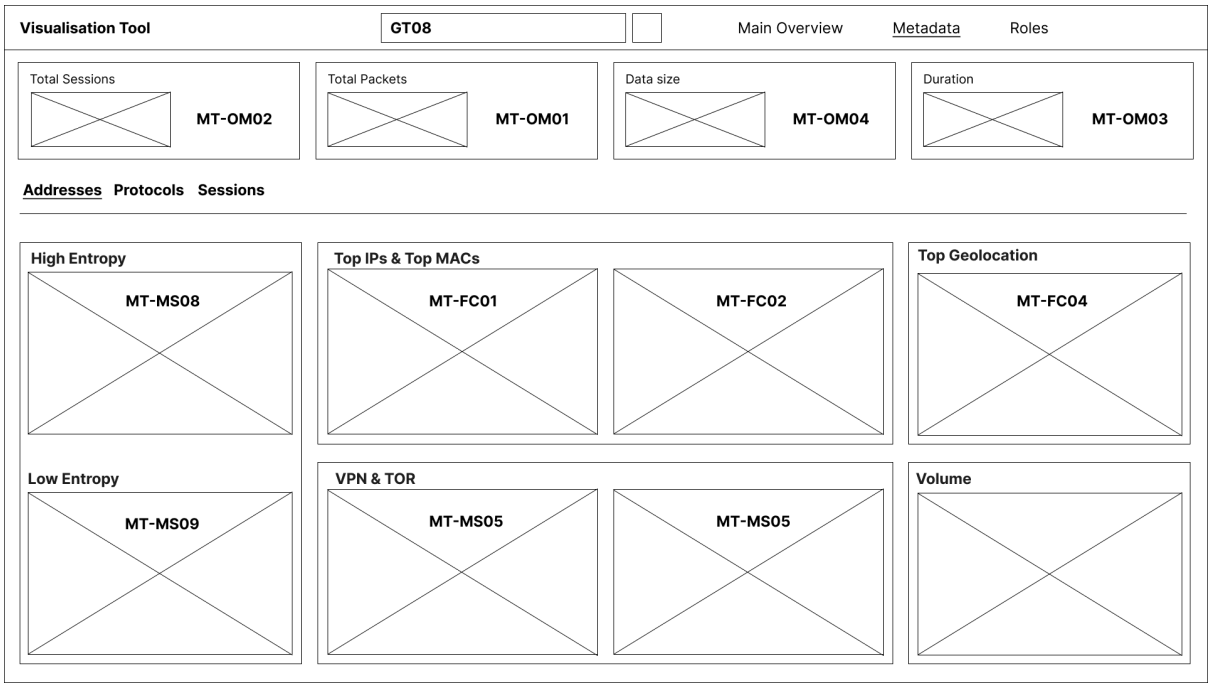

Figure 15: Metadata page - Addresses tab wireframe

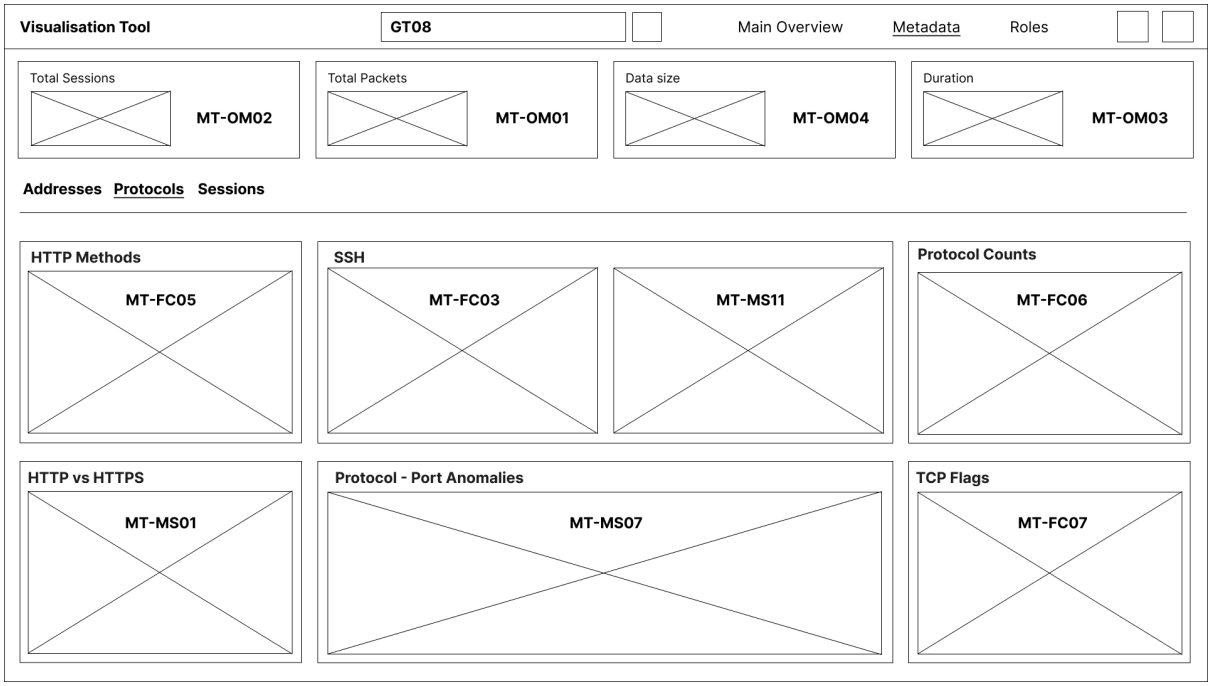

Figure 16: Metadata page - Protocols tab wireframe

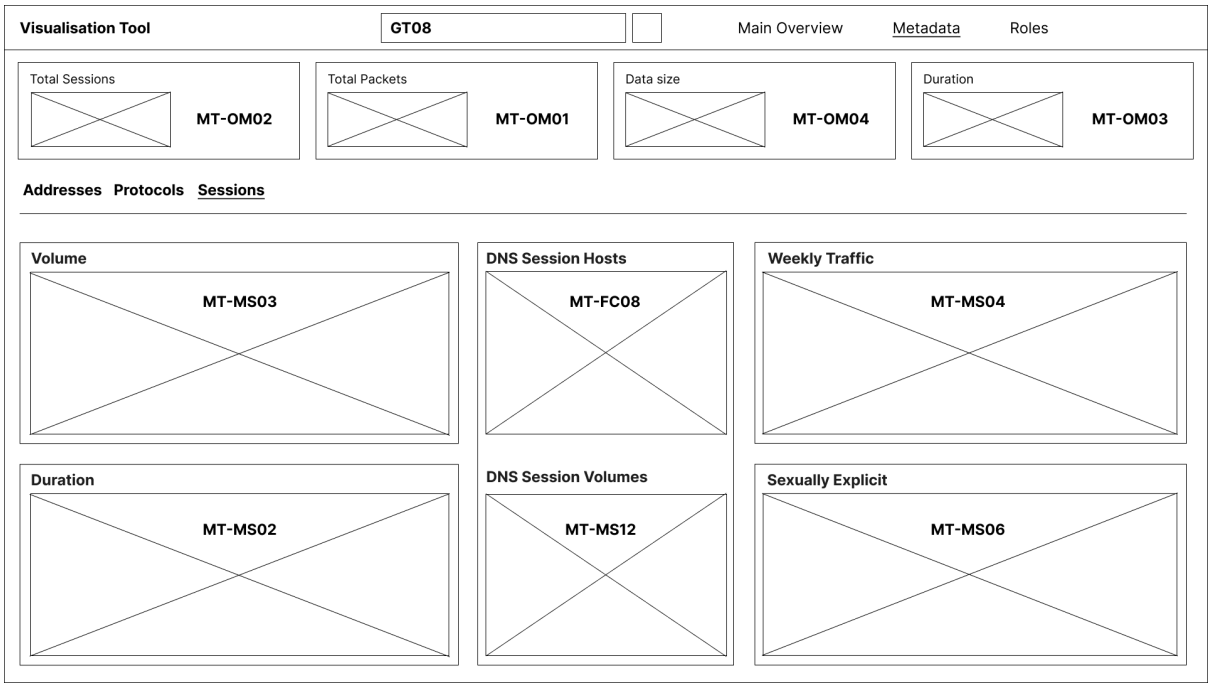

Figure 17: Metadata page - Sessions tab wireframe

Roles

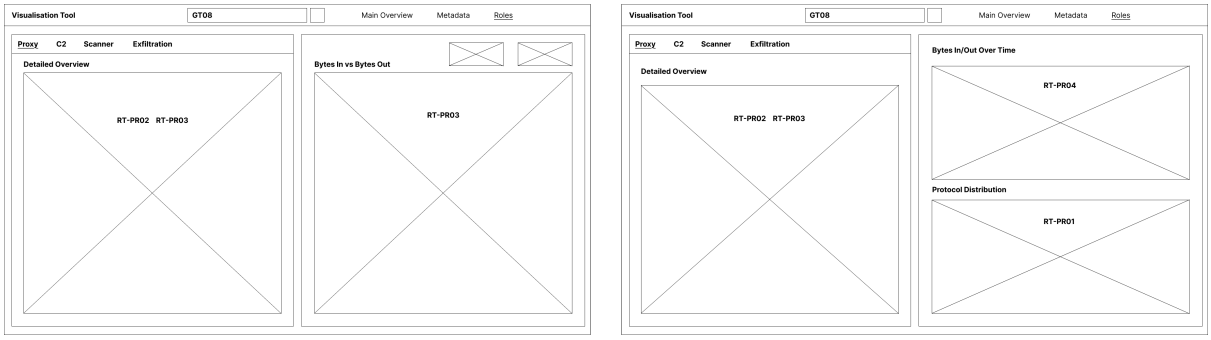

Figure 18: Roles page - Proxy tab wireframes

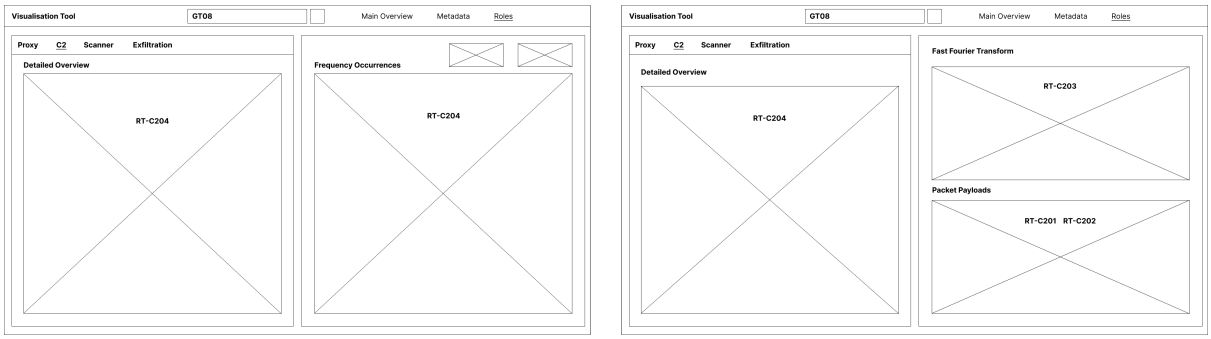

Figure 19: Roles page - C2 tab wireframes

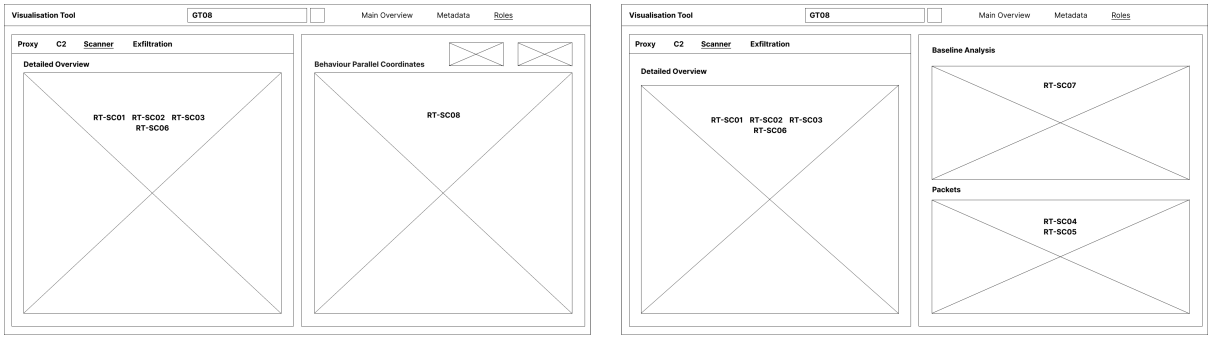

Figure 20: Roles page - Scanner tab wireframes

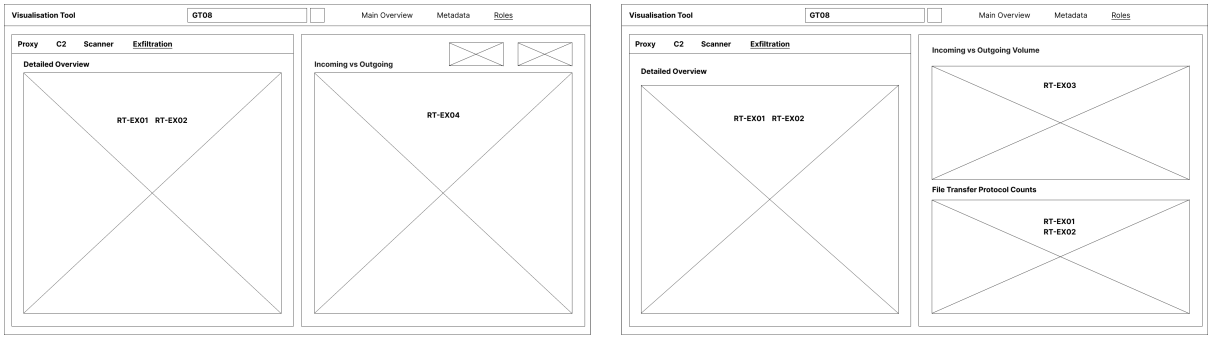

Figure 21: Roles page - Exfiltration tab wireframes

E Mockups

Main Overview

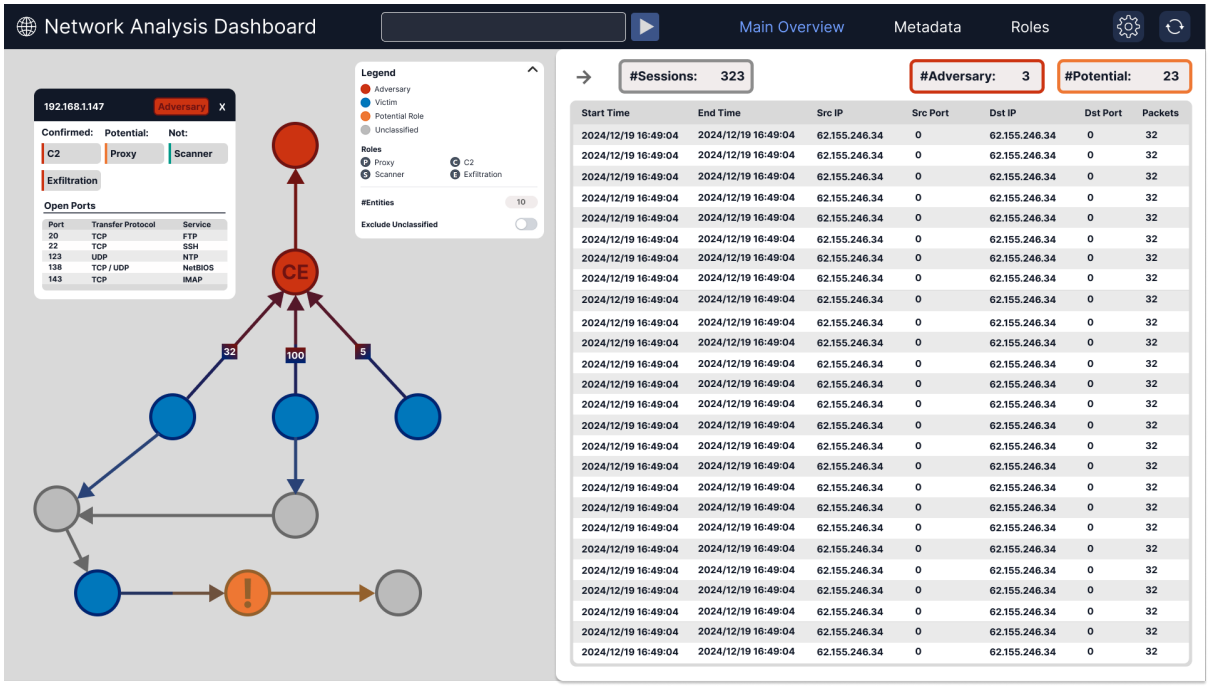

Figure 22: Main Overview mockup

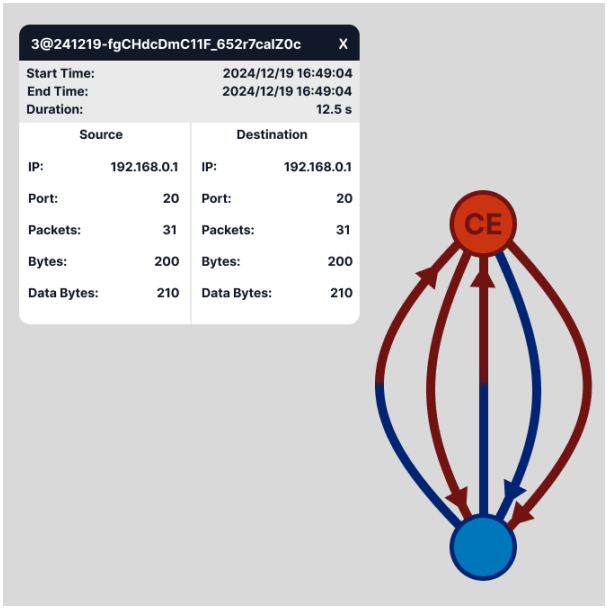

Figure 23: Detailed session overview mockup

Metadata

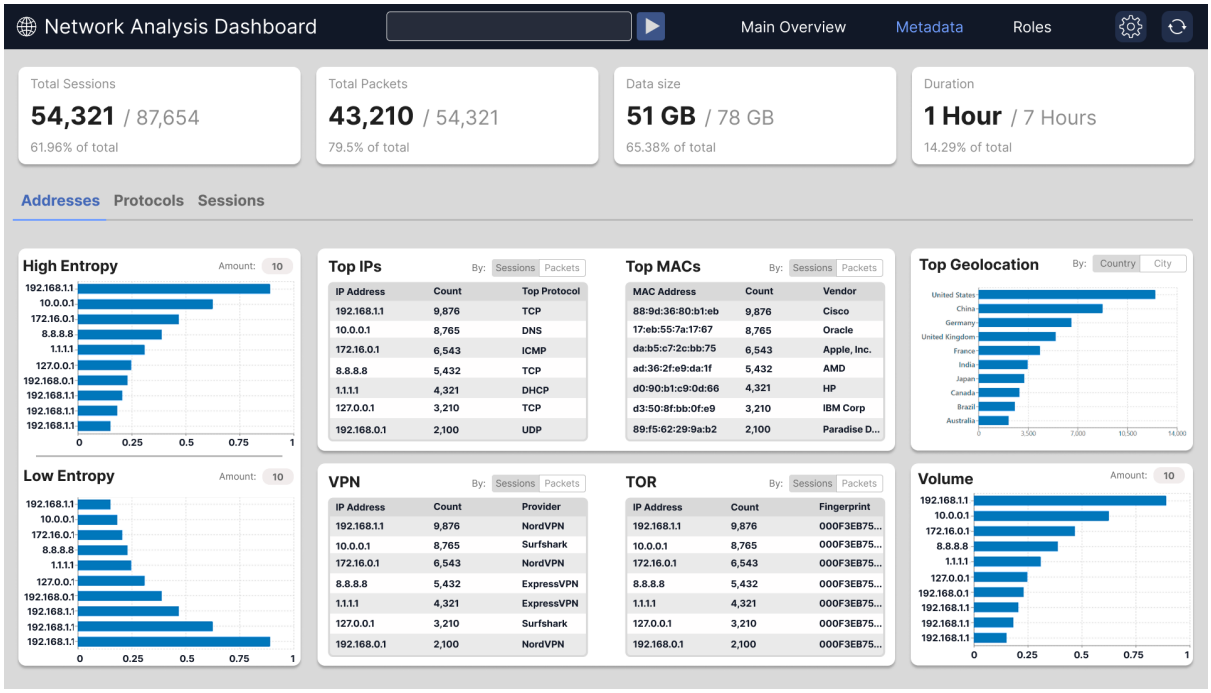

Figure 24: Metadata Addresses tab mockup

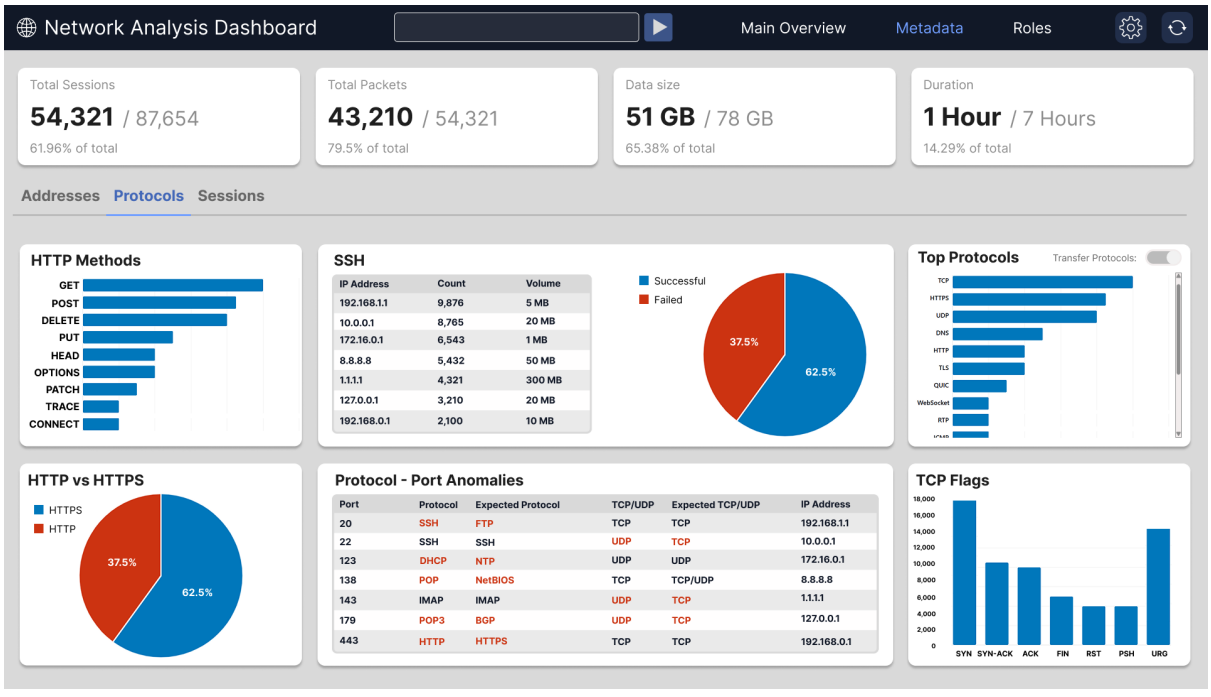

Figure 25: Metadata Protocols tab mockup

Figure 26: Metadata Sessions tab mockup

Roles

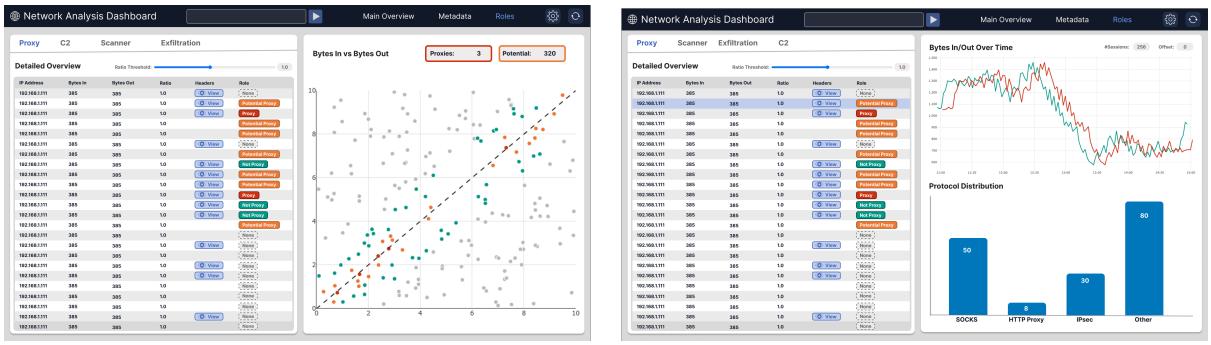

Figure 27: Roles Proxy tab mockups

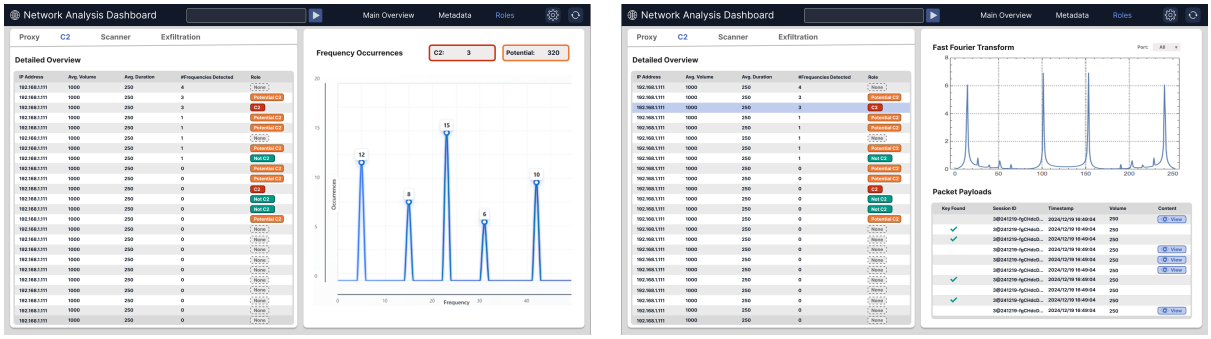

Figure 28: Roles C2 tabs mockups

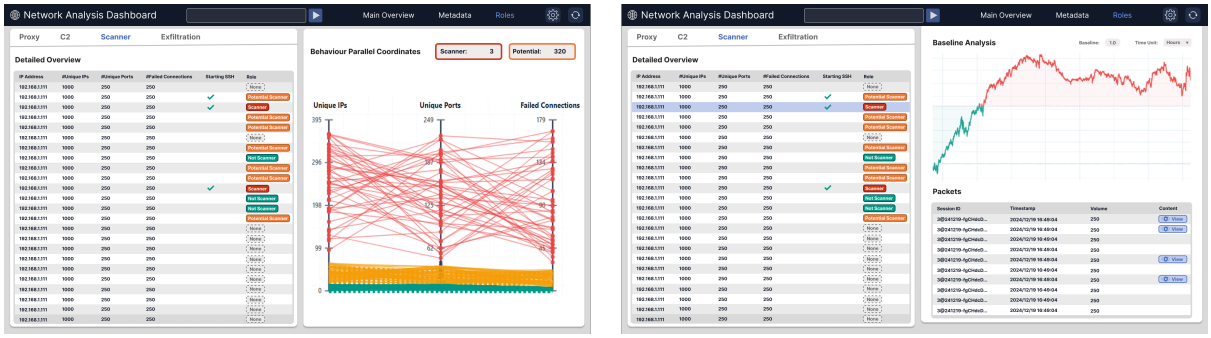

Figure 29: Roles Scanner tab mockups

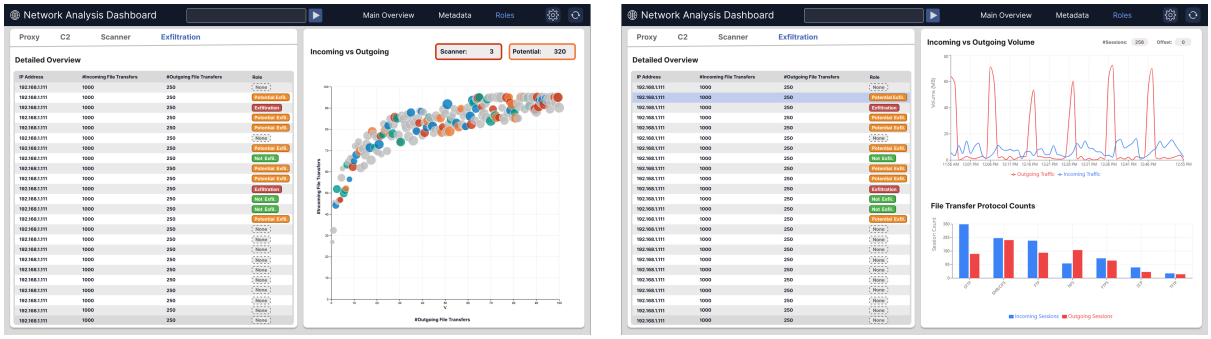

Figure 30: Roles Exfiltration tabs mockups

F Tool Documentation

Search Bar

Each page of the tool features a search bar at the top, similar to Arkime. It supports the same query language as Arkime, with a few additional features designed to improve usability and flexibility.

Query Input

When typing in the search input, two icons appear:

- **Query Summary icon**: opens a pop-up that displays a structured version of the query. This helps make long or complex queries easier to understand and handle.
- **Clear**: removes the current query.

Build Queries from Visualisations

You can also build queries by interacting with visual elements in the tool. Clicking on parts of a visualisation, such as a chart or label, adds that attribute to the current query. This allows you to construct queries directly from the data, whether it's a simple IP filter or something more detailed.

Buttons

Two buttons are located next to the input field:

- **Run Query**: executes the current query and refreshes the data. Queries do not run automatically while typing, so this gives you control over when to update the results.
- **Open in Arkime**: opens a new tab and runs the same query in Arkime. This makes it easy to continue your investigation there without copying and pasting, and helps avoid mistakes.

Pages Overview

The tool is organised into three pages, each focused on a different aspect of the data and its objectives:

- **Main Overview**: A network graph that represents the dataset. This feature is currently under development.
- **Metadata**: Contains visualisations for various metadata fields and aggregations. It helps users filter and explore different aspects of the data.
- **Roles**: Includes visualisations designed to assist in detecting a set of predefined adversary roles within the data.

Main Overview

To be implemented.

Metadata

The **Metadata** page provides an overview of the dataset through a combination of summary metrics and interactive visualisations.
At the top, four cards display general overview metrics:

- Number of sessions
- Number of packets
- Data size
- Duration of the dataset

These metrics update dynamically when filters are applied through queries, showing values for the filtered subset compared to the full dataset.
Below the summary cards are three tabs: **Addresses**, **Protocols**, and **Sessions**. Each tab contains visualisations relevant to its attribute category. Some visualisation cards also include action buttons in the top right corner, allowing users to customise their behaviour.

Tab Overviews

Addresses

- Destination IP entropy
- Top IPs
- Top MACs
- Top geolocations
- VPNs
- TOR nodes
- Total IP volumes

Protocols

- HTTP methods
- SSH
- Top protocols
- HTTP vs HTTPS
- Protocol-port anomalies
- TCP flags

Sessions

- Volume over time
- DNS host session counts
- Weekly traffic
- Duration buckets
- DNS session volumes
- Sexually explicit traffic

Roles

The **Roles** page is designed to assist in detecting four predefined adversary roles:

- Proxy
- C2
- Scanner *(not implemented)*
- Exfiltration *(not implemented)*

Each role section follows the same structure:
On the left is a table listing IP addresses, with columns showing attributes relevant to the selected role. The final column contains a badge indicating detection status, which can be one of the following:

- **None**
- **Potential [ROLE]** *(automatically detected)*
- **[ROLE]** *(manually confirmed)*
- **Not [ROLE]** *(manually dismissed)*

Some roles allow customisation through a value above the table. Changing this value adjusts the automatic detection behaviour for that role.
On the right is a visualisation showing detection activity across the full or filtered dataset. Above it, counters display the number of potential, confirmed, and dismissed roles.
When an IP address in the table is selected, the right panel updates to show two visualisations that assist in determining whether the selected IP fits the current role.