

**A UNIFIED FRAMEWORK TO ENHANCE MALWARE INCIDENT MAN-
AGEMENT IN SECURITY OPERATIONS CENTRES**

Ph.D. Thesis



Dakouri Raphael Gazo

Ph.D. in Computer Science
University of Staffordshire

This dissertation is submitted for the degree of
Doctor of Philosophy

September 2026

I dedicate this thesis to my parents, friends and family . . .

Declaration

I hereby declare that, except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this or any other university. This dissertation is my work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words, including appendices, bibliography, footnotes, tables and equations, and has fewer than 150 figures.

Dakouri Raphael Gazo
September 2026

Acknowledgements

Undertaking this PhD has been a truly life-changing experience for me, and I am grateful to have had the support and guidance of so many people along the way.

I would like to express my heartfelt gratitude and acknowledgement to my supervisor, Dr Asma PATEL, for their constant support, guidance, and inspiration throughout the realisation of this project. Dr Asma has been a constant source of motivation and a driving force behind the success of this endeavour. Their technical expertise, profound knowledge, and innovative thinking have been instrumental in shaping the direction and quality of this research work. I am deeply indebted to Dr Asma for their emotional support and encouragement during the most challenging phases of this project. Their empathy, understanding, and belief in my abilities have been a beacon of hope, propelling me forward even in the face of seemingly insurmountable obstacles. Dr Asma has not only been a mentor but also a true inspiration. Their passion for excellence, dedication to the pursuit of knowledge, and resilient commitment to academic rigour have set an exemplary standard for me to emulate. Their guidance has been invaluable in improving my research skills, critical thinking, and problem-solving abilities, ultimately shaping me into a better researcher and scholar. Furthermore, I am grateful for Dr Asma's willingness to engage in thought-provoking discussions and their openness to exploring unconventional ideas. Their ability to challenge my perspectives and push me beyond my intellectual boundaries has been a catalyst for personal and professional growth. Dr Asma's attention to detail, constructive feedback, and consistent support have been instrumental in refining and elevating the quality of this research work. Their insightful comments and suggestions have not only improved the technical aspects but have also enriched the overall depth and clarity of the project. I am truly fortunate to have had the opportunity to work under the guidance of such an exceptional supervisor, whose solid belief in me has been a constant source of motivation and inspiration. Dr Asma's mentorship has left an indelible mark on my academic journey, and I will forever cherish the invaluable lessons learnt from their wisdom and expertise. Words cannot fully express the depth of my gratitude and appreciation for Dr Asma's contributions to this project and my personal growth as a researcher. Their impact extends far beyond the confines of this work, and I am

forever bound to them for shaping me into the scholar I am today.

I am also grateful to Dr Mohammad HASSAN, whose expertise and guidance were instrumental in shaping my research. His insights and suggestions helped me to refine my ideas and to approach my research questions in new and innovative ways.

I would also like to thank the members of my thesis committee, who generously gave their time and expertise to provide feedback on my work. Their observations and advice played a key role in shaping my research and refining my ideas.

I would like to extend my deepest gratitude to Anicet Gnahoua GODI, whose invaluable assistance and expertise in coding have been instrumental to the success of this project. Your dedication, patience, and technical proficiency have been critical in the development of the application, which would not have been possible without your support. Your contributions have not only enhanced the project but have also ensured its functionality and user experience. Your willingness to share your knowledge and skills has been a constant source of inspiration and motivation. This project stands as a testament to the power of collaboration and the importance of having friends who are willing to go above and beyond to help achieve a common goal. Thank you, Anicet, for your support and for being an integral part of this journey. Your help has made a significant impact, and for that, I am profoundly grateful.

I would like to extend my sincere gratitude to Mr David H. SEYMOUR for their support throughout this arduous journey. David has been an invaluable source of strength and encouragement, offering a listening ear and a shoulder to lean on during the most challenging times. Their empathy, compassion, and belief in my abilities have been a constant source of motivation, pushing me forward even when the path seemed arduous. Their friendship has been a true blessing, providing a much-needed respite from the demands of academic life and reminding me of the importance of maintaining a balanced perspective. I am deeply grateful for their presence, their ability to lift my spirits, and their genuine interest in my research endeavours.

I would like to express my profound gratitude and heartfelt appreciation to my beloved spouse, Teta, and our cherished children for their unconditional love, support, and sacrifice throughout this incredible journey. Teta, your belief in me, patience, and willingness to shoulder additional responsibilities have been the driving force behind my perseverance. Your selfless support and understanding during the countless late nights and weekends dedicated

to this research have played a key role in making this effort a reality. You have been my rock, my solace, and my greatest source of strength, offering encouragement and comfort when the challenges seemed insurmountable. To my dearest children, Ange and Chrysostome, your innocent smiles, boundless love, and endless curiosity have been the ultimate motivation to push forward. Your understanding and sacrifices, despite my occasional absence, have been a humbling reminder of the importance of this work. Your unconditional love and admiration have been a constant source of inspiration, fuelling my determination to create a better future for you and generations to come.

Finally, to my colleagues and friends, thank you for your camaraderie and support. I am fortunate to have had the opportunity to work alongside such a talented and dedicated group of people, and I have learnt so much from each of you.

Thank you all for being a part of my life and for making this PhD journey such a meaningful and memorable experience.

List of Included Publications

The following publication was produced as a direct result of this research.

Conference Paper Gazo, D., Patel, A., Hasan, M. (2024). Effectiveness of Malware Incident Management in Security Operations Centres: Trends, Challenges and Research Directions. Proceedings of the 10th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP, 162-169. SciTePress. DOI: 10.5220/0012389900003648

This published work makes an important contribution to the dissertation by carefully examining and combining knowledge on malware incident response practices in different SOC settings. It provides a foundation for understanding current challenges in managing cybersecurity incidents, especially with the rise of more advanced malware. By pointing out key weaknesses in existing response methods, the research opens new directions for further study that connect directly with the main goals of this thesis.

The study focuses on the central question of how security operations centres can continue to respond effectively to malware threats in a constantly changing attack environment. Through a detailed review of published literature and real-world practices, it highlights major difficulties faced by cybersecurity teams when dealing with complex threats. These findings strongly support the thesis argument that SOCs need better incident management processes and stronger technical expertise.

This research was presented during the 10th International Conference on Information Systems Security and Privacy (ICISSP 2024), a respected academic forum that regularly draws cybersecurity researchers and practitioners from around the globe. SciTePress handles the conference publication process, with proceedings being catalogued in prominent academic indexing systems to maximise research visibility and impact.

Abstract

Security Operations Centres (SOCs) serve as the frontline defence for organisations battling cyber threats, yet managing malware incidents remains a substantial challenge. The constantly evolving nature of cyber threats, combined with limited resources and disconnected investigation tools, frequently hinders swift and precise incident responses. This research addresses these issues by exploring the operational difficulties SOC teams encounter when handling malware incidents and introducing an integrated solution to strengthen their decision-making processes.

The research examines current practices in malware incident management within SOC environments through comprehensive literature analysis and investigation of established procedures. It centres on four key research questions: what types of malware incidents are most prevalent? What obstacles do SOC analysts routinely face? Which solutions prove effective in addressing these difficulties? And how can incident response capabilities be improved? Key obstacles identified include poor documentation practices, challenges in detecting and categorising malware, the ever-evolving threat landscape, insufficient resources, breakdowns in communication, and issues surrounding incident triage and prioritisation.

To address these obstacles, this research proposes a unified framework for malware incident management that consolidates threat intelligence sources, streamlines analytical processes, and provides SOC personnel with an integrated approach to decision-making. This framework, implemented as a practical platform called *SecOps_360*, brings together various scattered tools and procedures into a single cohesive system. This framework comprises conceptual designs, high-level and low-level architectural specifications, UML diagrams, and process flow models that collectively define a structured approach to incident management. *SecOps_360* represents the practical realisation of this framework, delivering a functional platform that seeks to minimise investigation duration, boost accuracy, and strengthen overall incident response effectiveness.

The research employs a mixed-methods approach, combining systematic literature analysis, conceptual design principles, and practical implementation. Platform development follows recognised software engineering standards, incorporating both functional and non-functional requirements derived from real SOC operational environments. The implemen-

tation utilises modern frontend and backend technologies to create a robust, user-friendly interface that supports SOC workflows. The framework assessment involves technical validation, including functionality and compatibility testing, supplemented by scenario-based evaluations comparing conventional malware investigation approaches with SecOps_360-enhanced methods.

Results from comprehensive scenario testing demonstrate that SecOps_360 significantly enhances the efficiency and effectiveness of malware incident handling. Analysis of traditional investigation methods revealed alarmingly high resource utilisation, with CPU usage at 85% and memory consumption at 92% during concurrent searches across multiple threat intelligence platforms. The cumulative search time of 96.6 seconds for basic threat intelligence gathering highlighted significant inefficiencies in conventional approaches. In contrast, the SecOps_360-enhanced investigation process substantially reduced both system resource overhead and search times, demonstrating the platform's capacity to streamline workflows and accelerate incident response. These findings underscore the critical need for integrated solutions in modern SOC environments, particularly as the volume and complexity of cyber threats continue to escalate.

Theoretically, it presents a comprehensive framework encompassing conceptual designs, high-level and low-level architectures, UML diagrams, and process flow models that establish a structured approach to malware incident management. Practically, it delivers SecOps_360, a fully functional platform implementation that demonstrates the framework's real-world applicability through integrated frontend and backend systems. Additionally, it contributes a systematic literature review synthesising current SOC challenges and solutions. The comparative analysis with related work further positions SecOps_360 within the broader context of security platform development, highlighting its unique approach to addressing SOC operational challenges. Together, these contributions provide both theoretical foundations and practical solutions that advance the field of security operations and incident management.

Table of contents

List of figures	xvi
List of tables	xviii
1 Introduction	1
1.1 Background	2
1.2 Research Problem Statement	3
1.3 Research Context	5
1.3.1 Research Aim and Objectives	5
1.3.2 Scope of the research	6
1.3.3 Ethical Consideration	7
1.3.4 Research Contributions	8
1.4 Thesis Structure	9
2 Literature Review	11
2.1 Introduction	12
2.2 Systematic Literature Review Methodology	13
2.2.1 Review Protocol and Objectives	13
2.2.2 Search Strategy	13
2.2.3 Inclusion and Exclusion	14
2.2.4 Study Selection and Quality Assessment	14
2.2.5 Synthesis Approach	14
2.3 Background: Terminology and Theoretical Foundations	14
2.4 SOC Incident Management Lifecycle	17
2.4.1 Preparation	17
2.4.2 Detection and Analysis	19
2.4.3 Containment	20
2.4.4 Eradication	21

2.4.5	Recovery	21
2.5	SOC Architecture: People, Technology, and Processes	24
2.5.1	People	26
2.5.2	Technologies	30
2.5.3	Processes	34
2.6	Malware Detection and Analysis Techniques	38
2.6.1	Malware Detection	38
2.6.2	Malware analysis	40
2.7	SOC Operational Challenges: Critical Assessment	54
2.7.1	Alert Overload and Triage Inefficiency	54
2.7.2	Documentation and Incident Reporting	54
2.7.3	Malware Detection and Identification	55
2.7.4	Fragmented Tool Ecosystems and Investigation Latency	57
2.7.5	Resource and Staffing Constraints	57
2.7.6	Communication and Collaboration Deficits	58
2.7.7	Incident Triage and Prioritisation	59
2.7.8	Rapidly Changing Malware Environment	60
2.7.9	Incident Containment and Remediation	61
2.7.10	Triage and Incident Prioritisation	62
2.8	Criteria-Driven Comparison of Existing Industry Tools	70
2.9	Research Gap	70
2.10	Chapter Summary	75
3	Research Methodology	81
3.1	Introduction	82
3.2	Research Philosophy and Framework	82
3.3	Systematic Literature Review (Supporting RO1)	82
3.4	Conceptual Design Approach (Supporting RO2)	84
3.5	Implementation Approach (Supporting RO3)	84
3.6	Evaluation Methodology (Supporting RO4)	84
3.6.1	Functional and Compatibility Testing	85
3.6.2	Scenario-Based Performance Testing	85
3.7	Ethical Considerations in the Research Process	85
3.7.1	Data Handling in the SLR	85
3.7.2	Use of Threat Intelligence APIs	86
3.7.3	Research Integrity and Bias	86
3.7.4	Responsible Disclosure	86

3.8	Limitations of the Methodology	86
3.9	Chapter Summary	88
4	Conceptual Design of the Framework	89
4.1	Introduction	90
4.2	Collecting and Analysing Project Requirements	92
4.2.1	Functional Requirements	93
4.2.2	Non-functional Requirements 95	
4.2.3	Scope	96
4.3	Target audience	98
4.3.1	SOC Personnel	98
4.3.2	IT Security Professionals	99
4.4	System Detailed Design	101
4.4.1	SOC Team Roles and Module Functionality	101
4.4.2	UML and Process Flow Diagrams	103
4.4.3	High-Level Design	107
4.4.4	Low-Level Design	107
4.5	System Tools and Software	108
4.5.1	List of Utilised Design Software	111
4.5.2	Motivation for Tool Selection	113
4.6	How Will SOC Teams Benefit from <i>SecOps_360</i>	113
4.7	Sketching Graphical Interface	116
4.7.1	Brainstorming	116
4.7.2	Wireframing	119
4.7.3	Mockup	120
4.8	Chapter Summary	122
5	Implementation of <i>SecOps_360</i>	125
5.1	Introduction	126
5.2	Implementation Requirements	127
5.3	System Development	128
5.3.1	Backend Development	128
5.3.2	Frontend Development	130
5.4	Platform Interface and Components Prototype	133
5.5	Chapter Summary 136	

6	Testing and Evaluation	138
6.1	Introduction	139
6.2	Functional and Compatibility Testing	139
6.2.1	Functionality Testing	141
6.2.2	Compatibility Testing 142	
6.3	Scenario-based Performance Evaluation	144
6.3.1	Design Rationale	144
6.3.2	Scenario 1: Traditional Manual Investigation (without <i>SecOps_360</i>) 144	
6.3.3	Scenario 2: <i>SecOps_360</i> -Assisted Investigation	154
6.4	Evaluation	160
6.5	Comparative Analysis with Related Work	164
6.5.1	Secure Web Browsing with the OP Web Browser	165
6.5.2	Software Security Design for Cyber-Physical Systems	166
6.5.3	A Secure Computing Platform for Building Automation Using Microkernel- based Operating Systems	166
6.5.4	Justification for Our Testing Approach	166
6.6	Controlled Experiment with SOC Analyst Participants	167
6.6.1	Overview and Motivation	167
6.6.2	Research Question	168
6.6.3	Experimental Design	168
6.6.4	Participants	169
6.6.5	Ethical Considerations	170
6.6.6	Experimental Task and IoC Dataset	171
6.6.7	Individual Analyst Feedback Reports	172
6.6.8	Aggregated Results Across All Analysts	173
6.6.9	Statistical Analysis	175
6.6.10	Evidence Interpretation and Conclusions	178
6.7	Chapter Summary	181
7	Conclusion and Future Work	183
7.1	Research Summary	184
7.1.1	Research Objective 1: State of the Art and Challenges	184
7.1.2	Research Objective 2: Framework Design	185
7.1.3	Research Objective 3: Platform Implementation	186
7.1.4	Research Objective 4: Evaluation	186

7.2	Evaluation of <i>SecOps_360</i>	187
7.2.1	Preliminary Scenario-Based Evaluation	187
7.2.2	Controlled Experiment with SOC Analyst Participants	187
7.2.3	Comparative Position in the Tool Landscape	188
7.3	Contributions to the Field	189
7.3.1	Primary Contribution 1: The <i>SecOps_360</i> Framework	189
7.3.2	Primary Contribution 2: Quantified Empirical Evidence of Investigation Inefficiency	189
7.3.3	Primary Contribution 3: The <i>SecOps_360</i> Prototype Platform	190
7.3.4	Secondary Contributions	190
7.4	Limitations of the Study	191
7.4.1	Statistical Rigour and Sample Size in the Experimental Evaluation	191
7.4.2	Ecological Validity and Evaluation Context	193
7.4.3	Absence of Usability and Cognitive Load Assessment	193
7.4.4	Scope of the Industry Tool Comparison	194
7.4.5	Platform Maturity and Security	194
7.4.6	Applicability of the Systematic Literature Review	195
7.4.7	Threat Landscape Currency and Long-Term Relevance	195
7.5	Future Research Directions	197
7.6	Recommendations for Implementation	200
7.7	Final Reflections	201
References		203
Appendix A Appendix		217
A.1	Appendix 1 - BackEnd Source Code	218
Appendix B Appendix		229
B.1	Frontend Source Code	229
Appendix C Appendix		236
C.1	SOC Incident Creation Playbook	237
C.1.1	Step 1: Identify Potential Incident	237
C.1.2	Step 2: Triage the Event	237
C.1.3	Step 3: Determine Incident Priority	237
C.1.4	Step 4: Determine Incident Creation Criteria	238
C.1.5	Step 5: Create Incident Record (for P1, P2, and escalated P3 incidents)238	

C.1.6	Step 6: Notify and Escalate	238
C.1.7	Step 7: Transfer to Incident Response Team	238
Appendix D	Appendix	240
D.1	Scenario 1 Test Results	241
D.1.1	IP Search Results	241
D.1.2	URL/DOMAIN Search Results	241
D.1.3	Hash Search Results	241
D.2	Scenario 2 Test Results	241
D.2.1	IP Search Results	241
D.2.2	URL/DOMAIN Search Results	241
D.2.3	Hash Search Results	241

List of figures

2.1	Architecture of SOC in terms of People	26
2.2	SOC Technology Architecture	33
2.3	SOC Data-Collection Workflow	34
2.4	Architecture of SOC in terms of Processes	37
2.5	Signature-based Detection for Ocean-Lotus malware	40
2.6	Static Property Analysis Features of malware brbbot	42
2.7	Example of Assembly Instructions - Malware brbbot	45
2.8	Identified Literature by Scope	80
2.9	Chronology of Literature	80
4.1	SOC Incident Creation Process Sample	94
4.2	SOC Teams' Collaboration With the Centralised Research Platform	103
4.3	<i>SecOps_360</i> UML Diagram	104
4.4	SecOps360 SOC Process Flow Diagram	105
4.5	Flowchart	108
4.6	Low-Level Design	111
4.7	Proposed Wireframe	120
4.8	Sketched Mockup	122
5.1	Backend Source Code Block1	129
5.2	Backend Source Code Block2	130
5.3	Backend Source Code Block3	131
5.4	Component Class Declaration	132
5.5	Form Initialisation Code	133
5.6	Form Validation Code	134
5.7	Form Submission Code	135
5.8	Visual Representation of the Final Product	136
6.1	SOC Internal Incident Management Process Steps	141

6.2	IP Validation Test	144
6.3	Domain Validation Test	145
6.4	W3C Standard Test Result	146
6.5	Browser Compatibility Test Results	148
6.6	Scenario 1 Event Logs	149
6.7	Scenario 1 IP Search Results on AbuseIPDB	152
6.8	Scenario 1 Hash Search Results on MetaDefender	153
6.9	Scenario 1 Work Load	154
6.10	Scenario 2 Work Load	158
6.11	Scenario 2 IP Search Results	159
6.12	Scenario 2 Hash Search Results	160
A.1	Backend Source Code Block1	218
A.2	Backend Source Code Block2	219
A.3	Backend Source Code Block3	220
A.4	Backend Source Code Block4	221
A.5	Backend Source Code Block5	222
A.6	Backend Source Code Block6	223
A.7	Backend Source Code Block7	224
A.8	Backend Source Code Block8	225
A.9	Backend Source Code Block9	226
A.10	Backend Source Code Block10	227
A.11	Backend Source Code Block11	227
A.12	Backend Source Code Block12	228
A.13	Backend Source Code Block13	228
D.1	Scenario 1 IP Search Results on AbuseIPDB	241
D.2	Scenario 1 IP Search Results on VirusTotal	242
D.3	Scenario 1 URL/DOMAIN Search Results on MetaDefender	242
D.4	Scenario 1 URL/DOMAIN Search Results on VirusTotal	243
D.5	Scenario 1 Hash Search Results on MetaDefender	243
D.6	Scenario 1 Hash Search Results on VirusTotal	244
D.7	Scenario 2 IP Search Results	245
D.8	Scenario 2 URL/DOMAIN Search Results	246
D.9	Scenario 2 Hash Search Results	247

List of tables

2.1	List of reviewed Literature for Malware Incident Management	25
2.2	Technology Asset List by Category	31
2.3	List of Literature Regrouped by SOC Architecture	39
2.4	Malware Analysis Tools	41
2.5	List of API Call Sequences per Malicious Behaviour	44
2.6	Example of DLL used in Ransomware with Related Function Calls	46
2.7	Example List of Physical and Virtual Memory Analysis Tools	48
2.8	Classification of Literature	50
2.9	SOC Challenges by Category of SOC Architecture	64
2.10	List of literature in SOC Challenges Section	66
2.10	List of Literature Related to SOC Challenges	69
2.11	Criteria-Driven Comparison of Existing SOC Platforms and SecOps ₃₆₀	71
2.12	Steps and Recommendations for a Systematic Literature Review on Malware Incident Management in SOC _s	76
3.1	Research Framework Mapped to Objectives and Methods	83
4.1	High-Level Design Components and Descriptions	109
4.2	LLD IP Address – Detailed Actions Taken by Step	109
4.3	LLD HASH – Detailed Actions Taken by Step	110
4.4	LLD DOMAIN – Detailed Actions Taken by Step	110
4.5	LLD FILE - Detailed Actions Taken by Step	112
4.6	Brainstorming Ideas	117
4.6	Brainstorming Ideas	119
5.1	Imported Classes from Namespaces and Descriptions	130
6.1	Classification of Testing Inputs with Expected Outputs per IoC	142
6.2	Functionality Test Results	143

6.3	Browser Compatibility Test Results	143
6.4	Baseline Performance Test Plan	147
6.5	Scenario 1 Performance Test Results	150
6.6	<i>SecOps_360</i> Test Plan	156
6.7	Scenario 2 Performance Test Results	157
6.8	Comparative Table of the Traditional Threat Investigation and the Proposed Innovative Platform	163
6.9	Participant Profile Summary	170
6.10	Experimental IoC Dataset	172
6.11	Analyst A Individual Performance Record	173
6.12	Analyst B Individual Performance Record	174
6.13	Analyst C Individual Performance Record	174
6.14	Aggregated Performance Data (All Analysts, Both Conditions)	175
6.15	Wilcoxon Signed-Rank Calculations — Task Completion Time (DV1)	176
6.16	Wilcoxon Signed-Rank Calculations — Peak CPU Utilisation (DV3)	176
6.17	Effect Size Calculation: Cohen's d (Task Completion Time)	177
7.1	Summary of Research Contributions	192
7.2	Summary of Limitations and Their Mitigations	196
A.1	Imported Classes from Namespaces and Descriptions	218

Chapter 1

Introduction

1.1 Background

Cyber threats are increasing in frequency and complexity, forcing organisations to rethink their defensive posture and shorten the time between threat detection and resolution (Cho et al., 2020). The Security Operations Centre (SOC), referred to in CyBOK as the organisational hub for Security Operations and Incident Management (SOIM) (2021cybok, Ch. 8), is the primary entity responsible for this defence. SOC analysts rely on Security Information and Event Management (SIEM) platforms, Open Source Intelligence (OSINT) feeds, and threat-intelligence sharing mechanisms to monitor, detect, and respond to security events. Malware, defined in CyBOK as “malicious software designed to disrupt, damage, or gain unauthorised access to a computer system”, remains the most prevalent threat category facing SOC. Malware families include viruses, worms, Trojan horses, ransomware, spyware, and more advanced subtypes such as rootkits and advanced persistent threat (APT) implants (2021cybok, Ch. 6). The detection and analysis of these artefacts typically draws on both static and dynamic analysis techniques, supported by Indicators of Compromise (IoCs) such as IP addresses, file hashes, and domain names. Despite significant investment in tooling, many SOC. operate with fragmented workflows. Threat intelligence specialists, Tier 1/2/3 analysts, incident responders, and managers each rely on separate dashboards and processes. This fragmentation slows incident response, creates information gaps, and increases cognitive load on analysts who must correlate evidence across multiple platforms.

The need to control the impact or damage caused by the evolving threats demands fast investigations using appropriate and effective tools and technologies (Cook et al., 2023; Dhaka and Johari, 2016). As a security defence entity, a SOC is a team of security professionals who protect networks constantly against cyberattacks (Forsberg and Frantti, 2023; Oesch, 2020). They use security information and event management (SIEM) tools combined with news feeds and open-source intelligence to develop their awareness of the latest cybersecurity developments in the SOC (Alves et al., 2021; Preuveneers et al., 2023). One of the most significant solutions to counter modern cyberattacks is log-based cyber threat hunting. Cyber threat hunting is defined as the method of proactively and iteratively investigating for malicious activities in various types of logs that are critical to early-stage detection (Chen et al., 2022; Gao, P. et al., 2021). Security professionals examine potential malicious programmes to determine their harmful intent and classify them according to their functionality, such as viruses, spyware, Trojan horses, worms, ransomware, etc. (Chai et al., 2024; Moser et al., 2007a; Saqib et al., 2024). An SOC with the computer security incident response team (CSIRT) provides security operations services and incident management in proactive and reactive manners Jacobs et al. (2013). To prevent threats from exploiting system flaws,

various types of threat modelling techniques can be used to defend against anticipated threats (Gnanasekaran et al., 2025; Jacobs et al., 2013). The preventive methods established by an SOC are responsible for safeguarding vital information, planning, and creating contingency procedures, as well as identifying and responding to attacks. In short, conducting malware investigations in a SOC environment does not go without challenges (Alharbi, 2020; Mwangi et al., 2023).

This chapter establishes the foundation for the research. It begins with the background that outlines the evolving landscape of cybercrimes and the critical role of SOCs in defending against sophisticated threats. The research problem statement articulates the specific challenges faced by security professionals when conducting malware investigations in SOC environments. The research context situates the study within the broader field of cybersecurity operations and incident response. Finally, the chapter concludes with an overview of the thesis structure, providing a roadmap of the subsequent chapters.

1.2 Research Problem Statement

In today's fast-moving cyber threat landscape, SOCs are under constant pressure to detect, analyse, and neutralise incidents in real time. Yet, in many organisations, SOC teams operate in isolation. Threat intelligence specialists, analysts, engineers, incident responders, penetration testers, and managers each rely on their own tools, dashboards, and processes. This lack of integration not only slows down the overall response but also leaves dangerous gaps where vital information can slip through the cracks.

When workflows are fragmented, critical intelligence often fails to reach the right people at the right time. For instance, threat intelligence teams may uncover indicators of compromise (IoCs) that never make it to the analysts quickly enough, while responders might tackle incidents without access to essential vulnerability or asset details. On top of this, SOC managers face the challenge of piecing together a complete, up-to-the-minute view of their team's performance and overall security posture.

SecOps_360 (named to reflect Security Operations delivered around the clock (360 degrees)) is the practical implementation of the unified framework developed through our research. It comes to close these gaps by unifying all SOC functions into a single, fully integrated platform. By centralising operations into one shared environment, it promotes seamless collaboration, speeds up incident resolution, and ensures everyone, from frontline analysts to SOC leadership, works with the same real-time information. The result is not

just improved efficiency and accuracy but also the flexibility to expand and adapt as security demands evolve.

The central research problem of this study is the challenges faced by SOC's in handling malware incidents effectively. Malware incidents pose a significant threat to organisations, and SOC's play a critical role in detecting and responding to these incidents. However, SOC's face several challenges in handling malware incidents, such as the sheer volume of alerts generated by security tools, skills gaps, lack of integration between security tools, increasing sophistication of malware, shortage of qualified personnel, resource constraints, and regulatory compliance.

In the contemporary landscape of cybersecurity, the persistent threat of malware looms large, posing significant challenges to organisations worldwide. SOC's are at the forefront of protecting digital assets by monitoring, detecting, and responding to an array of cybersecurity incidents, with malware incidents being a prominent concern. This research embarks on a comprehensive investigation into the issues surrounding malware incident management, with a focus on addressing four central sub-research questions (SRQs):

SRQ1: What are the common types of malware incidents faced by SOC's?

Malware incidents come in various forms, from traditional viruses to sophisticated ransomware and zero-day exploits. Understanding the common types of malware incidents that SOC's encounter is pivotal to tailoring efficient response strategies and preventive measures. This question forms the foundation of our research by categorising and defining prevalent malware incident types.

SRQ2: What are the challenges faced by SOC's in handling malware incidents?

Managing malware incidents is a complex and dynamic process, and SOC's encounter various challenges in effectively responding to these threats. The challenges may stem from resource limitations, the evolving nature of malware, the scarcity of skilled personnel, or the intricacies of incident response workflows. Identifying and analysing these challenges is crucial to building a holistic understanding of the issues at hand.

SRQ3: What are the effective solutions to mitigate these challenges?

With the ever-increasing sophistication of malware, it is imperative to explore potential solutions that can help SOC's mitigate the challenges they face in handling malware incidents. This part of the research seeks to investigate strategies and technologies such as advanced threat detection systems, threat intelligence sharing, automation, and machine learning to

determine their effectiveness in streamlining incident management processes.

SRQ4: What effective solutions are proposed to enhance the incident response capabilities of SOC?

Identifying potential solutions is only the first step; assessing their impact on SOC incident response capabilities is equally crucial. To provide a comprehensive evaluation, this research will delve into key performance metrics, such as mean time to detect (MTTD) and mean time to respond (MTTR). By quantifying the improvements brought about by the proposed solutions, we aim to offer actionable insights into their efficacy in enhancing SOC incident response capabilities.

This research problem statement underscores the primary objectives of our study. It highlights the importance of understanding common malware incident types, recognising the challenges that impede SOC operations, exploring effective mitigation strategies, and assessing the real-world impact of these solutions on incident response capabilities. The overarching goal is to contribute to the advancement of malware incident management within SOC, thereby strengthening the cybersecurity posture of organisations in the face of a dynamic and persistent threat landscape.

1.3 Research Context

1.3.1 Research Aim and Objectives

The significance of this research lies in its contribution to the field of cybersecurity by identifying the challenges faced by SOC in handling malware incidents and proposing effective solutions to mitigate them. The study will provide insights into the incident response process, the critical role of SOC, and the importance of incident response planning. The findings will be of interest to academics and practitioners in the cybersecurity field, and the practical recommendations will provide a roadmap for organisations to enhance their incident response capabilities.

This research aims to investigate the challenges faced by SOC in managing malware incidents and to develop SecOps_360, a unified platform that brings together separate threat intelligence sources, streamlines analytical workflows, and reduces investigation time and system resource usage for effective malware threat investigation and incident handling in SOC environments. Although several studies have explored the challenges of incident

response, there is a significant gap in the literature on the challenges faced by SOC in handling malware incidents. This research hopes to fill this gap by identifying the common types of malware incidents and the associated challenges and proposing effective solutions to mitigate them.

The relevance and importance of this research lie in its contribution to the existing body of knowledge on malware incident handling in SOC. The study will provide insights into how organisations can better prepare for and respond to these incidents. The proposed solutions will be practical and implementable, taking into account the resource constraints faced by most organisations. They will provide a framework for improving incident response processes, enhancing the skill level of SOC personnel, and optimising the use of security tools. The findings will be of interest to both academics and practitioners in the cybersecurity field.

The key research objectives (ROs) are to:

- RO1 - To study and analyse the state-of-the-art and challenges of malware incident handling and threat investigation in an SOC.
- RO2 - To propose a novel framework to incorporate and optimise threat investigation sources and optimise decision-making for effective malware analysis.
- RO3 - To implement a practical system as a prototype platform, taking into account the resource constraints faced by SOC.
- RO4 - To evaluate the proposed framework by comparing it with traditional approaches using historical data and real-world scenarios.

1.3.2 Scope of the research

The research will focus on malware incident handling in SOC. The study will cover the common types of malware incidents faced by SOC, the challenges encountered in handling these incidents, and effective solutions to mitigate these challenges. The research will be limited to the examination of SOC in the context of medium- and large-sized organisations. The scope of the study is limited to a review of SOC challenges reported by relevant papers. The population of interest for the study includes SOC analysts, cybersecurity professionals, and researchers. The research method is qualitative, using secondary data from relevant papers to gain a better understanding of the existing challenges. The purpose of this study is to provide state-of-the-art challenges of malware investigation in an SOC and identify any gaps or areas for improvement in this field. This information can be used to inform the development of strategies and tactics for effectively dealing with malware threats and

addressing the challenges faced by SOCs.

1.3.3 Ethical Consideration

Ethical conduct underpins all phases of this research. No primary data involving human participants was collected; consequently, the research was not subject to institutional ethics board review for human subjects. Nevertheless, several ethical dimensions deserve explicit attention, in line with the Cybersecurity Ethics (Flechais and Chalhoub (2023)) and the ethical guidance provided in CyBOK (Rashid et al. (2021), Section 3.13).

First, all IoC data used in scenario-based testing (Chapter 6) was drawn from publicly disclosed threat-intelligence datasets and anonymised prior to inclusion in this thesis. No data attributable to specific organisations or individuals was retained or published.

Second, the OSINT APIs integrated within SecOps_360 are consumed strictly within the terms of service of the respective providers (AbuseIPDB, VirusTotal, MetaDefender). The platform does not perform any active scanning of third-party infrastructure, in compliance with the principle of avoiding unauthorised access to computer systems (ACM Code, Section 2.8).

Third, the research acknowledges the potential risks associated with the aggregation of threat-intelligence data within a unified platform. Whilst consolidation improves investigative efficiency, it also creates a single point of access to sensitive intelligence that, if compromised, could expose an organisation's detection capabilities to adversarial actors. This risk is recognised in the cybersecurity ethics literature as the dual-use dilemma, where tools designed to defend can equally be repurposed to cause harm.

Fourth, transparency and honesty in reporting research findings are treated as ethical obligations throughout this thesis, not merely as academic conventions. The evaluation results are reported without embellishment, in keeping with the principle of honest and trustworthy conduct expected of computing professionals. Similarly, the limitations of the controlled experiment, including the small sample size, the absence of a cognitive load measure, and the use of a simplified IoC dataset, are acknowledged explicitly in Chapter 7 rather than minimised. This commitment to transparent self-assessment reflects the broader ethical responsibility.

Finally, the study will ensure that the proposed solutions are ethically sound and do not violate any laws, regulations, or ethical principles. The proposed solutions will prioritise the protection of sensitive data, respect the privacy of individuals and organisations, and ensure that incident response processes are transparent and fair.

1.3.4 Research Contributions

This research makes three primary and independently verifiable contributions to knowledge in the field of Security Operations and Incident Management, as framed by the CyBOK SOIM Knowledge Area (Rashid et al. (2021), Ch. 8).

Primary Contribution 1: The SecOps_360 Framework. This thesis proposes a novel, evidence-derived conceptual framework for unified malware incident management in SOC environments. The framework is distinguished from existing SIEM and Security Orchestration, Automation, and Response (SOAR) platforms in three specific respects:

(a) It is explicitly designed to reduce Indicator-of-Compromise (IoC) investigation latency by aggregating multiple OSINT APIs behind a single query interface, rather than extending alert-correlation or playbook-automation functions already provided by tools such as Splunk SIEM, IBM QRadar, or Palo Alto XSOAR.

(b) Its architecture is derived systematically from a gap analysis of the peer-reviewed literature on SOC operational challenges, making the design choices traceable and reproducible.

(c) It targets the workflow layer (the cognitive and procedural steps performed by human analysts) rather than the detection or orchestration layers already addressed by existing platforms. No comparable platform combining these three properties was identified in the literature reviewed (see Section 2.8 for the criteria-driven comparative analysis).

Primary Contribution 2: Empirical Evidence of Investigation Inefficiency. This research produces the first systematically documented measurement of resource utilisation and elapsed time during multi-platform, manual IoC investigation in a simulated SOC environment. The baseline measurements (cumulative search time of 96.6 seconds, CPU utilisation of 85%, and memory consumption of 92% during concurrent searches) establish a quantified baseline that justifies the integration approach and provides a foundation for future comparative studies.

Primary Contribution 3: The SecOps_360 Prototype Platform and Empirical Validation. The third primary contribution is the fully functional web-based implementation of the SecOps_360 framework, together with the empirical validation (reported in full in Section 6.6) of its performance through a controlled experiment conducted with practising SOC analysts (one Tier 1, two Tier 2). Each analyst completed a standardised three-IoC investigation task under both the manual multi-platform condition and the SecOps_360-assisted condition, on their personal laptops. The results demonstrated a mean investigation time

reduction of 84.3% and a mean peak CPU reduction of 34.3 percentage points. The Wilcoxon Signed-Rank Test yielded $W = 0$ for both dependent variables (the most extreme possible outcome for $n = 3$) with an exact one-tailed p value of 0.125, reported transparently, and a very large effect size of Cohen's $d = 6.08$. The platform's REST API architecture and fully documented source code make it reproducible and extendable by future researchers.

Secondary contributions include: a Systematic Literature Review (SLR) synthesising the state of the art on SOC malware incident management; a criteria-driven comparative analysis of existing industry tools (Section 2.8); a structured scenario-based evaluation methodology applicable to future platform evaluations; and a set of proposed empirical validation experiments (Section 6.6) that constitute a roadmap for future rigorous experimental work.

1.4 Thesis Structure

This thesis is comprised of seven chapters described as follows. Chapter 1 introduces the topic of the thesis and its background. It identifies the research problem and delves into the research context, outlining the aim and objectives, scope, ethical considerations, and expected outcomes. Additionally, the chapter identifies the research gap and explains the relationship between the sub-research questions (SRQs) and the included publications. Chapter 2 provides a critical and comprehensive analysis of existing literature related to our research area. This literature review aims to determine a strong groundwork for our research by thoroughly investigating the challenges in SOC, delving into solutions to address the defined challenges and proposing future research directions based on the gaps identified in the existing literature. Chapter 3 provides the process and methodologies applied during our scientific inquiry into this thesis and serves as a thorough guide that offers clarity into the research process and ascertains the reliability and validity of the study's outcomes.

Chapter 4 lays the groundwork for the following phases of the research. It gives a clear understanding of the theoretical foundations through conceptual modelling and then turns those concepts into a tangible system design. The conceptual design is important, as it gives shape to the development and implementation phases by making sure that the research lines up with the determined theoretical framework. Chapter 5 is the implementation phase, which focuses on the practical development of the conceptual design. It illustrates how the conceptual design is converted into a working system, acting as a link between the theoretical and practical aspects of the study. It provides an understanding of the technical facets of

system development by exhibiting the abilities and techniques used in the implementation stage. The final product's presentation offers a concrete depiction of the research findings and establishes the framework for the phases of evaluation and analysis that follow.

Chapter 6 represents a pivotal point in the study, where the deployed system is thoroughly analysed and assessed. Decisions regarding system improvement and possible future additions are influenced by the results of testing and assessment. The section on immediate future work provides the basis for continued development and guarantees that the system keeps changing in response to input and needs from the real world. Chapter 7 provides a thorough overview of the study's impact on both theoretical and practical aspects within the field of study, serving to summarise and highlight the significance of the research contributions. It also opens the way for additional investigation and improvement in future studies.

Chapter 2

Literature Review

2.1 Introduction

This chapter presents a Systematic Literature Review (SLR) of the current state of knowledge on malware incident management within SOC environments. The review follows a structured protocol informed by the PRISMA 2020 Page et al. (2021) guidelines for reporting systematic reviews, Kitchenham et al. (2022) SEGRESS guidelines for secondary studies in software engineering and computing, and Carrera-Rivera et al. (2022) practical guide for conducting SLRs in computer science research. The methodology governing the review (including search strategy, inclusion and exclusion criteria, and quality assessment) is described in Section 2.2 before the substantive findings are presented.

The chapter is structured as follows. Section 2.2 describes the SLR methodology. While section 2.3 inspects the terminology and theoretical foundations, Section 2.4 examines the SOC incident management lifecycle. Section 2.5 reviews SOC architectural components. Section 2.6 analyses malware detection and analysis techniques. Section 2.7 identifies and critically evaluates the challenges facing SOCs. Section 2.8 presents a criteria-driven comparison of existing industry tools and platforms. Section 2.9 synthesises the research gap. Section 2.10 summarises the chapter.

Cybercrimes are occurring rapidly and use complex techniques that require organisations to think ahead and adapt their threat responses in a shorter time. Due to the urgency of time, faster decision-making is applied using limited or unreliable information to determine instant actions for threat resolution Cho et al. (2020). The need to control the damage caused by the evolving threats demands fast investigations using appropriate and effective tools and technologies Dhaka and Johari (2016). As a security defence entity, a security operation centre (SOC) is a team of security professionals who protect networks constantly, against cyber-attacks Oesch (2020). They use security information and event management (SIEM) tools, news feeds, and open-source intelligence to develop their awareness of the latest cybersecurity developments in the SOC Alves et al. (2021). Security professionals examine potential malicious programmes to determine their harmful intent and classify them according to their functionality, such as viruses, spyware, Trojan horses, worms, ransomware, etc. Moser et al. (2007a). An SOC with the computer security incident response team (CSIRT) provides security operations services and incident management in proactive and reactive manners Jacobs et al. (2013). To prevent threats from exploiting system flaws, various types of threat modelling techniques can be used to defend against anticipated threats Jacobs et al. (2013). The preventive methods established by a SOC are responsible for safeguarding vital information, planning and creating contingency procedures, as well as identifying and responding to attacks. In short, conducting malware investigations in an SOC environment does not go without challenges Alharbi (2020). This chapter aims to carry out a literature

review of malware incident handling and related challenges in SOCs.

2.2 Systematic Literature Review Methodology

2.2.1 Review Protocol and Objectives

The SLR was conducted to address RO1: to study and analyse the state of the art and challenges of malware incident handling and threat investigation in SOC environments. A formal review protocol was defined and documented prior to any search activity, in accordance with established best practice for systematic reviews in computing and software engineering research (Kitchenham et al. (2022)). The protocol specified, before any database was searched, the review objectives, the databases to be consulted, the search string to be applied, the inclusion and exclusion criteria by which candidate papers would be assessed, the quality assessment instrument to be used, and the data extraction form through which findings would be recorded. The overarching objective of the review was to produce a systematically derived, critically assessed synthesis of the current state of knowledge on SOC malware incident management, sufficient to characterise the principal operational challenges, evaluate the adequacy of proposed mitigations in the existing literature, and identify the specific research gaps that justify the design and development of SecOps_360.

2.2.2 Search Strategy

Five electronic databases were searched: IEEE Xplore, ACM Digital Library, ScienceDirect, Scopus, and Google Scholar. In addition, the websites of major cybersecurity vendors (Gartner, SANS Institute, MITRE ATT&CK, NIST) were manually searched for grey literature. The search was conducted between January 2023 and September 2024, with no lower date bound applied to capture foundational literature.

The following Boolean search string was used: ("Security Operations Centre" OR "SOC" OR "Security Operations Centre") AND ("malware" OR "malicious software" OR "ransomware" OR "trojan" OR "APT") AND ("incident management" OR "incident response" OR "incident handling" OR "threat investigation"). Supplementary searches were conducted using the terms "SIEM challenges", "SOAR limitations", "SOC analyst workload", and "threat intelligence integration".

2.2.3 Inclusion and Exclusion

Criteria Inclusion criteria: peer-reviewed journal articles and conference papers; grey literature from authoritative cybersecurity bodies (NIST, MITRE, SANS, CyBOK); papers in English; papers addressing SOC operations, malware analysis, or incident response; papers published from 2010 onwards (with exceptions for seminal earlier work). Exclusion criteria: papers not directly addressing SOC or incident management contexts; papers focused exclusively on network intrusion detection without relevance to malware analysis; duplicate publications.

2.2.4 Study Selection and Quality Assessment

The initial database search returned 847 candidate papers. After removal of duplicates (n=134), titles and abstracts were screened against the inclusion criteria, yielding 203 papers for full-text review. Of these, 89 were retained following quality assessment. Quality was assessed using a checklist adapted from Kitchenham et al. (2022), scoring papers on: clarity of research objectives; appropriateness of methodology; validity of results; and relevance to the research questions. Papers scoring below a threshold of 60% were excluded.

2.2.5 Synthesis Approach

Data were extracted using a standardised extraction form capturing: authors, year, research question addressed, methodology, key findings, limitations noted by authors, and relevance to each SRQ. A thematic synthesis approach was adopted, grouping findings into the categories of SOC challenges, detection and analysis techniques, and recommended mitigations. Where findings conflicted across papers, both positions are reported and the weight of evidence assessed.

2.3 Background: Terminology and Theoretical Foundations

This review adopts the terminology established in the Cyber Security Body of Knowledge CyBOK v1.1 (Rashid et al. (2021)) as the standard reference for cybersecurity concepts. Key terms used throughout this thesis are defined as follows, with CyBOK chapter references provided.

Security Operations and Incident Management (SOIM): The organisational and technical function responsible for monitoring an ICT infrastructure, detecting security events, analysing their nature and severity, and executing appropriate responses. CyBOK models SOIM as a MAPE-K loop (Monitor, Analyse, Plan, Execute, with a Knowledge base) instantiated in technical components deployed within a SOC (Rashid et al. (2021) Section 8.1.1).

Security Operations Centre (SOC): The organisational entity that consolidates the people, processes, and technologies required to perform SOIM functions (Rashid et al. (2021) , Section 8.1.2). The SOC provides both technological and human resources, centred around a SIEM platform.

Security Information and Event Management (SIEM): A platform that aggregates log data from diverse sources, correlates security events, and supports the Plan phase of the MAPE-K loop. CyBOK identifies SIEM as the core technological component of the SOC (Rashid et al. (2021), Section 8.4).

Security Orchestration, Automation and Response (SOAR): An extension of SIEM that enables partial automation of response actions (the Execute phase of MAPE-K) and supports risk and incident management at scale (Rashid et al. (2021), Section 8.5.4). SOAR platforms drive advanced analytics and global responses to cyber threats.

Malware: CyBOK (Ch. 6) defines malware as software intentionally designed to cause disruption, gain unauthorised access, or steal information. The CyBOK malware taxonomy distinguishes: viruses (self-replicating code requiring a host programme); worms (self-propagating, network-borne); Trojan horses (code masquerading as benign software); ransomware (malware that encrypts victim data and demands payment); spyware (malware designed for covert surveillance); rootkits (malware providing persistent privileged access); and botnets (networks of malware-infected hosts under centralised control). Advanced Persistent Threat (APT) actors often combine multiple malware types in a campaign.

Indicator of Compromise (IoC): An artefact observed on a network or host that indicates a security incident with high probability. Common IoC types include IP addresses associated with command-and-control (C2) infrastructure, file hashes (MD5, SHA-1, SHA-256) of malicious executables, domain names used for C2 or phishing, and URL patterns. IoCs are central to the threat-intelligence sharing and analysis functions of a SOC (Rashid et al. (2021), Section 8.6.3).

Cyber Threat Intelligence (CTI): Evidence-based knowledge about existing or emerging threats that can be used to inform decisions about how to respond to them. CyBOK locates

CTI within the Knowledge component of the MAPE-K loop (Rashid et al. (2021), Section 8.6.3). OSINT (Open Source Intelligence derived from publicly available sources) is a key component of CTI used in SOC investigations.

Malware incident handling refers to detecting, analysing, and responding to malware threats. SOCs play a key role in monitoring networks for malicious activity, identifying and containing malware outbreaks, and recovering from security incidents. Our research provides context by identifying and describing the specific problem of the prevalence and impact of malware incidents in SOC environments. The study explains how malware incidents can cause disruptions to the normal functioning of a SOC and lead to security breaches, data loss, and financial losses. The study also describes the various challenges SOC teams face in detecting, analysing, and responding to malware incidents. The study provides insights into how to improve the efficiency and effectiveness of SOCs in detecting and responding to malware incidents. It also explores new technologies and best practices for preventing and mitigating the impact of malware incidents in SOC environments. Overall, the research contributes to the larger body of knowledge in the field by offering a comprehensive understanding of the challenges of malware incidents in SOCs.

Additionally, the research explores the role of Cyber professionals and examines the effectiveness of incident response plans and incident management processes in dealing with malware incidents. Furthermore, the research considers the role of human factors, such as the role of user education and awareness, in preventing malware infections. It examines the effectiveness of the technologies in identifying and mitigating malware threats and the limitations of these technologies. Related work in this field typically includes research on malware detection and classification techniques, incident response frameworks and best practices, and the use of artificial intelligence and machine learning in malware incident handling.

The scope of review encompasses the time frame of papers reviewed, databases searched, keywords and search terms used, inclusion and exclusion criteria applied, number of papers identified and selected, and the specific focus areas covered related to SOC operations, malware incident management, and threat intelligence platforms.

In summary, the research on malware incidents and challenges in a SOC provides context by identifying and describing the specific problem of the prevalence and impact of malware incidents in SOC environments and how it fits into the larger body of knowledge by addressing gaps in the current understanding, offering new insights and solutions, and situating the study within the broader context of cybersecurity research and practice.

2.4 SOC Incident Management Lifecycle

Incident management in SOC environments follows a structured lifecycle. The NIST Computer Security Incident Handling Guide (Cichonski et al. (2012), Rev. 2) defines four phases: Preparation; Detection and Analysis; Containment, Eradication and Recovery; and Post-Incident Activity. CyBOK (Section 8.7) maps this to three phases within the SOIM context: Prepare, Handle, and Follow-up. This thesis uses the more granular NIST framework, as it is the most widely adopted in the literature reviewed.

SOC typically has a set of operational guides, local working instructions, procedures, and operations-level policies in place to ensure that all SOC processes are followed consistently and effectively Onwubiko and Ouazzane (2022a), Onwubiko (2022). These documents provide detailed guidance on how to perform various tasks and procedures within the SOC, such as incident detection and response, threat intelligence analysis, and vulnerability management. They also outline the roles and responsibilities of SOC personnel and guide how to comply with industry regulations and standards. Additionally, these guides and procedures are often reviewed and updated regularly to ensure that they reflect the latest best practices and to adapt to changing security threats and regulations.

Incident management in SOC is a continuous process of detecting, assessing, responding, and recovering from security incidents and using lessons learnt to improve the incident management process Miloslavskaya (2016). This enables organisations to respond effectively to security incidents, minimise their impact, and return to normal operations as quickly as possible. The response process to malware incidents (Akinrolabu et al., 2018; Kolokotronis and Shiaeles, 2021; Wael and Azer, 2018) is extremely challenging, and organisations are increasingly being targeted by malicious malware attacks. There are several models of incident management depending on the organisation (Bronk et al., 2006; Cichonski et al., 2012; US-CERT, 2013), however, the process (Ab Rahman, 2016; Jaramillo, 2019; Souppaya et al., 2013) may encompass preparation, detection, analysis, containment, eradication, recovery, and post-incident or identification, protection, detection, response, and recovery (Barrett, 2018; Thompson, 2018). This shows the variety of terminologies used to define the process of incident management.

2.4.1 Preparation

Preparation encompasses the establishment of an incident response capability, the creation of playbooks and Standard Operating Procedures (SOPs), and the training of personnel.

The literature identifies preparation as the phase most commonly neglected, with many SOC's lacking documented playbooks for malware-specific scenarios (Vielberth et al. (2020); Naseer et al. (2023)).

According to Onwubiko and Ouazzane's Onwubiko and Ouazzane (2020) research, having a well-prepared SOC is critical to minimising the effects of security incidents and underlines the need to plan when developing an efficient incident response strategy. A strong SOC incident management approach is built on preparation. The creation of a proactive framework that enables the organisation to react quickly and effectively to possible security events is what defines this phase. Critical asset identification, risk assessments, and the creation of an extensive incident response plan are important elements of this phase. Organisations carry out extensive risk assessments during the planning phase to pinpoint any weak points and rank assets that are essential to their operations. This entails being aware of how security events could affect data integrity, company operations, and overall organisational resilience. An illustrative example would be a banking institution that recognises client databases and transactional systems as essential assets during the preparatory stage. Using risk assessments and scenario-based exercises, the SOC builds an in-depth understanding of potential vulnerabilities and threats. The institution may create focused incident response plans that are adapted to particular risks thanks to this proactive strategy, which guarantees that it is prepared for a variety of cyber threats. Organisations create and maintain incident response plans, outlining roles, responsibilities, and communication protocols, as a means of laying the groundwork for incident response. Simulations and tabletop exercises are used to confirm the effectiveness of these strategies and improve the team's capacity to function as a unit under pressure.

The preparation stage reflects the preparatory measures put in place by an organisation to respond effectively to malware incidents. These measures include obtaining necessary tools and resources, developing and retaining malware-related skills within the incident response team, and enabling communication and coordination in the organisation Souppaya et al. (2013). This is also known as the "identify and protect" phase Barrett (2018); Thompson (2018), where the security risks related to the systems, assets, people, data, and capabilities are identified and managed. They protect by developing suitable safeguards to guarantee the delivery of critical services.

2.4.2 Detection and Analysis

Detection relies on data sources described in CyBOK Section 8.2, including network traffic analysis, system and kernel logs, application logs, and threat intelligence feeds. Analysis methods include misuse detection and anomaly detection (Rashid et al. (2021), Sections 8.3.1–8.3.2). A persistent challenge is the base-rate fallacy (Rashid et al. (2021), Section 8.3.6): because malicious events are rare relative to benign traffic, even classifiers with high precision generate many false positives, imposing significant triage workload on analysts.

A SOC incident management structure is essentially composed of detection and analysis. As to Velberth et al. (2020), the SOC keeps a close eye on the network to identify any possible security events as they arise. To detect any indications of potential security risks, this calls for ongoing observation. Then, during the analysis phase, SOC analysts carry out in-depth investigations to ascertain the type, extent, and attribution of the observed incidents. They do this by utilising forensic techniques, threat intelligence feeds, and cooperative knowledge-sharing. To create a response plan that is specific to each incident, the SOC must investigate the details of each occurrence to determine the attribution, impact, and root cause Danquah (2020).

Advanced tools like Intrusion Detection Systems (IDS) and Security Information and Event Management (SIEM) solutions are used in SOC incident management for detection and analysis. These systems monitor endpoint activity, system logs, and network traffic to spot suspicious patterns that might indicate violations of security Pawlicka et al. (2021). To improve security, it is advised to employ technical mitigation techniques, including frequent backups, firewall installation, network segmentation, and user access control based on the least privilege principle Nifakos et al. (2021).

An example to illustrate the detection could be a situation where the SIEM solution of a SOC notices an abrupt increase in unsuccessful login attempts across many user accounts. The SOC analysts are immediately alerted to this anomalous pattern and are prompted to investigate it further. The SOC detects a brute-force attack that aims to gain unauthorised access to the organisation's systems through real-time analysis. In this instance, detection makes it possible for the SOC to act quickly and stop possible data breaches.

The complementary element of the analysis explores the specifics of the incidents that have been detected. SOC analysts use collaborative knowledge-sharing, threat intelligence feeds, and forensic methodologies to determine the incident's attribution, impact, and cause. This in-depth analysis improves the SOC's capacity to design an accurate and successful response plan Agyepong et al. (2020).

Detection and analysis are the efforts every SOC organisation has to make to detect and confirm malware incidents rapidly to reduce the number of infected hosts and the amount of damage a business sustains. Sometimes, it takes significant analysis, involving large technical knowledge and experience, to realise that an incident has been caused by malware. The detection and analysis stages involve identifying the incident characteristics (malware category, ports, protocols, exploited vulnerabilities, malicious filenames, etc.), identifying the infected hosts (from a network device, DNS, application server logs, IPS/IDS sensors, manually), engaging incident response, and investigating malware Souppaya et al. (2013).

2.4.3 Containment

Containment limits the spread of a malware incident. Network-level containment may involve isolation of affected hosts or network segments. Host-level containment may involve termination of malicious processes or quarantine of affected files. The timely execution of containment actions is strongly influenced by the speed of the Detection and Analysis phase.

The containment stage is the quick and planned reaction to detected security events to minimise the damage and isolate the affected areas, such as network segmentation, blocking malicious communication channels, and isolating infected devices to stop more intrusions and harm to vital resources. By containing the incident promptly, the SOC creates a controlled environment for subsequent eradication efforts Oriola et al. (2021). The significance of coordinated efforts in efficiently addressing cybersecurity issues is emphasised by this collaborative approach. Furthermore, the CyberSecurity Optimisation and Training project for Enhanced Resilience in Finance (SOTER) provides a playbook for cybersecurity incident management to serve as a thorough framework for managing threats, especially in a SOC environment Onwubiko and Ouazzane (2022b). This playbook is intended to improve incident management procedures and boost the SOC response process. Bennett and Robertson's study highlights the significance of prompt containment in the event of a security problem. It indicates that to reduce downtime and stop the issue from spreading to other systems, reaction teams must act quickly to limit the situation, identify the problem, and restore the system to a previous secure condition Bennett and Robertson (2023). Bennett and Robertson's study highlights the significance of prompt containment in the event of a security incident. It indicates that to reduce downtime and stop the issue from spreading to other systems, reaction teams must act quickly to limit the situation, identify the problem, and restore the system to a previous secure condition Bennett and Robertson (2023). A descriptive case is to think of a scenario where an SOC detects a ransomware infection that spreads quickly through file servers. As a quick reaction, the SOC removes the compromised servers from the network,

stopping the ransomware from encrypting more files and reducing the damage overall. This containment method gives the SOC a contained environment in which to perform additional analysis, in addition to safeguarding sensitive data.

2.4.4 Eradication

Eradication involves the complete removal of malware artefacts, including all persistence mechanisms (registry entries, scheduled tasks, modified binaries). Eradication is critically dependent on the quality of the preceding analysis: partial malware understanding leads to incomplete eradication and reinfection.

Eradication refers to a deliberate attempt to fix the vulnerabilities that a threat exploits, as opposed to merely treating the visible indications of a security incident. Preventing chronic threats and strengthening the company's long-term resilience are the main goals of this phase. To guarantee an effective eradication procedure, SOC analysts use threat intelligence, forensic analysis, and updated security protocols Agyepong et al. (2020). Targeted efforts are made to remove malicious components, patch vulnerabilities, and fortify defences against future recurrences during the Eradication phase. By removing vulnerabilities that potential attackers could exploit, this proactive strategy not only lessens the immediate effect of incidents but also improves the company's security posture as a whole. To further improve the business's capacity to stop recurring incidents and strengthen its long-term security resilience, updated security protocols are essential in ensuring that eradication efforts are thorough and successful Reeves and Ashenden (2023).

As an example, we can imagine a situation where a malware infestation is successfully contained within an SOC. During the Eradication phase, a thorough analysis is carried out by the SOC to pinpoint the exact malware strain that is at fault. Equipped with this data, the SOC applies system patches, upgrades antivirus signatures, and implements extra security steps to eradicate the threat from all impacted computers. By being proactive, the chance of reinfection is decreased, and a complete eradication is ensured.

2.4.5 Recovery

Recovery restores affected systems to normal operation, validates the absence of remaining malware artefacts, and monitors for recurrence. Recovery timelines are substantially influenced by preparation: organisations with tested backup and recovery procedures recover significantly faster (Naseer et al. (2023)).

This stage entails methodical attempts to restore data, verify system integrity, and incorporate acquired knowledge to improve the cybersecurity posture of the company as a whole. To effectively manage crises, research by Mahmood Mahmood et al. (2024) highlights the importance of putting cybersecurity awareness programmes into place, supplying cyber assistance, changing roles and responsibilities, and embracing new technology. In addition, Neri et al. (2023) underline the need for organisational anticipation in the fight against cyber attacks and point out areas that need to be improved for increased resilience. Danquah (2020) points out the significance of effectively managing security threats, vulnerabilities, and incidents during the recovery phase and offers a framework for automated triage, containment, and escalation in SOC operations. Furthermore, Datta et al. (2019) emphasise the need for adaptive techniques for SOC recovery, demonstrating the flexibility of Battery Energy Storage Systems (BESS) in efficiently controlling SOC operations based on projected future events. Sawalha (2021) makes a distinction between Business Continuity Planning (BCP) and Disaster Recovery Planning (DRP) based on the recovery phase. BCP relates to the stages of mitigation, preparedness, and responses, whereas DRP involves the post-incident recovery phase. This contrast highlights how important recovery operations are to maintaining resilience and business continuity. To maintain data availability and integrity, it is important to guarantee the recovery and restoration of critical information from backup systems. Thorough preparation and regular testing are essential during this procedure to ensure that important data may be successfully recovered. To ensure that the restored systems run effectively and securely, further system validation procedures are put in place (Vielberth et al. (2020)). As an example, we can assume that an SOC successfully removes a ransomware attack. During the Recovery phase, encrypted files are restored from routinely validated and verified backups by the SOC. To verify the accuracy of the recovered data and guarantee that the organisation can quickly return to regular operations, system validation checks are carried out.

2.4.6 Post-Incident Activity (Lessons Learned)

Post-incident activities include root cause analysis, documentation of the incident timeline and response actions, identification of process improvements, and updating of playbooks. CyBOK (Section 8.7.3) identifies lessons-learned processes as a critical element of the Follow-up phase. The literature consistently identifies post-incident documentation as one of the most under-resourced activities in operational SOC's (Agyepong et al. (2023)).

During this stage, the SOC determines what needs to be improved and assesses how well the response tactics used during the incident worked. The incident's lessons are carefully recorded and used as a basis for improving incident response strategies, updating security

protocols, and enhancing the company's overall cybersecurity resilience. The SOC can improve its readiness for upcoming events and reduce the likelihood of such security breaches by utilising the knowledge acquired from these lessons Oriola et al. (2021). Knowing how to identify valuable knowledge from incident reports is critical to the continuous improvement process. Managers can prevent recurring problems by using this strategy to proactively detect trends and commonalities. Additionally, by incorporating these insights into incident response protocols, the SOC can expect to maintain its flexibility and adaptability in dealing with cybersecurity threats Oriola et al. (2021).

Several academic papers have made contributions to incident management 2.1 by putting forward frameworks, guidelines, and analyses aimed at improving the practices in the field. However, a more intensive examination exposes some criticisms that can guide future studies. A cybersecurity incident management framework called "SOTER" by Onwubiko et al Onwubiko and Ouazzane (2020) is one of the papers being reviewed. Although it provides an organised method for incident management, the SOTER framework is not supported by empirical research. Its practical utility might be restricted in the absence of proof of its efficacy in real-world situations. To confirm the framework's effectiveness and build its credibility, future research might concentrate on running case studies or simulations. The "Guide to Malware Incident Prevention and Handling" by Souppaya and Scarfone Souppaya et al. (2013) offers insightful information on malware incident handling. However, given how quickly cyber risks are evolving, the guide's advice might not be applicable today. The guide's relevance and practicality for practitioners would increase if it were updated to take into account the latest developments in malware and mitigation methods. Barrett and colleagues Barrett (2018) present a methodology for creating cyber resilience metrics, providing a thorough method of gauging cyber resilience. Nevertheless, the absence of useful implementation advice in the paper makes it difficult for businesses to use the framework efficiently. Its popularity and usefulness would increase if comprehensive guidelines and examples of how to apply and evaluate resilience measures were included. Although Thompson Thompson (2018) provides a thorough analysis of cybersecurity incident management, there may be an excessive amount of theoretical content in his work. The study might benefit from including additional real-world case studies and helpful advice for organisations, even though the theoretical talks are valuable. This would improve the paper's effectiveness for practitioners by bridging the theory-practice divide. While Danquah et al. Danquah (2020) evaluate some security incident management techniques, they might not include a formal comparison study. Practitioners could choose the method that best fits their organisation by weighing the advantages and disadvantages of various strategies. To close this gap, future work may concentrate on comparing incident management techniques in an organised

manner. Although they might not seem very generalisable, Nifakos et al. (2021) investigate how organisational culture affects cybersecurity incident handling. Restricting the findings' applicability to a particular workplace setting may restrict their usefulness. It would improve the research findings' generalisation and relevance by including case studies from a variety of sectors. While Agyepong et al. (2020) offer a thorough framework for dealing with incidents in critical infrastructure, there might not be enough involvement from stakeholders. Stakeholder involvement in the development process would guarantee the framework's applicability and relevance in many industries. Subsequent investigations may concentrate on obtaining input from relevant parties to improve and validate the framework. Lastly, Oriola et al. (2021) present an approach to the collaborative handling of cybersecurity emergencies, yet it might not include sufficient guidance for real-world applications. Companies would be more prepared to implement the framework and improve crisis management if they were given practical solutions for collaboration.

2.5 SOC Architecture: People, Technology, and Processes

SOC acts as a central hub for identifying, evaluating, and handling security events in the digital environments of businesses. It is difficult to excessively emphasise the importance of a well-designed SOC architecture, considering the ongoing evolution of cyber threats. This chapter explores the interaction between people, procedures, and technology in strengthening defences against cyber threats, delving into the many capabilities and components that make up a good SOC architecture. Human capital, which includes knowledgeable analysts, incident responders, and security specialists, is the foundation of any SOC architecture. As the first line of defence, these people use their knowledge and experience to identify suspicious behaviours, investigate potential threats, and prepare swift reactions to security issues. Established processes and protocols, when combined with human resources, provide the foundation of SOC operations, offering a well-organised structure for incident detection, analysis, and mitigation. The chapter also examines how important technology is to the SOC design, covering everything from automation tools and orchestration solutions to sophisticated threat detection systems and security analytics platforms. This chapter sheds light on the essential ideas and best practices behind the design of an effective SOC architecture in the dynamic cybersecurity context of today.

The three-pillar model of SOC architecture (People, Technology, and Processes) is well established in the literature and maps directly onto the CyBOK SOIM framework.

Stage	Description	Literature
Preparation	Creating the Groundwork for SOC Incident Management	Onwubiko and Ouazzane (2020), Souppaya et al. (2013), Barrett (2018), Thompson (2018)
Detection and Analysis	Detecting Cyberthreats Within the SOC	Vielberth et al. (2020), Danquah (2020), Pawlicka et al. (2021), Nifakos et al. (2021), Agyepong et al. (2020), Souppaya et al. (2013)
Containment	Limiting and Controlling a Situation	Oriola et al. (2021), Onwubiko and Ouazzane (2022b), Bennett and Robertson (2023), Bennett and Robertson (2023)
Eradication	Total Elimination of the Threat from the Core	Agyepong et al. (2020), Reeves and Ashenden (2023)
Recovery	Rebuilding Regular Operations and Cyber Resilience	Mahmood et al. (2024), Neri et al. (2023), Danquah (2020), Datta et al. (2019), Sawalha (2021), Vielberth et al. (2020)
Lessons Learnt	Recovery to Standard Operations After Cyber Incident	Oriola et al. (2021), Oriola et al. (2021)
Total	-	24

Table 2.1 List of reviewed Literature for Malware Incident Management

2.5.1 People

SOC teams typically comprise Tier 1 Analysts (alert triage), Tier 2 Analysts (incident investigation), Tier 3 Analysts/Threat Hunters (advanced analysis and proactive hunting), Incident Responders, Forensic Analysts, Threat Intelligence Analysts, and a SOC Manager. Skills gaps at all tiers are consistently identified as a primary operational challenge (Vielberth et al. (2020); Forsberg and Frantti (2023)).

SOC teams are responsible for detecting, addressing support tickets, implementing, configuring, and managing their security infrastructure. People play an important role in the security of businesses. From the analyst to the manager, a variety of roles can be identified, whereby an SOC must take care of staffing and recruitment Dietz et al. (2020). SOC people can be split into two distinct categories (Fig 2.1): cyber onboarding people and SOC monitoring and incident management personnel Onwubiko and Ouazzane (2022a), as shown in 2.1.

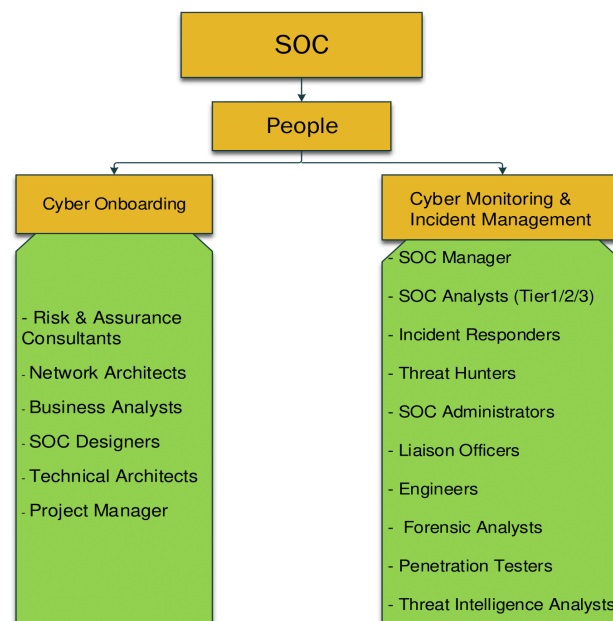


Fig. 2.1 Architecture of SOC in terms of People

A - Cyber onboarding personnel

Cyber onboarding personnel are responsible for training and equipping new employees or contractors to perform their duties within the SOC. They ensure that new employees understand the SOC's processes, procedures, and tools and provide them with the necessary access to perform their roles. They carry out project-related activities to ensure that each business service to be monitored is properly onboarded to a SOC monitoring and incident

management platform. Cyber onboarding is a multidisciplinary team comprised of SOC designers, business analysts, content engineers, solutions and technical architects, project managers, and risk and information assurance consultants.

B - SOC monitoring and incident management personnel

They are responsible for monitoring the onboarded services for potential threats and responding to actual incidents, cyber incident management, vulnerability scanning, incident triage, alerting and event analysis, coordination, and reporting. They also coordinate incident response efforts, which include identifying and mitigating threats, recovering from security breaches, and communicating with other teams within the organisation. The SOC monitoring and incident management team is composed of personnel such as the SOC manager, SOC analysts, SOC engineers, incident responders, security architects, threat hunters, compliance officers, security awareness and training officers, forensic analysts, penetration testers, threat intelligence analysts, SOC administrators, liaison officers, etc. Kowtha, S. et al (2012); Onwubiko (2015); Shahjee and Ware (2022); Tafazzoli (2016); Torres (2015). The staff roles and responsibilities within an SOC can vary depending on the organisation and the specific security needs of the organisation Basyurt et al. (2022); Muniz et al. (2015). Some common roles and responsibilities include:

SOC Manager: The SOC manager is responsible for overseeing the overall operations of the SOC, including setting goals and objectives, developing policies and procedures, and managing the budget.

SOC Analyst: SOC analysts are responsible for monitoring the organisation's network, servers, and endpoints for signs of malicious activity or security breaches. They use various tools and techniques to detect and analyse potential threats, such as intrusion detection systems, security information and event management (SIEM) systems, and threat intelligence feeds. SOC analysts are typically organised into a tiered structure. Each tier of SOC analysts is responsible for a specific set of tasks and has a different level of expertise. The tiered structure allows SOC teams to efficiently triage and respond to incidents and allows for the escalation of incidents to the appropriate level of expertise. There are typically three tiers of SOC analysts Agyepong et al. (2023); Skendžić et al. (2022); Vaarandi and Mäses (2022) including *Tier 1*: the first line of defence responsible for triaging and escalating incidents to higher tiers; *Tier 2*: responsible for investigating and analysing incidents that have been escalated from Tier 1, and *Tier 3*: the most senior and experienced members of the SOC responsible for performing in-depth analysis of incidents, implementing incident response plans, and coordinating incident response efforts with other teams within the organisation.

SOC Engineer: SOC engineers are responsible for maintaining and troubleshooting the tools and systems used by the SOC, including firewalls, intrusion detection systems, and security information and event management (SIEM) systems.

Incident Responder: Incident responders are responsible for containing and mitigating threats, as well as investigating and analysing incidents to determine the extent of the damage and the attackers' objectives.

Threat Hunter: Threat hunters are responsible for proactively searching for potential threats and vulnerabilities within the organisation's systems and networks.

Security Architect: Security Architects are responsible for designing and implementing security measures that align with the organisation's overall security strategy.

Compliance Officer: Compliance Officers are responsible for ensuring that the organisation is meeting regulatory requirements and compliance standards related to cybersecurity.

Security Awareness and Training Officer: Security awareness and training officers are responsible for educating employees and other stakeholders on cybersecurity best practices and providing training on how to recognise and respond to potential threats.

It's worth mentioning that some SOC's may also have additional staff roles depending on the size and complexity of the organisations Basyurt et al. (2022); Muniz et al. (2015). Additional staff roles include:

Forensics Analyst: Forensics analysts are responsible for collecting and analysing digital evidence from incidents and investigations. They use specialised tools and techniques to extract data from computers and other devices and analyse that data to determine what happened during an incident and who was responsible.

Penetration Tester: Penetration testers are responsible for simulating real-world attacks on the organisation's systems and networks. They use various tools and techniques to identify vulnerabilities and test the effectiveness of the organisation's security controls.

Threat Intelligence Analyst: Threat intelligence analysts are responsible for researching and analysing threat intelligence feeds and other data sources to identify emerging threats and trends.

Operations Support: Operations support staff are responsible for providing administrative and logistical support to the SOC, including managing equipment and supplies, scheduling and coordinating training, and maintaining documentation and records.

Data Privacy Officer: Data Privacy Officers are responsible for ensuring that the organisation's data privacy policies and regulations are being adhered to and that data is being protected from breaches.

Legal Officer: Legal Officers are responsible for providing legal advice and guidance to the SOC on issues such as data privacy, incident response, and compliance with regulations.

Business Continuity Officer: Business Continuity Officers are responsible for ensuring that the organisation is prepared for and can recover from disruptions, including cyber incidents, natural disasters, and other emergencies.

The identified literature for SOC people suggests that there is a significant emphasis on understanding the specific duties and expectations of SOC personnel in the literature. However, it is important to underline areas where improvement may be needed. Critiques of selected papers that focus on SOC people architecture shed light on areas for improvement and suggest ways to enhance the effectiveness of SOC personnel. Initially, an integrated paradigm for SOC personnel management was proposed by Shahjee et al. Shahjee and Ware (2022), yet there is no empirical support for its efficacy. Although the paradigm might provide theoretical understanding, its applicability in real-world SOC contexts has yet to be demonstrated. Onwubiko et al. Onwubiko (2015) also discuss the infrastructure of Cyber-Security Operations Centres (CSOCs), but they don't provide a thorough investigation of human elements. An in-depth analysis of SOC staff responsibilities and training requirements would enhance the conversation and offer a comprehensive perspective on CSOC efficacy. Effective human management is overlooked in Torres et al. Torres (2015)'s primary focus on the technical components of constructing an SOC. To help SOC managers with real-world advice, the paper might use some insights into hiring, developing, and keeping SOC staff. Similarly, Kowtha et al. Kowtha, S. et al (2012) describe the architecture of CSOC but do not highlight the importance of human-centred design concepts. Tafazzoli et al. Tafazzoli (2016) go over SOC positions and duties, but they don't provide a thorough examination of how people's functions are changing. The relevance and application of the paper would be improved by a more thorough analysis of how SOC personnel's responsibilities adjust to new cyber threats. Also, Muniz et al. Muniz et al. (2015) suggest a systematic approach to SOC; however, their suggestions are not backed up by any empirical data. The suggested method would be validated and given more credibility in actual SOC situations with case studies or real-world examples. While discussing helping security analysts in SOC, Basyurt et al. Basyurt et al. (2022) undervalue the significance of employee empowerment via skill development and training. The paper's suggestions would be strengthened, and SOC operations would become more efficient with the implementation of strategies for improving SOC personnel capabilities. Similarly to this, Vaarandi et al. Vaarandi and Mäses (2022) offer advice on creating and running SOC but fall short of providing specific details on sensitive people management techniques. Incorporating conversations about hiring, developing, and keeping SOC staff would increase the paper's usefulness and importance for SOC managers. Agyepong et al. Agyepong et al. (2023)'s comprehensive analysis of SOC staff training and

development may leave out important details, such as how organisational culture affects training efficacy. A more thorough examination would enhance the paper's conclusions and offer useful suggestions for SOC staff development. Lastly, Skendzic et al. Skendžić et al. (2022) cover best practices and difficulties in operating SOC, but they don't go far enough in addressing difficulties with staff. A deeper awareness of the difficulties associated with hiring, retaining, and developing skills in employees would be necessary for a more comprehensive grasp of SOC management.

2.5.2 Technologies

Key technology components include SIEM platforms, Intrusion Detection and Prevention Systems (IDPS), Endpoint Detection and Response (EDR) solutions, threat-intelligence platforms, vulnerability management tools, and forensic analysis tools. CyBOK (Section 8.4) describes how SIEM platforms have evolved from basic log aggregators into complex analytics engines supporting both detection and planning. More recently, SOAR platforms have emerged to automate parts of the Execute phase.

Technology plays a crucial role in SOC to strengthen corporate defences against a wide range of cyber threats. The adequate deployment and utilisation of state-of-the-art technology within SOC environments has become crucial as organisations face more complex and persistent malicious actors. This section explores the wide range of tools, systems, and platforms that SOC staff use to proactively detect, analyse, and respond to security problems. SOC's technological arsenal, including automation and orchestration tools, advanced threat detection systems and security information and event management (SIEM) solutions, is essential for protecting digital assets, maintaining operational continuity, and preventing the effects of cyberattacks. The constant search for innovation and adaptation to the dynamic threat landscape is at the core of SOC technology. The security of critical infrastructure needs to be updated regularly to meet the growing complexity of cyber attacks. This calls for a comprehensive strategy for technological integration that includes responsive response mechanisms, strong automation capabilities, and smooth interoperability of various security solutions. In addition, an extensive knowledge of threat intelligence, incident response techniques, and organisational risk profiles is necessary for the deployment of SOC technology. Organisations may strengthen their cyber resilience, improve threat detection capabilities, and promote a proactive security posture in the face of changing cyber threats by utilising technology inside SOC environments. Key technologies Mutemwa, M. et al. (2018); Onwubiko (2021); Vielberth et al. (2020) include threat intelligence platforms, automation tools, intrusion detection systems (IDS), endpoint detection and response (EDR) solutions,

Asset Category	Asset List
Network assets	routers, switches, servers, proxies, hosts
Operational technology	actuators, sensors, PLCs
Physical security assets	access controls, Security cameras,
Threat intelligence	open source intelligence (OSINT), DNS lookup and Geolocation, intelligence from threat-sharing platforms or other organisations
Security software	SIEM Platforms, intrusion detection/prevention systems (IDS/IPS), anti-virus (AV) systems, identity and access management, firewalls, vulnerability scanners
Virtualisation environment	Virtual machines (VMs), hypervisors, cloud environments
Other software	operating systems (OSs), Open-Source Big Data Analytics, mail servers, databases, identity and access management

Table 2.2 Technology Asset List by Category

and security information and event management (SIEM) systems. These technologies provide workflows for incident response, detection of abnormalities and potential risks, and aggregation and correlation of security events. The main characteristics of an SOC are log management, event visualisation, and incident reporting de Céspedes III and Dimitoglou (2021). These three features are intricately related since the collected logs provide input for visualisation, later used to report incidents. There are various techniques to collect data that can be organised into four categories: partial/full collection, real-time/historical, push/pull, and distributed/centralised Vielberth et al. (2020). The data can be collected/pulled from the source, or the source can push it to the collector Muniz et al. (2015). The data can also be collected in a centralised log collector Wang and Zhu (2017) or in a spread topology over different sub-nodes Gad et al. (2015). The collected data/logs are fed into a security information and event management (SIEM) tool Hossain et al. (2021), a software solution that is a combination of 2 systems: the security event management for the long-term event storage, analysis, security-log reporting and the security information system focusing on real-time monitoring. A non-exhaustive list of technologies Vielberth et al. (2020) related to the SOC can be categorised as follows in Table 2.2:

The architecture of SOC technologies 2.2 includes a complex network of tools and systems that are intended to identify, evaluate, and counteract cyber threats on multiple fronts, from virtualisation environments to operational technology (OT). To develop a strong defence mechanism against evolving cyber threats, this all-encompassing approach incorporates threat intelligence tools, security software, virtualised environments, physical security assets, and operational technologies Abd Majid and Zainol Ariffin (2021). Operational technology (OT) components, such as sensors, actuators, and programmable logic controllers (PLCs), which track and supervise processes in industries, are the fundamental building blocks of SOC technologies' architecture. Industrial control systems (ICS) and supervisory control and data acquisition (SCADA) systems rely heavily on these OT assets to identify anomalies and possible cyber intrusions, protecting vital infrastructure from cyberattacks and interruptions to operations Aslam et al. (2024). Physical security assets like security cameras and access controls are just as important as OT assets when it comes to the architecture of SOC technology. Security cameras offer continuous monitoring and legal proof in the case of security incidents, while access controls supervise physical access to areas of concern and assets. Adding to cyber threat detection and response activities, the connection with SOC monitoring systems improves situational awareness and allows quick responses to physical security breaches Marquardson (2022). For SOC analysts, threat intelligence tools are an essential source of contextual data, including knowledge about new threats, enemy strategies, and indicators of compromise (IOCs). SOC operations are improved by threat intelligence platforms, DNS lookup geolocation services, and open-source intelligence (OSINT) sources because they offer real-time threat data, empower proactive threat hunting, and strengthen incident response capabilities Mughal (2022).

The foundation of SOC technologies architecture is security software, which includes a variety of tools and systems for preventing cyberattacks. With the use of SIEM systems, SOC analysts may efficiently identify and address security issues by gathering, correlating, and analysing security event data from various sources. SOC defences include firewalls, vulnerability scanners, antivirus software, intrusion prevention systems (IPS), intrusion detection systems (IDS), and endpoint detection and response (EDR) systems. These systems secure networks, endpoints, and applications against a range of cyber threats González-Granadillo et al. (2021).

Finally, a key component of contemporary SOC architectures is the virtualisation environment, which offers resource optimisation, scalability, and flexibility for SOC operations. SOC teams can deploy and manage security tools and systems dynamically and swiftly, responding to evolving threat landscapes and operational demands, thanks to virtual machines (VMs), hypervisors, and cloud environments de Céspedes III and Dimitoglou (2021).

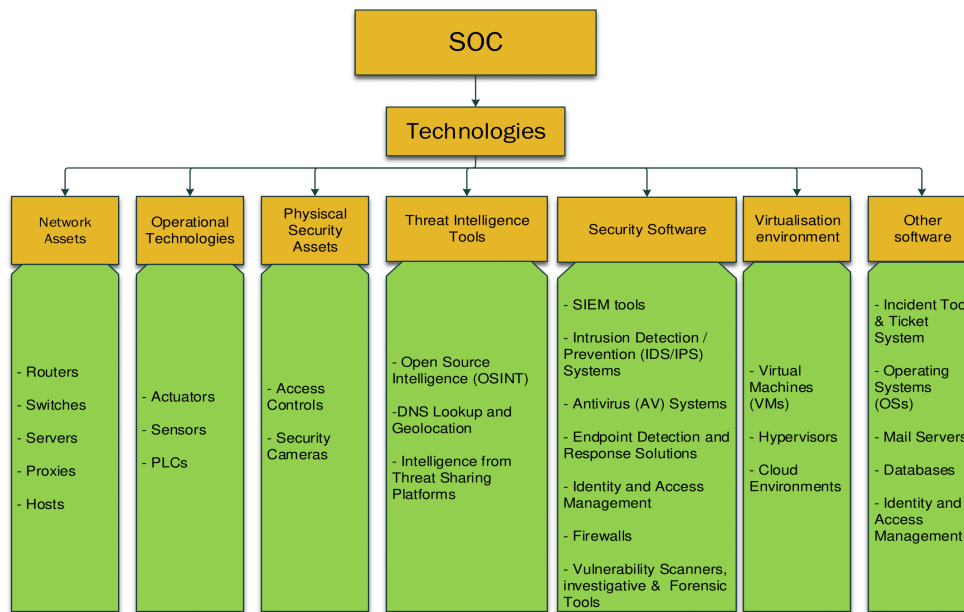


Fig. 2.2 SOC Technology Architecture

Within the identified literature, we can retain that getting the right balance between capturing too much and too little is fundamental when designing SOC technological abilities. The take is to gather data from as many sources as possible and rely on a well-known system for data normalisation, correlation and analysis. The data type may differ depending on the source of collection and categorised as data logs or intelligence, with the workflow summarised in Fig 2.3

In the pursuit of improving SOC technology design, researchers have produced several instructive papers that offer useful concepts and insights. Nevertheless, a critical analysis highlights gaps in the current research and suggests areas for development. In a remarkable article, Vielberth et al. Vielberth et al. (2020) attempt to protect critical infrastructure from cyber threats by proposing a security architecture for Industrial Control Systems (ICS). Although the study offers a thorough structure, it fails to offer helpful implementation guidelines, which could impede its acceptance in actual industrial settings. In the same spirit, Onwubiko et al. Onwubiko (2021) support a comprehensive approach to SOC architecture, but they might put theoretical ideas ahead of real-world implementation issues. Practical insights and real-world case examples would give SOC practitioners a more rounded viewpoint and increase the paper's usefulness in operational environments. Mutemwa et al. Mutemwa, M. et al. (2018) also address the integration of cyber and physical security in SOC systems,

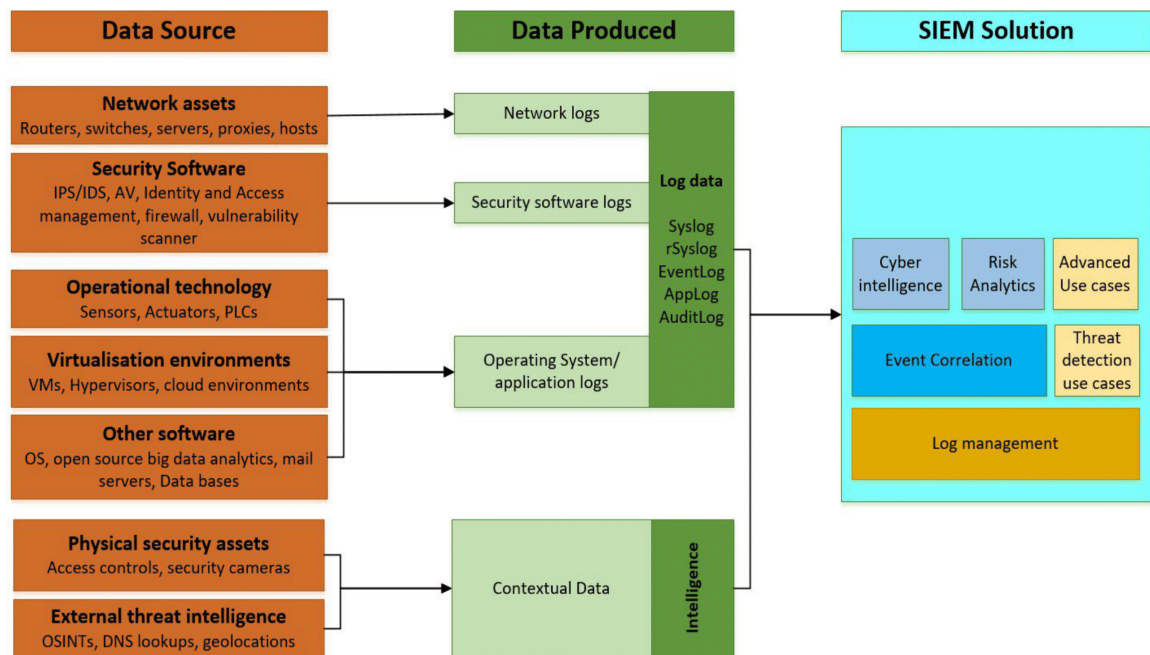


Fig. 2.3 SOC Data-Collection Workflow

emphasising the significance of bringing these domains closer together. However, the paper's practical applicability may be limited as a result of its potential lack of ability to sufficiently address the intricacies and difficulties related to integration. Comparably, De et al. de Céspedes III and Dimitoglou (2021) concentrate on creating an SOC architecture specifically designed for Small and Medium Enterprises (SMEs); however, they may overlook scalability factors that are essential for meeting a range of organisational requirements. It would be more applicable and effective for a wider range of organisations if scaling issues and resource limitations were addressed. These criticisms highlight how critical it is for SOC technology architecture research to close the gap between theoretical ideas and real-world application scenarios.

2.5.3 Processes

Processes formalise how people and technology interact. Key processes include alert triage, incident classification, escalation procedures, and post-incident review. The absence or inadequacy of formal processes is a recurring theme in the SOC challenges literature.

Processes are the fundamental structure that controls the smooth coordination of people and technologies in the context of responding to cyber threats in SOC architecture. The

development and execution of strong SOC procedures are crucial for enterprises seeking to strengthen their cyber defences and mitigate the effects of security incidents. Examining the methodical approaches, procedures, and workflows that direct incident response and operating activities in SOC environments, this section dives into the complex subject of SOC processes. SOC processes are essential for improving resource allocation, increasing the effectiveness of incident response, and promoting a continuous improvement culture. They range from event prioritisation and escalation procedures to incident response playbooks and post-incident analysis frameworks.

SOC procedures need to be flexible and responsive to allow SOC teams to quickly identify, evaluate, and mitigate security issues as a result of the increasing complexity and sophistication of cyber threats. In addition, a culture of cooperation and communication between SOC staff, cross-functional teams, and outside stakeholders should support SOC procedures. In an increasingly hostile digital ecosystem, organisations can strengthen their resilience against cyber threats and cultivate a proactive security posture by implementing well-defined and standardised procedures for incident management, exchanging threat intelligence, and knowledge transfer.

Fig 2.4 shows a list of SOC processes involving the cyber incident playbook, Incident response, Joiners, movers, leavers, vulnerability and change management. Cyber Incident Playbook Onwubiko and Ouazzane (2022a) is a process that focuses on the importance of developed procedures and instructions to ensure organised and coordinated response actions. It addresses teamwork and communication, incident triage, documentation and reporting, handling workflow, and tools and technologies. SOC teams need a Cyber Incident Playbook as an important guide to efficiently handle and react to cyber threats and security incidents. This detailed document defines predetermined protocols, tasks, and processes that must be done in the event of particular security incident types. SOC analysts can more quickly and effectively reduce the effects of cyberattacks on organisational assets and operations by using a Cyber Incident Playbook, which is primarily designed to simplify the incident response process. The process of creating a cyber incident playbook starts with an in-depth examination of the threat landscape facing the company, which includes locating potential points of entry, weak points, and common strategies employed by attackers. Playbooks designed to handle ransomware outbreaks, phishing assaults, malware infections, data breaches, and insider threats are developed based on this evaluation. Every playbook offers a methodical framework for identifying, analysing, containing, eliminating, recovering from, and reviewing incidents once they have occurred.

The Incident Response Process Ahmad et al. (2021); Vielberth et al. (2020) focuses on steps such as incident identification, containment, analysis, mitigation, and post-incident

activities. It is a crucial part of cybersecurity strategy, which tries to reduce the damage that security breaches do to the data, assets, and operations of organisations. Usually, there are multiple important phases in the incident response process, each of which is intended to handle a certain component of incident management. Organisations utilise vulnerability management as a preventative measure to find, prioritise, address, and patch security flaws in their systems, apps, and IT infrastructure. It is an essential part of a cybersecurity strategy that lowers the attack surface and minimises the chance of data and security breaches. Usually, the vulnerability management process consists of multiple crucial stages, each of which is intended to handle a particular facet of risk reduction, repair, and vulnerability assessment Foreman (2019); Walkowski et al. (2021). Entities prioritise vulnerabilities according to their severity and possible impact on business operations, starting with vulnerability identification through automated scanning tools and manual testing. Using risk assessments, entities appraise the probability of exploitation and its consequences on data availability, confidentiality, and integrity. This allows them to formulate tailored mitigation tactics. To resolve vulnerabilities found and lower the risk of exploitation, remediation procedures include the prompt application of security updates and configuration modifications. The efficacy of remediation efforts is ensured by ongoing monitoring and review, which enables organisations to modify their vulnerability management plans in reaction to new threats and shifts in the threat environment.

Employee lifecycle management, commonly referred to as the Joiner, Movers, and Leavers (JML) process, is the process of managing user accounts and permissions during the whole employment lifecycle. This includes onboarding new hires (joiners), updating access privileges for current hires (movers), and removing access for departing employees (leavers). Ensuring regulatory compliance, protecting sensitive data from unauthorised access, and preserving data security all rely on the JML process being implemented effectively Onwubiko and Ouazzane (2019). The JML process involves provisioning new user accounts, allocating suitable access rights based on job roles and responsibilities, and giving required training on security rules and procedures during the onboarding phase. This guarantees that new hires have the access needed to carry out their duties while preserving the principle of least privilege. The JML process makes it easier for employees to promptly modify their access privileges to match their new jobs and responsibilities as they move within the company. Lastly, to guard against possible security lapses or data leaks, the JML procedure suggests that workers' access privileges are promptly revoked when they leave the company.

The list of academic papers mentioned in this section presents a variety of understandings of the SOC process architecture. However, a critical examination reveals areas where these papers could be enhanced to provide more valuable insights for SOC practitioners. Ahmad

et al. Ahmad et al. (2021) investigate how SOC processes could enhance cybersecurity; however, to improve the practical relevance of their conclusions, their research could include case studies to support their assertions. Furthermore, Vielberth et al. Vielberth et al. (2020) talk about industrial control systems (ICS) security architecture but miss details of SOC procedures unique to ICS environments. Their study would be enhanced by extending their reach to include insights into SOC procedures designed for critical infrastructure sectors. Foreman et al. Foreman (2019) also look at vulnerability management in SOC setups; however, they don't provide any useful advice on how to put those approaches into practice.

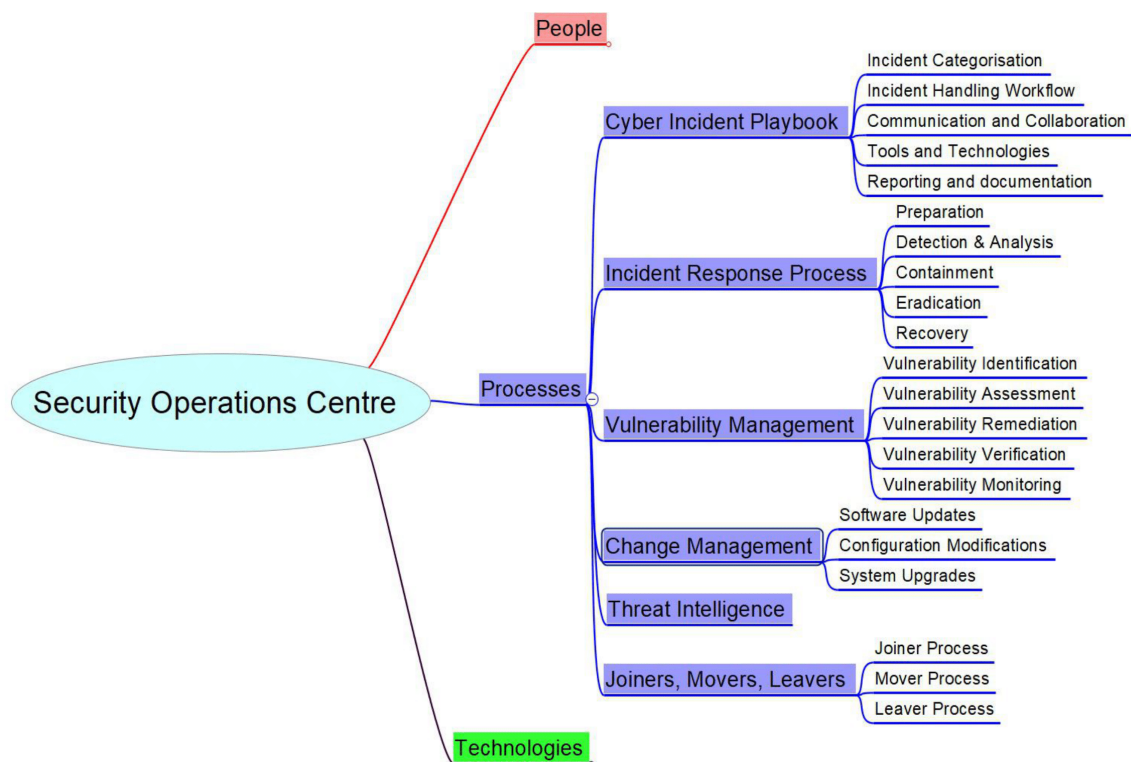


Fig. 2.4 Architecture of SOC in terms of Processes

Table 2.3 lists the academic publications in the SOC Architecture area, arranged by the three sub-sections that make up the section: People (13/36), Technologies (16/35), and Processes (6/35). After carefully examining the information shown in Table 2.3, some significant findings emerge concerning the allocation of research projects among the SOC architectural subcategories. Remarkably, the 16 papers devoted to technologies suggest that most scholarly contributions seem to focus on the technological aspects of SOC architecture. This tendency points to a significant academic focus and interest in clarifying and improving the technology frameworks and solutions that are operationalised in SOC contexts. On the other hand, the number of pieces related to the human-centred elements of SOC architecture

(13 sources) devoted to people and the procedural aspects (6 sources) devoted to processes seems to be rather smaller in contrast. Moreover, the great focus on technology implies that individuals are acknowledging the importance of cutting-edge technology to enhance the efficiency of SOC operations. However, we need to keep in mind that people and processes are still crucial to SOC. Another reason why processes are crucial is that they offer a foundation for planning and completing SOC duties.

2.6 Malware Detection and Analysis Techniques

The analysis process Plohmann, D. et al. (2013) of malware detection is composed of four phases (initialisation, preliminary, in-depth analysis, and mitigation) supported by an analysis management system. Once the goals and the initial tasks have been defined, the preliminary analysis stage will start and continue until the malware has been classified. If required, an in-depth analysis will be conducted with reverse engineering at the third stage. The mitigation concept draws the active countermeasure in the last stage that would have fallen into the containment, eradication, and recovery stages regarding the big picture of the incident management process.

The detection and analysis stage involves identifying the incident characteristics (malware category, ports, protocols, exploited vulnerabilities, malicious filenames, etc.), identifying the infected hosts, engaging incident response, and investigating malware Souppaya et al. (2013). This is described as the identification phase Naseer (2018), where the incident response team determines whether an alert is an incident.

2.6.1 Malware Detection

Detection approaches are categorised as signature-based (misuse detection) and behaviour-based (anomaly detection), as defined in CyBOK Section 8.3. Signature-based detection relies on pattern matching against a known-malware database; it is highly accurate for known threats but ineffective against novel or obfuscated malware. Behaviour-based detection monitors system or network activity for deviations from a learned baseline; it can detect unknown malware but generates significant false positives.

Malware detection can be performed either automatically or manually. Malware detection methods are composed of three types of methods Guo et al. (2020): signature-based, static, and dynamic detections. Static detection disassembles the malware and analyses the opcodes, static API sequences and execution logic without running it. Dynamic detection acquires

Literature	People	Technologies	Processes
Dietz et al. (2020)	✓		
Onwubiko and Ouazzane (2022a)			✓
de Céspedes III and Dimitoglou (2021)		✓	
González-Granadillo et al. (2021)		✓	
Torres (2015)	✓		
Kowtha, S. et al (2012)	✓		
Abd Majid and Zainol Ariffin (2021)		✓	
Tafazzoli (2016)	✓		
Walkowski et al. (2021)			✓
Aslam et al. (2024)		✓	
Basyurt et al. (2022)	✓		
Muniz et al. (2015)	✓	✓	
Vaarandi and Mäses (2022)	✓		
Hossain et al. (2021)		✓	
Skendžić et al. (2022)	✓		
Basyurt et al. (2022)	✓		
Vielberth et al. (2020)			✓
Vielberth et al. (2020)		✓	
Onwubiko (2021)		✓	
Shahjee and Ware (2022)	✓		
de Céspedes III and Dimitoglou (2021)		✓	
Onwubiko (2015)	✓		
Vielberth et al. (2020)		✓	
Wang and Zhu (2017)		✓	
Onwubiko and Ouazzane (2019)			✓
Gad et al. (2015)		✓	
Marquardson (2022)		✓	
Mughal (2022)		✓	
Onwubiko and Ouazzane (2022a)	✓		
Ahmad et al. (2021)			✓
Mutemwa, M. et al. (2018)		✓	
Foreman (2019)			✓
Agyepong et al. (2023)	✓		
de Céspedes III and Dimitoglou (2021)		✓	
Total	13/35 (37.14%)	16/35 (45.71%)	6/35 (17.14%)

Table 2.3 List of Literature Regrouped by SOC Architecture

behavioural features (network activity, system calls, file operations) by executing the file sample. Signature-based detection works by extracting common characteristics (byte sequence, file size, file hash, imported/exported functions, offsets, strings) for each file and matching them with known signatures that have been collected before. Fig. 2.5 shows a sample of code of the malware *Ocean-Lotus* and its corresponding YARA signature, showing that any file with a size less than 200KB with the type 'Macho' containing the strings *a1=80 7C 39 3C 32 BA BB 80 F3 B9 B4 34 39 80*, *a2=FC BF 34 BA 7C BA 34 36 B9 BC BA 3C 80 7C 39 3C*, *a3=32 BA BB 76 BA 34 3C BF B9 B7 8F 30 B3 B9 3C 39*, and *b1=9C 85 89 27 8B 9C 85 89 27 8B 9C 85 89 27 8B 9C* should be identified as the threat actor *Ocean-Lotus*.

There are three malware detection techniques Sharma and Bharti (2021): signature, heuristic and specification-based. Heuristic-based detection examines code for suspicious properties. This detection approach is characterised by two phases: the training phase, which studies system behaviour in the absence of an attack, and the detection phase, which consists of comparing the profile against the current performance of the system, where any changes are labelled as potential attacks. Specification-based techniques are similar to heuristic detection. However, instead of relying on machine learning techniques, specification-based methods are centred on manually developed descriptions that capture legitimate system behaviours.

```

00011ef0: 8e3e c6d0 d1c4 d1c7 8e3d c6c1 c221 c4c8  ,>.....=...!..
00011f00: dac6 d6c2 8e91 9191 918e 4a8c 8b1b aa19  ....J....
00011f10: 994a 2baa 1b1b c8ce d5ce dcc5 0000 0000  .J+.....
00011f20: 807c 393c 32ba bb80 f3b9 b434 b834 3980  .|9<2.....4.49.
00011f30: fcbf 34ba 7cba 3436 b9bc ba3c 807c 393c  ..4|.46...<|9<
00011f40: 32ba bb76 ba34 3cb9 bfb7 8f30 b3b9 3c32  2..v.4<...0..<2
00011f50: 2012 9751 1556 11a3 5495 55aa b39d a587  ..Q.V..T.U....
00011f60: 91a7 ba85 b393 8d9d bd00 0000 0000 0000  ....
00011f70: 9c85 8927 8b9c 8589 278b 9c85 8927 8b9c  ...'.....'...
00011f80: 8589 278b 9c85 8927 8b9c 8589 278b fd3c  ..'.....'...<

strings:
$a1 = { 80 7C 39 3C 32 BA BB 80 F3 B9 B4 34 39 80 }
$a2 = { FC BF 34 BA 7C BA 34 36 B9 BC BA 3C 80 7C 39 3C }
$a3 = { 32 BA BB 76 BA 34 3C B9 BF B7 8F 30 B3 B9 3C 32 }
$b1 = { 9C 85 89 27 8B 9C 85 89 27 8B 9C 85 89 27 8B 9C }
condition:
Macho and filesize < 200KB and all of them

```

Fig. 2.5 Signature-based Detection for Ocean-Lotus malware

2.6.2 Malware analysis

Static analysis examines a malware binary without executing it, extracting metadata such as file hash, embedded strings, imported libraries, and disassembled code (Moser et al. (2007b)). Dynamic analysis executes the malware in a controlled environment (sandbox) and observes its behaviour, including file system modifications, registry changes, network communications, and API call sequences. Hybrid analysis combines both approaches. The literature reviewed indicates that most SOCs rely on automated sandboxing for initial triage, with manual static

Category	Tool	Literature
Virtualisation	VMWare Workstation/Fusion, VirtualBox, Hyper-V, SHADE Sandbox, Anlyz Sandbox, Cuckoo Sandbox, Joe Sandbox, Mbox	Murthy et al. (2019), Mohanta and Saldanha (2020)
Dynamic Analysis	Process Hacker, Process Monitor, RegShot, Wireshark, fakeDNS, ProcDOT, Wireshark, Fiddler, TCPLogView, Signsrch, pscan	Mohanta and Saldanha (2020)
Static Analysis (Property, code)	(property: BinText, PeStudio, Strings, CFF Explorer, peframe, Detect it Easy, Exeinfo pe, PEiD, ssdeep, pafish, Yara, Strings) (code: Ghidra, x64dbg/x32dbg, OllyDumpEx, jmp2it, Scylla, IDA, OllyDbg, OllyDump)	Mohanta and Saldanha (2020), Pachhala et al. (2021)
Memory Analysis	WinPMEM, BelkaSoft Live RAM Capturer, Volatility Framework, Rekall, Redline	Mohanta and Saldanha (2020)

Table 2.4 Malware Analysis Tools

analysis reserved for high-severity or novel samples.

Malware analysis is the investigation of malware behaviour to identify its mechanisms Makandar and Patrot (2015). Malware is a programme developed to disturb, harm, or gain access to data and systems in an unauthorised manner Naik et al. (2020). There are many types of malware Ehsan et al. (2022), Dener et al. (2022), Wazid et al. (2019), Murthy et al. (2019) categorised as Trojan, virus, worm, ransomware, rootkit, key loggers, spam, adware, spyware, fileless malware, and backdoor. Generally, there are 2 ways to go about malware analysis Chaudhary (2020): Static and dynamic, on top of which the hybrid model or the memory malware analysis is added. Analysing malware requires a set of tools Salman et al. (2019), Mohanta and Saldanha (2020) Sharma and Bharti (2021), Pachhala et al. (2021), Chaudhary (2020) summarised in Tab 2.4 depending on the type of analysis performed, the operating system or the virtual environment.

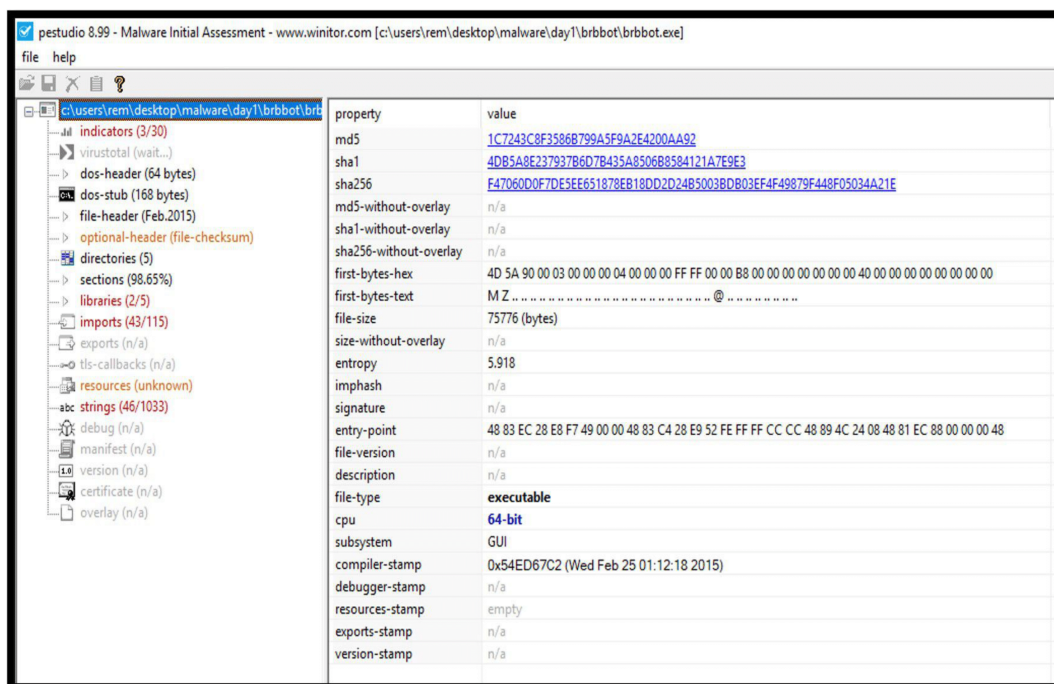
(A) Static Analysis

It is an analysis concentrating on a signature of extracted portable executable files (PE) file types such as exe, DLL, documents, assembly code, byte code, etc. Pachhala et al.

(2021). The static analysis starts with basic analysis and moves to advanced static analysis Aslan and Samet (2017).

The basic static analysis provides a signature involving the complete structure of the programme. The basic static analysis is to define whether a file is malicious or not. This is the triage phase to determine if the sample is malware, how bad it is, how to detect it, and how to analyse it. Fig 2.6 is an example of static properties of the malware brbbot from the tool Pestudio 8.99. The Sections of PE header include *indicators*, *the DOS header*, *the DOS hub*, *the file header*, *the optional header*, *directories*, *sections*, *libraries*, *imports*, *exports*, *tls-callbacks*, *resources*, *strings*, *debugs*, *manifest*, *version*, *certificate*, and *overlay*. Every field of the PE header has properties. The main static properties of malware brbbot are *md5*, *sha1*, *sha256*, *md5-without-overlay*, *sha1-without-overlay*, *sha256-without-overlay*, *first-bytes-hex*, *file-size*, *size-without-overlay*, *entropy*, *impash*, *signature*, *entry-point*, *file-version*, *description*, *file-type*, *CPU*, *subsystem*, *compiler-stamp*, *resources-stamp*, *export-stamp*, and *version-stamp*.

The Advanced static analysis works by investigating the static structure and the



property	value
md5	1C7243C8F3586B799A5F9A2E4200AA92
sha1	4DB5A8E237937B6D7B435A8506B8584121A7E9E3
sha256	F47060D0F7DE5EE651878EB18DD2D24B5003BD03EF4F49879F448F05034A21E
md5-without-overlay	n/a
sha1-without-overlay	n/a
sha256-without-overlay	n/a
first-bytes-hex	4D 5A 90 00 03 00 00 04 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
first-bytes-text	M Z @
file-size	75776 (bytes)
size-without-overlay	n/a
entropy	5.918
impash	n/a
signature	n/a
entry-point	48 83 EC 28 E8 F7 49 00 00 48 83 C4 28 E9 52 FE FF CC CC 48 89 4C 24 08 48 81 EC 88 00 00 00 48
file-version	n/a
description	n/a
file-type	executable
cpu	64-bit
subsystem	GUI
compiler-stamp	0x54ED67C2 (Wed Feb 25 01:12:18 2015)
debugger-stamp	n/a
resources-stamp	empty
exports-stamp	n/a
version-stamp	n/a

Fig. 2.6 Static Property Analysis Features of malware brbbot

features of a programme without actually executing it Prähofer et al. (2012). The advanced static analysis provides instructions which define the intended purpose of a programme by using a debugger and a disassembler. Static code analysis emphasises analysing the source code of the software. The main goal is to check the programme

structure and elements or to estimate the programme states and behaviour.

Executable files (*.BAT*, *.COM*, *.EXE*, *.BIN*, etc.) are used to fulfil various functions or operations on a computer. They can be represented as a series of hexadecimal values for corresponding bytes of a binary file. Disassembling is the process of translating the content of the binary file into assembly language Soni et al. (2022). Malware usually uses binary packers, such as *UPX* and *ASP Pack Shell*, to avoid being analysed Sihwail et al. (2018). It has to be unpacked and decompressed before being analysed. A disassembler such as *IDA Pro* and *OllyDbg* can be used to decompile Windows executable files and display assembly instructions to identify indicators of compromise (IoC). Analysts identify static patterns Gandotra et al. (2014), Sihwail et al. (2018), Wei et al. (2019) to detect the intent of a malicious code. API calls are commands in the code that instruct systems to execute certain operations. APIs with malicious behaviour Murthy et al. (2019) are listed in Tab 2.5. Analysts can determine whether a file is malicious by its API calls, some of which are characteristic of certain types of malware. For instance, *NtReadFile*, *NtWriteFile*, *LdrGetProcedureAddress*, *RegQueryValueExW*, *NtClose* are API calls invoked by the ransomware *JigsawLocker* where *NtReadFile* routinely reads data from an open file, *NtWriteFile* writes data, *LdrGetProcedureAddress* retrieves the address of an exported function/procedure or variable, *RegQueryValueExW* retrieves the type and data for the specified value name associated with an open registry key, and *NtClose* closes handles to objects. The APIs in malware PE files are kept in IATs (Import Address Table) and can be obtained through reverse engineering tools such as IDA. Every API contains a sequence of assembly instructions, and each assembly instruction contains a mnemonic and a sequence of operands. The assembly instructions Hao et al. (2022) include *push*, *sub*, *xor*, *mov*, *test*, *jnz*, *call* as illustrated in Fig 2.7.

(B) Dynamic Analysis

The Dynamic analysis, also known as behavioural analysis, focuses on studying a programme's behaviour as it executes Murali et al. (2020). This is to take a closer look at malware by observing how it behaves when it runs within a simulated or controlled virtual environment. Dynamic analysis environments also use emulators and hypervisors Singh and Singh (2018). The idea is to compare the snapshots of the complete system state before and after a suspicious sample is run. Reports of a variety of activities Guo et al. (2020), Jindal et al. (2019), Chanajitt et al. (2021) are examined, such as API Calls, Mutexes, File System Changes, Registry Changes, and Loaded DLLs.

- **API Calls:** In behaviour-based malware detection, Windows Application Program-

Malicious Behaviour	API Call Sequence
Modify File Attribute	GetFileAttribute, SetFileAttribute
Modify Time of File	GetFileTime, SetFileTime
Load Register	RegCreateKey, RegSetValue, RegCloseKey
Enumerate all process	CreateToolhelp32Snapshot, Process32First, Process32Next, WTSEnumerateProcesses
Privilege Escalation	OpenProcessToken, LookupPrivilegeValueA, AdjustTokenPrivileges
Terminate Process	TerminateProcess
Screen Capture	(GetDC, GetWindowDC), CreateCompatibleDC, CreateCompatibleBitmap, SelectObject, BitBlt, WriteFile
Hooking	SetWindowsHookA, CallNextHookEx
Downloader	URLDownloadToFile, (WinExec, ShellExecute)
Enumerate all process	CreateToolhelp32Snapshot, Process32First, Process32Next
Anti debugging	(IsDebuggerPresent, CheckRemoteDebuggerPresent, OutputDebugStringA, OutputDebugStringW)
Synchronisation	CreateMutexA, CreateSemaphoreW
Key Logger	(FindWindowA, ShowWindow, GetAsyncKeyState) (SetWindowsHookEx, RegisterHotKey, GetMessage, UnhookWindowsHookEx)
Dropper	FindResource, LoadResource, SizeOfResource

Table 2.5 List of API Call Sequences per Malicious Behaviour

```

undefined8      Stack[-0x38...local_38]      XREF[2]: 14000124a(W),
                                                    140001292(W)
FUN_140001150    XREF[1]: FUN_1400027c0:140002808
140001150    PUSH    RBX
140001152    PUSH    RDI
140001153    SUB     RSP, 0x48
140001157    XOR     EBX, EBX
140001159    XOR     ECX, ECX
14000115b    MOV     dword ptr [RSP + local_res8], EBX
14000115f    CALL    qword ptr [->KERNEL32.DLL::GetModuleHandleA]
140001165    MOV     RDI, RAX
140001168    TEST    RAX, RAX
14000116b    JNZ     LAB_14000118f
14000116d    CALL    qword ptr [->KERNEL32.DLL::GetLastError]
140001173    MOV     EBX, EAX
140001175    TEST    EAX, EAX

```

Fig. 2.7 Example of Assembly Instructions - Malware brbbot

ming Interface (API) call sequences are regarded as one of the most representative traits. In the Windows operating system, user applications cannot gain access to hardware or system resources directly. However, they can depend on interfaces provided by dynamic-link libraries (DLLs). These libraries include *advapi32.dll*, *kernel32.dll*, *gdi32.dll*, *user32.dll*, and offer functionalities to access system resources and hardware Amer and Zelinka (2020), Choudhary and Vidyarthi (2015).

- **Mutexes:** Mutual exclusion objects (Mutexes) control the simultaneous access of the system resources. Malware creators use them to avoid infecting a system more than once. When malware infects a system, it creates an infection marker. It might be the name of its process, a particular file it creates on the disk, a registry key it looks for to see whether it is already on the system or a common marker that can be a mutant/mutex. A mutant with a strange name can be an indicator of malware. For instance, a meaningful mutant could be *DBMinMutex*. A strange one would be *LXCVOIMGIXSORTAI* Zeltser (2019).

- **File System Changes:** A malware sample's interaction with the host file system may serve as a good indicator of malicious behaviour. We consider every crucial file operation, including creation, reading, writing, modifying, and deleting Or-Meir et al. (2019).

- **Registry Changes:** The registry stores information about a piece of hardware, installed programmes, values, and options that other processes can use. A registry key is a hierarchical database of settings and information. There is a higher chance that an executed file is malicious if certain registry changes have been made. Features that

Name of DLL	Functions (API Call)
ADVAPI32.dll	CryptReleaseContext, CryptAcquireContextA, CryptGenRandom, CryptEncrypt, CryptGetKeyParam, CryptAcquireContextW, CryptDestroyKey
CRYPT32.dll	CryptQueryObject, CertFreeCertificateContext, CertFindCertificateInStore, CryptMsgGetParam, CryptDecodeObjectEx, CryptImportPublicKey-Info, CryptBinaryToStringA
CRYPTNET.dll	CryptGetObjectUrl
CRYPTUI.dll	CryptUIDlgSelectCertificateFromStore

Table 2.6 Example of DLL used in Ransomware with Related Function Calls

were deleted from the registry and those that were added came from both malicious and benign files Ijaz et al. (2019).

• **Loaded DLLs:** Analysis reports include shared library code loaded by a programme. Almost every executable programme imports DLLs during execution. These DLLs can provide insights into the types of APIs invoked by the programme Subedi et al. (2018). Tab 2.6 is an example of a list of DLLs used by the crypto-ransomware with function calls used by the corresponding DLLs.

(C) Hybrid Analysis

To lessen the drawbacks of both analysis approaches and achieve a higher rate of detection, some earlier studies combined data extracted through static and dynamic analysis together Aboaoja et al. (2022). This technique is a combination of both static analysis and dynamic analysis to address the weaknesses of each methodology Alsmadi and Alqudah (2021). Certain actions that can be concealed at run-time might be spotted when unpacking the binary files or viewing them in assembly code. Equally, obfuscated opcodes can be discovered when they are executed, and actions detected live Hadiprakoso et al. (2020). The hybrid analysis aims to identify the key sources of variation in a data set. This type of analysis is useful for multiple sources of data that overlap partially or completely, such that it would be difficult to interpret one study without the other. The goal of a hybrid-based analysis is to reveal which variables are correlated and, therefore, may be related to each other; then, those variables can be used for subsequent analyses together.

(D) Memory Analysis

Malware can use some obfuscation techniques to deceive static analysis by modifying PE code and some evasion techniques to mislead dynamic analysis. Therefore, memory dumps are needed to carry out a comprehensive and accurate analysis Arfeen et al. (2022). A memory dump is an exact copy of what's stored in RAM and contains all types of information (data from both physical and virtual memory locations) stored by programmes that are currently running on a computer Shree et al. (2022). Memory analysis can be broken into two phases: the memory dump's parsing and feature proposition. In the first phase, the memory dumps are parsed using the volatility tool, and relevant data is obtained through plugins. In the latter phase, the extracted data is changed into valuable features Okolica and Peterson (2010). Many tools Carrier et al. (2022) are used to conduct forensic memory analysis, including examples listed in Tab 2.7. Memory Forensics offers a valuable perspective on malware. The memory includes active processes, details about network connections, recently processed data, the registry, and more. Examining the memory image of an infected system offers valuable details regarding the specimen's capability. Memory images of infected production systems provide context and are useful even when the actual specimen is unavailable. An example of a memory analysis workflow Kara (2022) starts with gathering memory data, followed by the memory analysis, and is finalised by an analysis report.

(E) Detection and Response Automation

The automation of detection and response improves advanced technologies such as artificial intelligence, machine learning, and orchestration to optimise threat detection and mitigation's speed and accuracy. The NotPetya malware attack is a notable case that highlights the significance of automated detection and response Jha (2023). The prompt containment of incidents and identification of the malware's rapid lateral movement are made possible by automated systems, demonstrating the effectiveness of automated responses in reducing the impact of large-scale attacks. In addition, the emergence of polymorphic malware (which continuously modifies its code to avoid detection) makes automated methods for real-time threat identification necessary. The harm inflicted by the Emotet ransomware Kovács (2022) is a notable illustration of how automated detection and response systems were instrumental in recognising and eliminating the constantly changing threat landscape. Also, the integration of Generative AI (GenAI) tools Gupta et al. (2023), like ChatGPT and Google Bard, into cybersecurity defence and offensive strategies underlines how automated systems can now be used to launch

Type	Tool	Description
Physical Memory	- MAGNET RAM - ManTech Memory DD - Forensic Tool Kit (FTK)	- Free imaging tool intended to capture the physical memory - Captures a record of physical or RAM, which is lost when a computer is shut down - It scans a hard drive, searching for a variety of information such as deleted emails and text strings used as password dictionaries to crack encryption
Virtual Memory	- Volatility - VolMemLyzer-V2 - WinPMEM - BelkaSoft - DumpIt	- Open-source memory forensics framework - Volatility Memory Analyser, a feature extraction module that utilises Volatility plugins to extract memory features to generate a CSV file - For capturing memory - Live RAM Capturer - For capturing memory

Table 2.7 Example List of Physical and Virtual Memory Analysis Tools

attacks or proactively detect and address sophisticated threats. Machine learning (ML) is used to automate the detection, classification, and analysis of malware threats in real time and can also help automate many of the manual processes that are typically used in malware analysis. Some examples of ML-based methods used in malware analysis within SOC teams include behavioural analysis, malware classification and prediction, automated incident response, anomaly detection, malware reverse engineering, clustering, signature generation, evasion detection, and prioritisation Aboaoja et al. (2022); Wei et al. (2019). AI/ML-based technology strengthens and enhances the capabilities of SOC teams, and it is also essential to evaluate and update the models regularly, continuously monitor their performance, and ensure they remain effective in detecting and responding to malware incidents. However, ML technology also has its own set of challenges, such as false positives and false negatives and the need for a large dataset to train the models. Additionally, AI and ML are only as good as the data they are trained on, and a lack of diverse and representative data can lead to bias in the models Markevych and Dawson (2023).

It is possible to define machine learning (ML) as a branch of artificial intelligence (AI) science that deals with the study and application of algorithms that can learn from data. According to its description, it is a technique for data analysis that enables prediction-making from data Pachhala et al. (2021). ML is increasingly being used in malware analysis within SOC environments. It can be used to detect, classify and analyse malware threats in real time and can help to automate many of the manual processes that are typically used in malware analysis. Some examples of how ML is being used in malware analysis within SOC teams include behavioural analysis, malware classification, malware prediction, automated incident response, anomaly detection, malware reverse engineering, clustering, signature generation, evasion detection, prioritisation, etc. Aboaoja et al. (2022); De Andrade et al. (2013); Sharma and Bharti (2021); Soni et al. (2022); Wei et al. (2019). It's important to note that ML is not a replacement for human expertise but rather a powerful tool that can strengthen and enhance the capabilities of SOC teams. It's also essential to evaluate and update the models, continuously monitor their performance, and ensure they remain effective in detecting and responding to malware incidents. However, ML technology also has its own set of challenges, such as false positives, false negatives, and the need for a large dataset to train the models. Additionally, AI and ML are only as good as the data they are trained on, and a lack of diverse and representative data can lead to bias in the models.

Table 2.8 Classification of Literature

Review Paper	Detection	Static Analysis	Dynamic Analysis	Hybrid Analysis	Memory Analysis	Response Automation
Sharma and Bharti (2021)	✓	×	×	×	×	×
Pachhala et al. (2021)	×	✓	×	×	×	×
Zeltser (2019)	×	×	✓	×	×	×
Carrier et al. (2022)	×	×	×	×	✓	×
Pachhala et al. (2021)	×	✓	×	×	×	×
Gandotra et al. (2014)	×	✓	×	×	×	×
Kovács (2022)	×	×	×	×	×	✓
Aslan and Samet (2017)	×	✓	×	×	×	×
Jindal et al. (2019)	×	×	✓	×	×	×
Hao et al. (2022)	×	✓	×	×	×	×
Murali et al. (2020)	×	×	✓	×	×	×
Gupta et al. (2023)	×	×	×	×	×	✓
Guo et al. (2020)	×	×	✓	×	×	×
Markevych and Dawson (2023)	×	×	×	×	×	✓
Chanajitt et al. (2021)	×	×	✓	×	×	×
Chaudhary (2020)	×	✓	×	×	×	×
Amer and Zelinka (2020)	×	×	✓	×	×	×
Alsmadi and Alqudah (2021)	×	×	×	✓	×	×
Prähofer et al. (2012)	×	✓	×	×	×	×
Wei et al. (2019)	×	×	×	×	×	✓
Or-Meir et al. (2019)	×	×	✓	×	×	×
Wei et al. (2019)	×	✓	×	×	×	×

Continued on next page

Review Paper	Detection	Static Analysis	Dynamic Analysis	Hybrid Analysis	Memory Analysis	Response Automation
Ijaz et al. (2019)	×	×	✓	×	×	×
Subedi et al. (2018)	×	×	✓	×	×	×
Aboaoja et al. (2022)	×	×	×	✓	×	×
Shree et al. (2022)	×	×	×	×	✓	×
Hadiprakoso et al. (2020)	×	×	×	✓	×	×
Arfeen et al. (2022)	×	×	×	×	✓	×
Choudhary and Vidyarthi (2015)	×	×	✓	×	×	×
Okolica and Peterson (2010)	×	×	×	×	✓	×
Sihwail et al. (2018)	×	✓	×	×	×	×
Kara (2022)	×	×	×	×	✓	×
Jha (2023)	×	×	×	×	×	✓
Singh and Singh (2018)	×	×	✓	×	×	×
Soni et al. (2022)	×	✓	×	×	×	×
Sihwail et al. (2018)	×	✓	×	×	×	×
Aboaoja et al. (2022)	×	×	×	×	×	✓
Pachhala et al. (2021)	×	×	×	×	×	✓
Guo et al. (2020)	×	✓	×	×	×	×
Aboaoja et al. (2022)	×	×	×	×	×	✓
De Andrade et al. (2013)	×	×	×	×	×	✓
Murthy et al. (2019)	×	✓	×	×	×	×
Soni et al. (2022)	×	×	×	×	×	✓
Wei et al. (2019)	×	×	×	×	×	✓
Sharma and Bharti (2021)	×	×	×	×	×	✓

Continued on next page

Review Paper	Detection	Static Analysis	Dynamic Analysis	Hybrid Analysis	Memory Analysis	Response Automation
Total	1/45 (2.22%)	13/45 (28.89%)	11/45 (24.45%)	3/45 (6.67%)	5/45 (11.11%)	12/45 (26.67%)

Overall, the identified literature for SOC Technologies is focused on various types of analysis. The categories of data collection, malware incident handling, static analysis, dynamic analysis, hybrid analysis, and memory analysis suggest that there is significant importance in understanding the different techniques and methods used to analyse data and identify potential security threats. Each of these categories may provide information on different aspects of analysis. The majority of the literature is focused on Automation techniques. The critiques of reviewed papers can be summarised around the limited coverage of advanced malware techniques due to the constant evolution of techniques to evade detection, the insufficient consideration of scalability that may include the efficient processing of a high volume of samples, and the lack of concern for the integration of threat intelligence, which could aid SOC analysts in understanding the context, attribution, and potential impacts of malware incidents. Future literature should focus on the above issues to improve the capabilities of SOC teams in identifying and mitigating malware threats successfully.

Table 2.8 presents a list of literature reviewed in the Malware Incident Handling section. The reviewed papers are organised by topic, including detection (1 paper), static analysis (13), dynamic analysis (11), hybrid analysis (3), memory analysis (5 papers), and automation (12 papers). Categorising the papers by topic area allows for a structured overview of the current research in each malware analysis subdomain. Based on the above data, we can make the following analysis: Automation techniques represent the largest group of the reviewed literature on malware event handling, with 12 papers. This implies that there is a strong need for research on automated techniques and instruments to improve and optimise malware analysis practices. Eleven papers cover topics related to the second-largest group: dynamic analysis techniques, which involve running malware samples and observing their behaviour. This emphasises how crucial it is to comprehend the traits and behaviours of harmful code at runtime since dynamic analysis can offer important insights into the capabilities and possible effects of malware. Ten publications cover static analysis approaches, which look at the structure or code of malware without running it. Through disassembly, decompilation, and other code analysis techniques, this field of study seeks to retrieve valuable information from malware samples, frequently assisting dynamic analysis techniques. In contrast, there are fewer studies in the remaining categories: hybrid analysis (3 papers), memory analysis (5 papers), and detection (1 paper). Their existence does, however, highlight how complex malware analysis is and how different approaches are required to tackle different facets of the issue.

2.7 SOC Operational Challenges: Critical Assessment

The following challenges were consistently identified across the literature reviewed. For each, the evidence base is assessed and gaps in existing research noted.

One essential responsibility of SOC is the efficient management of malware events. However, SOC teams encounter some challenges when responding to these occurrences. To increase incident response capabilities and reduce the risks related to malware incidents, it is essential to recognise and address these issues. This Section identifies SOC challenges and provides necessary recommendations and guidance.

2.7.1 Alert Overload and Triage Inefficiency

Multiple studies identify alert volume as a primary source of SOC analyst fatigue and error (Vielberth et al. (2020); Agyepong et al. (2023)). Alert overload is a direct consequence of the base-rate fallacy (Rashid et al. (2021), Section 8.3.6): detection systems optimised for sensitivity generate many false positives in large-scale environments. A critical gap in the literature is the absence of empirically validated triage prioritisation models that account for both alert severity and available analyst capacity. Existing papers propose heuristic prioritisation rules but do not validate them against measured analyst performance data.

2.7.2 Documentation and Incident Reporting

Poor documentation practices impede post-incident learning and regulatory compliance (Naseer et al. (2023)). The literature documents this challenge descriptively but offers few quantified measures of documentation quality or its operational impact. This is a notable gap: without quantified baselines, it is difficult to measure the effect of proposed improvements.

The absence of standardised documentation processes and templates is one of the main constraints. The unstructured or ad hoc documenting methods many SOC use result in inconsistently high-quality and comprehensive incident reports. This lack of standardisation can lead to important information being overlooked or inadequately documented, which makes it more difficult to analyse and draw lessons from past experiences properly. Creating and applying thorough documentation standards and templates adapted to the needs and processes of the SOC are necessary to resolve this issue Wang et al. (2021). Keeping accurate and current documentation throughout an incident's lifecycle is another difficulty. Documentation must be updated regularly whenever new observations and information about malware

incidents become available. Future analysis and incident response attempts may be affected by inadequate or out-of-date incident data resulting from a failure to properly track these revisions. This difficulty highlights how crucial it is to create procedures to guarantee that incident reports are updated regularly when new information becomes available Danneman and Gove (2022). Furthermore, the literature Onwubiko and Ouazzane (2020) points out the segmentation of documentation that occurs when incident-related data is dispersed among several tools, systems, and archives. As a result of this fragmentation, incident analysis and response processes may become inefficient and may encounter information gaps while attempting to collect and retrieve pertinent data. Developing methods for smooth integration and data aggregation across different systems and technologies used inside the SOC, or putting in place centralised incident management platforms, could be one way to solve this problem. Additional challenges to documenting and reporting procedures come from the sensitivity and confidentiality of incident-related data. To secure sensitive data and comply with the relevant laws and privacy legislation, SOC's must balance the need to keep full records for analysis and knowledge-sharing. Strong data encryption methods, safe information-sharing protocols, and access control mechanisms must be developed and implemented to address this issue Onwubiko and Ouazzane (2019).

For compliance requirements, legal investigations, and learning from previous incidents, accurate and thorough incident reporting and record-keeping are essential. However, documentation might be forgotten or left out in the hectic and high-stress environment of incident handling. Critical information must be recorded by SOC analysts, including the timeline of events, the actions performed, any signs of compromise, and any lessons discovered throughout the incident response. The lack of standardised documentation techniques, time constraints, knowledge transfer and retention, compliance and legal considerations, and knowledge transfer and retention may provide challenges in documenting incidents Bhatt et al. (2014); Kokulu, F. et al. (2019); Onwubiko and Ouazzane (2020); Yaacoub et al. (2022). SOC teams should make incident reporting and documentation a priority, as they are essential components of the incident-handling process to overcome these difficulties. To do this, clear instructions, training, and tools must be given for successful documentation, and it is crucial to stress the need for full and correct reporting to facilitate post-event analysis and enhance incident response skills.

2.7.3 Malware Detection and Identification

The rapidly evolving malware landscape (characterised by polymorphism, packing, and anti-analysis techniques) consistently challenges signature-based detection. The literature is

strong on characterising the detection problem but weaker on evaluating hybrid detection approaches under realistic SOC workload conditions.

One of the most important steps in incident handling is finding and correctly identifying malware. Nevertheless, malware authors use a variety of strategies to obfuscate their code and conceal their presence, making it difficult to identify and link malware to particular individuals or campaigns. Polymorphism, advanced encryption, or rootkit techniques are frequently used by sophisticated malware to avoid being discovered by conventional detection methods. Further complicating detection, malware may use fileless techniques or be embedded inside genuine files Hadiprakoso et al. (2020); O’Kane et al. (2011); Vielberth et al. (2020).

SOC teams must use cutting-edge threat intelligence, behavioural analysis, machine learning, and sandboxing technologies to improve their detection skills.

Keeping up with the quickly changing malware landscape is one of the main issues SOC’s confront. It is getting harder for SOC’s to keep detection signatures and analysis tools up to date due to the constant release of new malware variants and sophisticated evasion tactics used by threat actors. To overcome this difficulty, adaptive and automated methods must be used to analyse and recognise fresh malware samples. Additionally, threat intelligence feeds must be integrated to remain up-to-date on the most recent malware trends Aslan and Yilmaz (2021). Another major issue that businesses face is accurately classifying malware samples. Malware classification errors may result in incorrect or inadequate incident response plans, which may increase the effects of an attack. Improving the accuracy of malware classification requires the development of strong methodologies that make use of machine-learning algorithms Schofield et al. (2021). SOC’s face substantial hurdles in detecting and identifying malware due to the constant inflow of new malware variants. To effectively counter the ever-changing threat landscape associated with malware, machine learning-based intelligent defences must be implemented Nadeem et al. (2022). The sharp increase in malware incidents every year makes malware attacks more frequent and serious, underlining the necessity for sophisticated threat intelligence and detection systems Soltani et al. (2022). Staffing, finance, and technology limits are just a few of the resources that frequently make it difficult for SOC’s to handle malware occurrences efficiently, which affects their entire incident response capabilities Darem (2022). The SOC’s capacity to stay updated on the constantly changing threat landscape is severely restricted by a lack of qualified cybersecurity specialists, tight financial constraints, and an ongoing requirement for investment in state-of-the-art technologies Gupta et al. (2020). AI-assisted malware analysis and detection methods have been developed to improve cybersecurity workforce capabilities to solve these concerns Chae et al.

(2021). Moreover, SOC's may find it difficult to properly cooperate during malware incidents due to communication and collaboration issues. Various teams and stakeholders must work seamlessly together, both inside and outside the company, to respond to incidents effectively. Communication breakdowns or isolated activities can cause bottlenecks, inefficiencies, and security risks. Novel deep learning approach-based frameworks for malware detection have been presented to improve incident response coordination and increase the effectiveness of incident response teams Darem (2022).

2.7.4 Fragmented Tool Ecosystems and Investigation Latency

A recurring operational problem, observed in multiple practitioner reports and several academic studies, is the need to query multiple threat-intelligence platforms sequentially to investigate a single IoC. This introduces measurable latency and resource overhead. Critically, no peer-reviewed study quantifies this latency under controlled conditions. This gap directly motivates the empirical baseline measurements reported in Chapter 6.

2.7.5 Resource and Staffing Constraints

The global cybersecurity skills shortage is well-documented (ISC2 (2023)). SOC's in small and medium-sized organisations are particularly affected, operating with insufficient Tier 2 and Tier 3 analyst capacity. The literature identifies automation as a mitigation but does not adequately address the risk of automation failures or the skills required to configure and maintain automated systems.

Resource limitations present a serious obstacle for SOC teams handling malware incidents. SOC analysts frequently have to manage many incidents at once while also handling other duties, such as keeping track of security alerts and carrying out proactive threat hunting. Incident response activities may be hampered by inadequate staffing, funding, and technological resources Ahmad et al. (2021); János and Dai (2018); Kokulu, F. et al. (2019); Rajapakse et al. (2022).

One of the biggest issues facing SOC's worldwide is the lack of qualified cybersecurity specialists. Recent studies have highlighted the lack of skilled analysts, incident responders, and malware researchers Goupil et al. (2022). The demand for these individuals far outweighs the supply, which can ultimately affect the SOC's capacity to react to malware crises quickly. Understaffed SOC's, increasing workloads, and potential employee burnout are all consequences of this situation. To meet this problem, there needs to be a concentrated effort put into cultivating cybersecurity expertise and creating plans to draw and keep qualified workers

in the SOC environment. Furthermore, Tchernetsky and Vogt in a recent study Tchernetsky and Vogt (2023) pointed out that financial limitations provide yet another major barrier for SOCs. Budgets for cybersecurity operations are often restricted in many organisations, which might limit access to the necessary tools, technology, and resources needed for the efficient management of malware incidents. This restriction makes it more difficult for SOCs to maintain current threat information feeds, invest in cutting-edge detection and analysis tools, and give staff members extensive training opportunities. Fighting for more financing for cybersecurity and highlighting the possible financial and reputational consequences of insufficient malware incident management capabilities are necessary to overcome this obstacle. A diversified strategy is needed to address the difficulties SOCs face in keeping up with the constantly evolving malware landscape. SOCs may be less successful in identifying and analysing complex malware variants due to technological constraints. It is essential to have access to state-of-the-art technologies like automated analysis platforms, memory forensics tools, and sophisticated sandboxing environments to overcome this Aslan and Yilmaz (2021). These technologies are essential for increasing the effectiveness of incident response and malware detection Jin et al. (2022). Financial limitations and the swift advancement of technology increase the difficulties SOCs have in preserving a strong technological foundation. To tackle this, it is essential to investigate economic solutions, utilise open-source tools, and establish protocols for ongoing advancement in technology Park et al. (2022). SOCs may use their technology resources and make sure they continue to be effective in preventing evolving cyber threats by implementing such strategies. Resource constraints also make it more difficult for SOCs to work together and exchange intelligence with outside parties, which influences the efficient handling of malware incidents. Comprehensive threat intelligence requires teamwork and information exchange with other business partners and law enforcement authorities Oh et al. (2021). To improve incident response skills and close knowledge gaps, it is important to explore cost-effective collaboration mechanisms, use existing information-sharing platforms, and promote public-private collaborations Zheng et al. (2022).

Organisations should provide the SOC with the resources to handle this issue, including skilled staff and cutting-edge security resources, as well as adequate training to enable effective incident handling.

2.7.6 Communication and Collaboration Deficits

Fragmented communication between SOC tiers and between SOC and other organisational functions (IT operations, legal, executive) is consistently identified as slowing incident resolution. The literature offers largely prescriptive recommendations (e.g., establish com-

munication protocols) without empirical evidence of their effectiveness. Successful malware incident response requires effective coordination and communication both within the SOC and with external parties. To obtain the data they need, plan responses, and come to wise conclusions, SOC teams must collaborate closely with many departments, including IT, legal, and senior management. However, due to organisational barriers, a lack of standardised communication routes, or the participation of third-party vendors, creating seamless collaboration can be difficult Ioannou (2019); Nyre-Yu (2019); Onwubiko and Ouazzane (2020); Vielberth et al. (2020).

To meet these obstacles, SOC teams should create effective lines of communication, put incident response procedures into place, and promote a spirit of cooperation and information sharing.

Important issues that need to be addressed in malware incident management are the difficulties in sharing incident-related information and encouraging cooperation between diverse teams. Delays and misunderstandings result from unclear procedures for exchanging incident-related data Hall et al. (2018). This shows how important it is for stakeholders involved in malware incident management to have established communication methods in place to guarantee accurate and timely information flow. To respond effectively to malware events, several different teams must interact with one another. Collaboration can be affected by organisational barriers, competing agendas, and cultural differences, which can lead to gaps or redundancies in incident response operations Hofhuis et al. (2018). Establishing cross-functional teams, keeping open lines of communication, and making sure everyone is aware of their roles and responsibilities during malware attacks are crucial steps in overcoming this difficulty. It has been demonstrated that successful coordination and collaboration amongst heterogeneous teams improve project outcomes and innovation performance Xia et al. (2021). Improving innovation performance in global operations is largely dependent on collaborative knowledge sharing Xia et al. (2021). Moreover, the development of well-defined team dynamics via instruments such as the RACI matrix and Team Charter can improve the efficacy of the team Brower et al. (2021). In situations such as incident response, it is essential to motivate team members to collaborate across boundaries and exchange information Chen et al. (2014).

2.7.7 Incident Triage and Prioritisation

The absence of standardised, evidence-based triage criteria means that prioritisation decisions are often based on analyst intuition rather than systematic risk assessment. CyBOK (Section 8.7.2) notes that effective incident response requires structured prioritisation based on severity,

business impact, and asset criticality. The literature reviewed does not provide validated prioritisation frameworks tested in operational SOC environments.

2.7.8 Rapidly Changing Malware Environment

The fact that malware is always developing is one of the major difficulties SOCs encounter when responding to malware incidents. To exploit vulnerabilities and avoid detection, malicious actors constantly create brand-new, highly developed malware variants. For SOC analysts who must keep up with the most recent malware evasion techniques, trends, and strategies, this rapid evolution presents obstacles Afreen et al. (2020); Beqqali et al. (2019); Islam et al. (2019); Kayani and Saeed (2021); Murali et al. (2020).

Given that malware is dynamic, SOC teams must take proactive measures to foresee and successfully address new threats.

Threat actors are constantly creating and releasing new malware strains with sophisticated obfuscation techniques, polymorphic behaviours, and evasion mechanisms, as demonstrated by Darem et al. (2021). SOCs face a great deal of difficulty because of the evolving malware landscape. To stay abreast of the most recent threats, SOCs must regularly upgrade their threat information, analytic tools, and detection signatures. This challenge also involves the growing diversity and complexity of malware delivery routes. Roseline et al. (2020) highlights the increasing prevalence of malware spreading via a variety of channels, including drive-by downloads, hacked websites, email attachments and social engineering techniques. The variety of attack vectors makes it more difficult for SOCs to detect and prevent malware since they must be vigilant on several fronts and use several approaches to malware incident management. The adoption of emerging technologies and their quick growth have greatly increased the attack surfaces and vulnerabilities available to malware developers. Detection and response now face particular obstacles due to the growth of Internet of Things (IoT) devices, cloud computing environments, and the growing need for containerised applications Alrubayyi et al. (2021). To keep up with ever-changing technological environments and malicious methods used by hostile actors, SOCs must constantly modify their strategies and tools Aslan et al. (2021). For SOCs, handling the increasing number of malware samples and incident data is also a major difficulty. Security teams may become overwhelmed by the constant stream of new malware variations and the need to evaluate and correlate enormous volumes of data, which could cause delays or omissions in incident response Vurdelja et al. (2021). Scalable and automated analytic methods, as well as effective data management and storage solutions, are essential to meet this challenge and guarantee thorough and rapid malware event management Aurangzeb and Aleem (2023).

2.7.9 Incident Containment and Remediation

To reduce the impact on systems and data, malware events must be quickly contained and remedied. However, isolating compromised systems, eliminating malware, and repairing damaged assets can be difficult and time-consuming tasks. Malware may have already spread throughout the network or infected crucial systems, involving rigorous analysis and a restoration strategy. Organisations must also make sure that incident response procedures do not interfere with crucial business operations McLaughlin and Gogan (2018); Nyre-Yu et al. (2019); Nyre-Yu (2019).

SOC teams may speed up the containment and remediation process by developing effective incident response playbooks, automation capabilities, and regular incident response practices.

The complexity of systems, network segments, and software components can be difficult for corporations to manage, which can make it difficult to accurately identify and contain malware and put remedial procedures in place Gupta et al. (2020). These problems are made more difficult by the modern malware strains' rapid dissemination and persistence, which can quickly infiltrate networks, leave a lasting impression, and use evasion techniques to elude detection and containment efforts Chae et al. (2021). The containment and remediation procedures may become more difficult if they lead to serious harm or data breaches before they are discovered. SOCs must maintain comprehensive asset inventories, create specialised incident response plans, and implement centralised monitoring and control systems to address these issues (Gupta et al., 2020). Furthermore, to enable quick containment and reduce the effect of malware incidents, efficient network segmentation, real-time monitoring, and automated incident response capabilities are essential Chae et al. (2021). Utilising cutting-edge technology can improve malware detection and attribution, assisting in the timely discovery and response to malware threats. Examples of these technologies include machine learning algorithms, deep learning-based methodologies, and AI-assisted malware analysis Gorment et al. (2023); Wang and Fu (2024). Moreover, utilising tools like the Malware Analysis and Intelligence Tool can offer valuable insights by utilising state-of-the-art malware analysers and open-source databases to generate intelligence reports sourced from cyber threat intelligence Yucel et al. (2021). Understanding the propagation dynamics of malware, such as through epidemiology-based models and complex network analysis, is essential for devising effective strategies to combat malware threats Li et al. (2020); Zhu and Huang (2021). Furthermore, the development of innovative approaches like dynamic defence mechanisms and intelligent vision-based malware detection using deep random forest paradigms can strengthen cybersecurity defences against evolving malware threats Roseline et al. (2020); Xiao et al. (2020).

The literature has identified some obstacles that SOC's must overcome to control and remediate incidents effectively. Data restoration and system recovery following malware attacks present a big obstacle. Critical data can be corrupted or encrypted by malware types, making conventional backup techniques useless Meyer and Métile (2023). It can take a lot of resources to check the integrity of backup data and ensure malware remnants are absent Meyer and Métile (2023). Meyer and Métile Meyer and Métile (2023) advocate the implementation of secure offline backups, improved data recovery techniques, and meticulous verification processes as solutions to this issue. SOC's can improve their incident response capabilities and provide business continuity in the face of cyber threats by tackling issues like handling complex environments, preventing sophisticated malware threats, putting strong data recovery procedures in place, and encouraging efficient coordination and collaboration. According to Elradi et al. Elradi et al. (2021), this underlines how crucial it is to take preventative action in incident response to safeguard the security and integrity of organisational systems.

2.7.10 Triage and Incident Prioritisation

Malware-related incidents are among the numerous issues that SOC's frequently deal with. These incidents need to be prioritised and triaged according to their potential importance and impact. It can be difficult to assess the urgency and severity of each occurrence since the earliest signs of compromise might not accurately reflect the full scope of the issue Danquah (2020); Vielberth et al. (2020).

SOC analysts must create effective workflows and standards for classifying incidents according to criteria such as the effect on vital systems, the potential for data exfiltration, or legal constraints. Effective resource management and incident prioritisation are crucial for promptly mitigating the most serious risks. Assessing the severity and impact of a malware attack precisely requires in-depth knowledge of the company's risk profile, asset criticality, and possible outcomes Dawodu et al. (2023). This assessment procedure can be laborious and prone to prejudice or human error, which makes it more difficult to effectively prioritise and triage issues. Incident response involves more than just detecting and classifying incidents. The efficient and effective handling of cyberattacks poses considerable challenges for security Incident Response (IR) teams Elradi et al. (2021). These teams' time-consuming procedures may limit their overall efficacy as they work to identify, contain, and prevent cyberattacks. Effective incident management requires that contextual data and threat intelligence be incorporated into the triage and prioritisation process. Threat intelligence offers important insights into the danger and urgency of incidents through the analysis of elements such as malware sources, capabilities, and indicators of compromise Islam et al. (2022). Accessing

and integrating this data, however, can be difficult, especially if threat intelligence is out of date Van Haastrecht et al. (2021). The challenges of triage and incident prioritisation are made worse by resource constraints. SOCs frequently struggle with manpower and equipment shortages, which makes it difficult for them to fully investigate and rank incidents. Due to this restriction, low-priority events may take up precious resources while high-priority incidents are ignored, which could compromise the efficacy of malware incident management initiatives Preuveneers et al. (2020). Businesses can improve their incident response capabilities by utilising technologies such as cyber threat intelligence platforms and machine learning to tackle these difficulties. Organisations can enhance their comprehension and ability to defend against sophisticated cyber attacks by sharing threat intelligence data and employing cutting-edge analytical tools like machine learning algorithms Ali et al. (2022). The challenges identified within SOCs are grouped and described in Table 2.9, encompassing the three core dimensions of people, technologies, and processes. This comprehensive taxonomy encompasses 38 distinct challenges, with 16 challenges about personnel-related factors, 14 challenges associated with technological considerations, and 8 challenges related to operational processes within SOCs. This categorisation facilitates a structured analysis of the multifaceted challenges confronting SOCs, enabling a nuanced understanding of the human, technical, and procedural factors that influence the effectiveness of malware incident management efforts.

Several informative analyses can be made using the data shown in Table 2.9, which groups the difficulties that have been found based on the people, technology, and process aspects within SOCs:

- (a) The largest category involves difficulties related to personnel: It is clear that human elements are important to the success of malware incident management because 16 issues related to the "people" part of SOCs have been highlighted. These difficulties could include problems like shortages of staff and skills, inadequate training, poor communication, and difficulties related to the organisational culture. Optimising the performance and capacities of SOC teams requires addressing these personnel-related issues.
- (b) There are significant technological challenges as well. The fourteen technology-related difficulties focus on the substantial influence that systems, tools, and technical infrastructure have on the management of malware incidents. Limitations in detection and analysis capabilities, integration difficulties, scalability problems, and the requirement for ongoing technological improvements to keep up with changing threats are a few examples of these hurdles. To effectively tackle malware threats, SOCs must be equipped

Challenges	People	Technology	Process
Lack of standardised documentation processes			✓
Unstructured or ad hoc documenting methods			✓
Documentation not regularly updated			✓
Dispersed incident-related data			✓
High-stress environment	✓		
Time constraints	✓		✓
Polymorphism		✓	
Advanced encryption		✓	
Use of rootkit techniques		✓	
Use fileless techniques		✓	
Quickly changing malware landscape		✓	
Outdated detection signatures and analysis tools		✓	
Constant release of new malware variants		✓	
Sophisticated evasion tactics		✓	
Incorrect or inadequate incident response plans			✓
Lack of qualified cybersecurity specialists	✓		
Tight financial constraints	✓	✓	✓
Communication and collaboration issues	✓		
The adoption of emerging technologies and their quick growth have greatly increased the attack surfaces		✓	
Vulnerabilities available to malware developers		✓	
Security teams become overwhelmed by the constant stream of new malware variations and need to evaluate and correlate enormous volumes of data	✓		
Understaffed SOC's	✓		
Increasing workloads	✓		
Potential employee burnout	✓		
Lack of standardised communication routes	✓		
Difficulties in sharing incident-related information	✓		
Lack of cooperation between diverse teams	✓		
The complexity of systems, network segments, and software components		✓	
Lack of comprehensive asset inventories, and specialised incident response plans		✓	✓
Difficulty in assessing the urgency and severity of each occurrence	✓		
Laborious severity and impact assessment prone to prejudice or a human mistake	✓		
Time-consuming procedures limiting the overall efficacy of identifying, containing, and preventing attacks	✓		
Frequent struggle with manpower and equipment shortages	✓	✓	
Total identified challenges	16	14	8

Table 2.9 SOC Challenges by Category of SOC Architecture

with the technological capabilities required by effective technology management and strategic investments.

- (c) Process-related challenges are less common but still important: Even though they make up the smallest group, the eight process-related difficulties shouldn't be taken lightly. For incident response activities inside SOC's to be coordinated and consistent, effective and established processes are essential. These difficulties could include things like a deficiency of standardised protocols, poor protocols for collaborating, or flaws in operational workflows. The entire malware incident management lifecycle within SOC's can be simplified and optimised by addressing process-related issues.
- (d) Relationship between categories: It's important to understand that the difficulties that fall under these headings are frequently connected rather than isolated. For example, issues about personnel can affect how well technologies are used, and issues of processes can affect how well SOC teams work and how technical solutions are implemented. To improve malware incident management capabilities within SOC's, comprehensive strategies that take into account the interdependencies across these categories must be developed.

Table 2.10 presents a categorisation of the literature based on the challenges faced by SOC's. The issues of Documentation and Incident Reporting, as well as Triage and Incident prioritisation, have gathered 8 contributions each. Malware Detection and Identification, Rapidly Changing Malware Environment, and Resource Limitations have individually received 11 contributions. Communication and Collaboration challenges have been cited 10 times in the literature. Furthermore, the problems associated with Incident Containment and Remediation have been addressed in 16 contributions, indicating a significant focus on this aspect of SOC operations. The information provided in Table 2.10 allows us to perform the following analysis and observations:

- (a) The greatest number of contributions (16) has been made to the issues of incident containment and remediation, indicating that these facets of SOC operations rank among the most important and difficult ones, demanding substantial attention from practitioners and researchers.
- (b) With eleven contributions, the topics of malware detection and identification, the rapidly changing malware environment, and resource limitations show that SOC's are also very concerned about the constantly changing threat landscape and the limitations imposed by limited resources.

- (c) Ten times, the topic of communication and collaboration issues has been brought up, underlining how crucial it is for different stakeholders participating in cybersecurity operations to effectively share information and coordinate efforts.
- (d) Eight contributions have been made to the categories of Documentation and Incident Reporting, Triage, and Incident Prioritisation. To guarantee effective and efficient incident response procedures inside SOC, several areas are essential.
- (e) The way contributions are distributed among the problems suggests an integrated focus on multiple facets of SOC operations, which reflects the complexity of cybersecurity issues that businesses must deal with.
- (f) Researchers and practitioners are aware of the complexity of SOC operations and the necessity to address a wide variety of difficulties to enhance the overall effectiveness of cybersecurity efforts, as evidenced by the relatively high number of contributions across various categories.

This analysis shows the necessity of tackling the problems encountered by SOC with a thorough, coordinated strategy. It also points out how crucial it is to continue conducting research and sharing knowledge to create novel solutions and best practices that can improve SOC's ability to identify, address, and mitigate cyber risks.

Table 2.10 List of literature in SOC Challenges Section

SOC Challenges	Literature	Year	Total
Documentation and Incident Reporting	Wang et al. (2021)	2021	7
	Danneman and Gove (2022)	2022	
	Onwubiko and Ouazzane (2020)	2020	
	Onwubiko and Ouazzane (2019)	2019	
	Onwubiko and Ouazzane (2019)	2019	
	Kokulu, F. et al. (2019)	2019	
	Yaacoub et al. (2022)	2022	
	Bhatt et al. (2014)	2014	
Continued on next page			

Table 2.10 (continued)

SOC Challenges	Literature	Year	Total
Malware Detection and Identification	Hadiprakoso et al. (2020)	2020	11
	O’Kane et al. (2011)	2011	
	Vielberth et al. (2020)	2020	
	Aslan and Yilmaz (2021)	2021	
	Schofield et al. (2021)	2021	
	Nadeem et al. (2022)	2022	
	Soltani et al. (2022)	2022	
	Darem (2022)	2022	
	Gupta et al. (2020)	2020	
	Chae et al. (2021)	2021	
	Darem (2022)	2022	
Rapidly Changing Malware Environment	Islam et al. (2019)	2019	11
	Murali et al. (2020)	2020	
	Afreen et al. (2020)	2020	
	Kayani and Saeed (2021)	2021	
	Beqqali et al. (2019)	2019	
	Darem et al. (2021)	2021	
	Roseline et al. (2020)	2020	
	Alrubayyi et al. (2021)	2021	
	Aslan et al. (2021)	2021	
	Vurdelja et al. (2021)	2021	
	Aurangzeb and Aleem (2023)	2023	
Continued on next page			

Table 2.10 (continued)

SOC Challenges	Literature	Year	Total
Resource Limitations	Kokulu, F. et al. (2019)	2019	11
	Ahmad et al. (2021)	2021	
	János and Dai (2018)	2018	
	Rajapakse et al. (2022)	2022	
	Goupil et al. (2022)	2022	
	Tcharnetsky and Vogt (2023)	2023	
	Aslan and Yilmaz (2021)	2021	
	Jin et al. (2022)	2022	
	Park et al. (2022)	2022	
	Oh et al. (2021)	2021	
	Zheng et al. (2022)	2022	
	Communication and Collaboration	Onwubiko and Ouazzane (2020)	
		2020	
Vielberth et al. (2020)		2019	
Nyre-Yu (2019)		2019	
Ioannou (2019)		2018	
Hall et al. (2018)		2018	
Hofhuis et al. (2018)		2021	
Xia et al. (2021)		2021	
Xia et al. (2021)		2021	
Brower et al. (2021)		2014	
Chen et al. (2014)			
Continued on next page			

Table 2.10 (continued)

SOC Challenges	Literature	Year	Total
Incident Containment and Remediation	Nyre-Yu et al. (2019)	2019	16
	McLaughlin and Gogan (2018)	2018	
		2019	
	Nyre-Yu (2019)	2018	
	McLaughlin and Gogan (2018)	2020	
		2021	
	Gupta et al. (2020)	2021	
	Chae et al. (2021)	2023	
	Chae et al. (2021)	2024	
	Gormont et al. (2023)	2021	
	Wang and Fu (2024)	2021	
	Yucel et al. (2021)	2020	
	Zhu and Huang (2021)	2020	
	Li et al. (2020)	2020	
	Xiao et al. (2020)	2023	
	Roseline et al. (2020)	2021	
	Meyer and Métille (2023)		
	Elradi et al. (2021)		
Triage and Incident Prioritisation	Vielberth et al. (2020)	2020	8
	Danquah (2020)	2020	
	Dawodu et al. (2023)	2023	
	Elradi et al. (2021)	2021	
	Islam et al. (2022)	2022	
	Van Haastrecht et al. (2021)	2021	
	Preuveneers et al. (2020)	2020	
	Ali et al. (2022)	2022	

Table 2.10 List of Literature Related to SOC Challenges

Critiques of research papers on SOC challenges highlight areas that require improvement for deeper comprehension and practical relevance. One perspective focuses on a lack of empirical data and real-world confirmation. Some publications lack specific examples or case studies from real SOC contexts and rely primarily on theoretical frameworks. Future research

should prioritise empirical investigations, industry collaboration, and case-based analysis to address this concern. Dealing with these critiques will improve the value and pertinence of research papers on SOC challenges, making certain that they offer useful information for SOC professionals and make an effective contribution to the field. Researchers can connect the gap between theoretical ideas and practical performance by combining in-depth analyses, practical validation and interacting with experts in the sector, eventually improving the success of SOC operations overall.

2.8 Criteria-Driven Comparison of Existing Industry Tools

To contextualise the contribution of *SecOps_360*, a structured comparison of representative industry tools and platforms addressing SOC malware incident management was conducted. The comparison was guided by four criteria derived from the research gap analysis: (C1) IoC investigation integration (ability to query multiple threat-intelligence sources from a single interface); (C2) workflow optimisation (structured support for the analyst's investigation workflow); (C3) resource efficiency (computational overhead during investigation); and (C4) open accessibility (availability for use without requiring a commercial licence). Tab 2.11 presents the comparison.

The comparison reveals that commercial SIEM and SOAR platforms (Splunk, QRadar, XSOAR, Sentinel) provide strong workflow support and broad IoC integration but require substantial commercial investment and are resource-intensive. Open-source tools (TheHive, MISP) offer a lower cost but provide limited integration between IoC investigation and workflow management. VirusTotal Enterprise provides a strong IoC lookup but no investigation workflow. No existing tool combines unified multi-source IoC investigation, lightweight resource consumption, and open accessibility for SOC analysts, the specific gap addressed by *SecOps_360*. This comparison is revisited in the evaluation chapter (Chapter 6, Section 6.5).

2.9 Research Gap

The SLR reveals a clearly defined and convincingly justified gap in the existing body of knowledge. Three specific gaps are identified:

Gap 1 Quantified investigation latency: No peer-reviewed study has produced controlled, quantified measurements of the time and computational resources consumed by multi-platform IoC investigation in a simulated or operational SOC environment. This thesis addresses this gap directly through the scenario-based evaluation reported in Chapter 6.

Platform / Tool	C1: IoC Integration	C2: Workflow Support	C3: Resource Efficiency	C4: Open Access	Primary Function
Splunk SIEM	Partial (via add-ons)	Strong (dashboards, alerts)	Moderate (resource-intensive)	No (commercial)	Log aggregation, correlation, and alerting
IBM QRadar	Partial (via apps)	Strong (case management)	Moderate	No (commercial)	SIEM with threat intelligence
Palo Alto XSOAR	Strong (playbook-driven)	Very strong (full SOAR)	Moderate	No (commercial)	Incident orchestration, automation
Microsoft Sentinel	Partial (connector-based)	Strong	Cloud-dependent	No (commercial)	Cloud-native SIEM/SOAR
TheHive / Cortex	Moderate (Cortex analysers)	Moderate (case management)	Low-moderate	Yes (open source)	Incident case management + analysis
VirusTotal Enterprise	Strong (multi-source IoC)	Limited (lookup only)	Low (API-based)	No (commercial API)	IoC reputation lookup
MISP	Moderate (threat sharing)	Moderate	Low	Yes (open source)	Threat intelligence sharing
SecOps_360 (this work)	Strong (unified OSINT API)	Moderate (workflow-focused)	Low (lightweight web app)	Yes (prototype)	Unified IoC investigation for SOC analysts

Table 2.11 Criteria-Driven Comparison of Existing SOC Platforms and SecOps360

Gap 2 Unified, lightweight IoC investigation platform: Existing platforms either provide broad IoC integration with high resource cost (commercial SIEM/SOAR), or lightweight access with limited workflow integration (standalone lookup tools). No tool designed specifically for SOC analyst workflows (combining multi-source OSINT IoC investigation with a structured investigation workflow in a resource-efficient, openly accessible form) was identified. *SecOps_360* directly addresses this gap.

Gap 3 Evidence-based platform design from literature synthesis: Existing platforms are designed primarily around engineering and product requirements rather than from a systematic synthesis of the academic literature on SOC operational challenges. This thesis demonstrates a design science approach that derives platform requirements directly from the SLR findings, producing a traceable chain of evidence from identified challenges to design decisions.

The following suggestions are aimed at enhancing the handling of malware incidents within SOC. They are based on the challenges discussed previously. They can help SOC teams improve their overall cybersecurity posture, tighten their incident response procedures, and better minimise the effects of malware incidents.

- Effective Incident Analysis and Detection Tools

To improve their incident response capabilities, SOC. must use powerful event detection and analysis technologies. SOC teams can proactively identify and respond to evolving malware threats by investing in threat intelligence platforms, advanced malware detection systems, and security analytics and automation technologies. With the use of these technologies, analysts can analyse indicators of compromise (IOCs), do behavioural analysis, and automate incident triage, greatly enhancing their capacity to quickly identify, assess, and deal with malware occurrences.

- Perpetual Learning and Training

Maintaining SOC analysts' knowledge and abilities to deal with malware incidents successfully requires ongoing education and training. Analysts may stay updated with the most recent malware trends, attack methodologies, and incident-handling best practices by implementing ongoing training programmes, allowing certifications and professional growth, and creating an environment that promotes learning inside the SOC. This guarantees that they have the tools necessary to swiftly and effectively detect, examine, and respond to malware events.

- Improvements in Information Sharing and Collaboration

Effective malware incident handling requires cooperation and information exchange between SOC teams, external stakeholders, and industry peers. SOC teams can gain from shared expertise and insights by promoting improved collaboration and creating routes for information sharing. Sharing threat intelligence, best practices, and lessons learnt from prior incidents is all part of this. A community-oriented approach to incident handling is fostered through improved collaboration and information sharing, which enables SOC teams to stay ahead of new threats and respond to malware attacks more successfully.

- Standard Operating Procedures and Playbooks for Incident Response

Standard operating processes and incident response playbooks that are established are crucial for improving incident-handling processes within the SOC. SOC teams may ensure consistent and efficient handling of malware incidents by creating incident response playbooks adapted to various types of malware incidents, setting incident category and prioritisation criteria, and routinely evaluating and updating the playbooks. This enables an organised and effective incident response process by allowing analysts to follow planned actions, escalation procedures, and decision trees.

- Regular Drill for Incident Response

Periodic incident response drills are necessary for evaluating and enhancing the incident-handling skills of SOC teams. By running simulated malware incident situations, SOC analysts can test the effectiveness of incident response playbooks and SOPs, practise their response techniques, and determine areas for improvement. Regular training sessions encourage cooperation, coordination, and teamwork among SOC analysts, which improves their capacity to respond to malware incidents in the real world quickly and effectively.

- Systems for tracking and managing incidents

To maintain a disciplined and organised approach to incident handling, the SOC must implement incident tracking and management systems. SOC teams may efficiently track and manage the lifecycle of malware incidents by implementing incident tracking technologies or utilising current security information and event management (SIEM) solutions. This involves gathering incident facts, monitoring progress, assigning duties, and maintaining appropriate documentation to ensure that events are handled effectively and to promote effective communication among SOC analysts.

- Post-Incidental Analysis

Retrospective Incident analysis is essential for perpetual operation process improvement and learning from previous incidents. SOC teams can identify the causes of incidents, assess how well they responded and put corrective measures in place to stop them from happening again by performing full post-incident assessments. The SOC's overall incident-handling capabilities are improved via incident aftermath analysis, which also helps to share information and improve incident response playbooks.

By putting these recommendations into practice, SOC teams can improve their overall cybersecurity posture and their incident handling skills and reduce the impact of malware incidents. With the help of these preventative measures, SOC analysts will be more prepared to identify, evaluate, and respond to malware incidents, strengthening the security and resilience of the organisational environment. Another criticism focuses on the depth of some papers' analyses. They might just give explanations of SOC difficulties, omitting the complexities and interdependencies at play. Researchers can offer a more comprehensive and in-depth analysis by looking at the complex relationships between various difficulties and taking into account their combined impact on SOC operations. This criticism emphasises the need for comprehensive studies that investigate the various aspects of SOC issues to guide the creation of effective solutions.

2.10 Chapter Summary

This chapter presented a Systematic Literature Review conducted according to a documented protocol. The review identified the core operational lifecycle of SOC incident management, the three-pillar architectural model, the principal malware detection and analysis techniques, and the major challenges confronting operational SOCs. A critical assessment of the literature revealed several gaps in existing research, particularly the absence of quantified investigation latency measurements and the lack of an integrated, lightweight, workflow-aware IoC investigation platform. A criteria-driven comparison of twelve existing industry tools confirmed that no current solution simultaneously addresses the identified gaps. These findings establish the research gap that *SecOps_360* is designed to address.

In conducting the systematic literature review, a rigorous methodology was followed to ensure the reliability and validity of the study. Table 2.12 provides a detailed methodology for carrying out an organised study of the literature on malware incident management in SOCs. The table is organised into two main columns: the first column lists the typical steps involved in a systematic review process, while the second column provides specific recommendations and suggestions tailored to the research area of study. The table provides useful suggestions and elements unique to the context of malware incident management in SOCs for each phase. These suggestions are meant to assist in successfully navigating the process of systematic reviews, ensuring a rigorous method of locating, assessing, and integrating pertinent material in this crucial field of cybersecurity operations.

The methodology involved a systematic review of the literature on malware incident handling and challenges faced by SOCs. The research studies were selected based on their relevance and reliability in providing insights into the topic. The data collected from the literature review were analysed to identify common themes and patterns related to malware incident handling and challenges faced by SOCs. The analysis helped to provide a comprehensive understanding of the topic and identify the best practices for handling malware incidents in SOCs. The literature identified several common challenges faced by SOCs in handling malware incidents. These challenges include the increasing sophistication of malware attacks, the volume and complexity of incidents, the lack of skilled personnel, inadequate incident response plans, and coordination and communication issues. The search also identified several best practices for handling malware incidents, including implementing automated malware detection and removal tools, developing incident response playbooks specific to malware incidents, conducting regular malware training and awareness sessions for SOC staff, leveraging threat intelligence, integrating endpoint detection and response

Systematic Review Steps	Recommendations
Define research questions and objectives	- What are the current challenges and best practices in malware incident management within SOC? - How can SOC detect, analyse, and respond to malware incidents effectively? - What are the critical factors that influence the success of malware incident management processes in SOC?
Develop the search string and	Use keywords and Boolean operators like: ("malware" OR "virus" OR "trojan" OR "worm") AND ("incident management" OR "incident response" OR "incident handling") ("Security Operations Centre" OR "SOC") AND ("malware" OR "cybersecurity" OR "incident response") "malware incident handling" "security operations centres" "challenges" "security threats" "Cybersecurity incident response"
Identify relevant databases and sources	- Electronic databases: IEEE Xplore, ACM Digital Library, ScienceDirect, Web of Science, Scopus, Google Scholar, Springer, Wiley Online Library - Cybersecurity-specific databases: DBLP, CVE Details, MITRE CVE - Conference proceedings: RSA Conference, Black Hat, DEF CON, USENIX Security
Set inclusion and exclusion criteria	Include: - Peer-reviewed journal articles, conference papers, and book chapters - Studies focused on malware incident management in SOC - Studies published within a specific date range (last 10 years) Exclude: - Studies not available in full-text - Studies not published in English - Studies not focused on SOC operations or malware incidents
Conduct a literature search	Perform the search strategy across the identified databases and sources, and document the search process thoroughly.
Extract and synthesise data	Extract appropriate data from included studies, such as: - Malware incident management processes and frameworks - Malware detection and analysis techniques - Incident response and remediation strategies - Challenges and limitations in malware incident management
Assess the quality of the study and risk of bias	Use appropriate quality assessment tools such as QUADAS-2 or Cochrane for diagnostic accuracy studies or checklists based on the study designs
Synthesise and analyse the data	-Narrative synthesis: Describe and summarise the findings from the included studies -Quantitative synthesis (if applicable): Conduct meta-analyses for specific outcomes or variables related to malware incident management
Interpret and report the findings	- Summarise the key findings, strengths, and limitations of the review - Discuss the implications for SOC operations, malware incident management processes, and future research directions

Table 2.12 Steps and Recommendations for a Systematic Literature Review on Malware Incident Management in SOC

(EDR) solutions, creating a dedicated team of malware analysts, proactively monitoring of network traffic and systems, conducting regular penetration testing, establishing a clear incident response plan with defined roles and responsibilities, and regularly reviewing and updating SOC policies and procedures. The literature review also highlighted the importance of a well-designed incident-handling process that involves several stages, including detection, classification, containment, analysis, response, and recovery. Each stage requires a range of tools and techniques, as well as skilled personnel who can identify and isolate malware-infected systems, analyse the malware, and determine its impact on the organisation. The literature review also identified the need for SOCs to adapt and evolve in response to new and emerging threats. Malware attacks are constantly evolving, and SOCs need to keep up with the latest trends and technologies to ensure that they can effectively handle malware incidents. The literature review also highlighted the importance of regular training and awareness sessions for SOC staff to ensure that they are up to date with the latest tools and techniques for handling malware incidents. SOC staff should also be trained on the latest trends and technologies in malware attacks to ensure that they can effectively respond to new and emerging threats. The literature review also emphasised the importance of collaboration and communication between different teams within the organisation. SOCs need to work closely with other teams, such as IT and legal, to ensure that they can effectively handle malware incidents and minimise their impact on the organisation.

In conclusion, the methodology used involved a systematic review of the literature on malware incident handling and challenges faced by SOCs. The literature review helped to identify common themes and patterns related to the topic and identify the best practices for handling malware incidents in SOCs. The literature review also highlighted the importance of a well-designed incident-handling process, regular training and awareness sessions for SOC staff, and collaboration and communication between different teams within the organisation.

SOCs face a wide range of challenges when it comes to handling malware incidents, including technical, operational, and people-related challenges. These challenges include identifying unknown malware, keeping up with the latest software, reverse engineering, evasive techniques, limited resources, handling large numbers of malware samples, keeping up with the latest malware detection techniques, lack of visibility, collaboration, automation, scalability, identifying malware variants, handling encrypted or obfuscated malware, managing false positives, and balancing speed and accuracy. SOCs also face challenges with staffing, lack of training, burnout, limited resources, keeping up with the latest threats, limited collaboration, limited ability to measure and report on performance, limited ability to perform root cause analysis, managing false positives, balancing speed and accuracy, lack of incident response and incident management maturity, and limited ability to manage the incident

response process. To address these challenges, SOC's should invest in advanced technologies such as machine learning and artificial intelligence to improve their malware detection and analysis capabilities. There is a need to utilise threat intelligence and conduct regular training and exercises to improve incident response capabilities. SOC's must maintain a strong incident management plan that includes incident response procedures, incident escalation processes, incident reporting, and incident recovery procedures. Collaboration with other organisations, industry groups, and law enforcement agencies can also help to stay current with the latest malware trends and techniques and effectively respond to cyberattacks. Additionally, SOC's can benefit from automating certain aspects of malware analysis, but it requires significant investment in tools and resources and can be difficult to implement and manage effectively. It is important to consider integrating their incident response processes with other workflows within the organisation to ensure a seamless and efficient response to cyberattacks. Another important aspect is to invest in developing a skilled and knowledgeable team. SOC's should participate in recruiting and retaining qualified personnel, especially those with advanced technical skills and experience. Additionally, provide ongoing training and development opportunities for their staff to keep them current with the latest threats and trends in the cybersecurity field. In terms of data management, SOC's need to implement a comprehensive data management strategy, including data governance, data quality, and data security, to ensure that they have accurate, complete, and timely data to support their incident response and analysis efforts. Finally, SOC's have to consider implementing a continuous improvement programme to evaluate their current processes, tools, and personnel and identify areas for improvement. This should include regular performance assessments, incident response exercises, and threat-hunting campaigns to identify any gaps and opportunities for improvement.

Fig 2.8 represents the total literature identified in each sub-section of the chapter, as shown by the pie chart.

The analysis of the identified literature reveals a diverse range of research interests within the domain of SOC's. The largest portion of the identified literature, comprising 42.1% of the total publications, is dedicated to "SOC Challenges". This meaningful emphasis on challenges focuses on the complex nature of SOC operations and the continuing efforts to address the issues and obstacles these SOC's face. Researchers and practitioners have recognised the importance of understanding and mitigating the challenges confronting SOC's to enhance their effectiveness and efficiency.

The second largest category, accounting for 24.2% of the publications, focuses on "Malware Incident Handling in SOC Environment". This field of study points out how important SOC's are to the identification, evaluation, and handling of malware incidents. This

is an important field of research because of the dynamic nature of malicious software threats and the need to have strong incident-handling capabilities.

"SOC Architecture" constitutes the third largest portion, representing 20.2% of the identified literature. Research on the planning, execution, and optimisation of SOC infrastructures falls under this topic. The architectural foundation of SOC is becoming more and more crucial to guarantee resilience, scalability, and efficient threat detection and response capabilities as the complexity and scope of cyber threats continue to rise.

The smallest portion, comprising 12.9% of the publications, is dedicated to "Incident Management". Despite its lower volume, this field of study is also essential for SOC because it deals with the processes, guidelines, and best practices for efficiently handling security issues. Effective incident handling is essential to good SOC operations, which makes this a valuable field of research.

Overall, the analysis of the identified literature by scope reveals a multifaceted research landscape, with a predominant focus on understanding and addressing the challenges faced by SOC, followed by an emphasis on malware incident handling, SOC architecture, and incident management. This distribution of research efforts highlights the diverse and interconnected aspects of SOC operations, underscoring the need for a comprehensive and holistic approach to enhancing the effectiveness and resilience of these critical cybersecurity entities

Fig 2.9 represents the chronology of publications related to the research topic over the years from 2010 to 2024. The data shows a gradual increase in the number of publications from 2010 to 2018, indicating a growing interest and research activity in the field. The number of publications remained relatively low, ranging from 1 to 6, until 2018. A significant spike was observed in 2020, with 32 publications, suggesting a substantial surge in research efforts or a heightened focus on the topic during that year. This could potentially be attributed to emerging trends, technological advancements, or specific events that prompted increased scholarly attention. Following the peak in 2020, the number of publications declined but remained relatively high, with 30 publications in 2022 and 12 publications in 2024. This sustained level of research activity implies that the topic remained relevant and garnered continued interest from the academic and research community. Overall, the graph illustrates the evolution of research efforts in the given field, highlighting periods of increased and decreased publication activity. The spike in 2020 and the subsequent sustained interest suggest the topic's significance and the potential impact of external factors or developments on research trends.

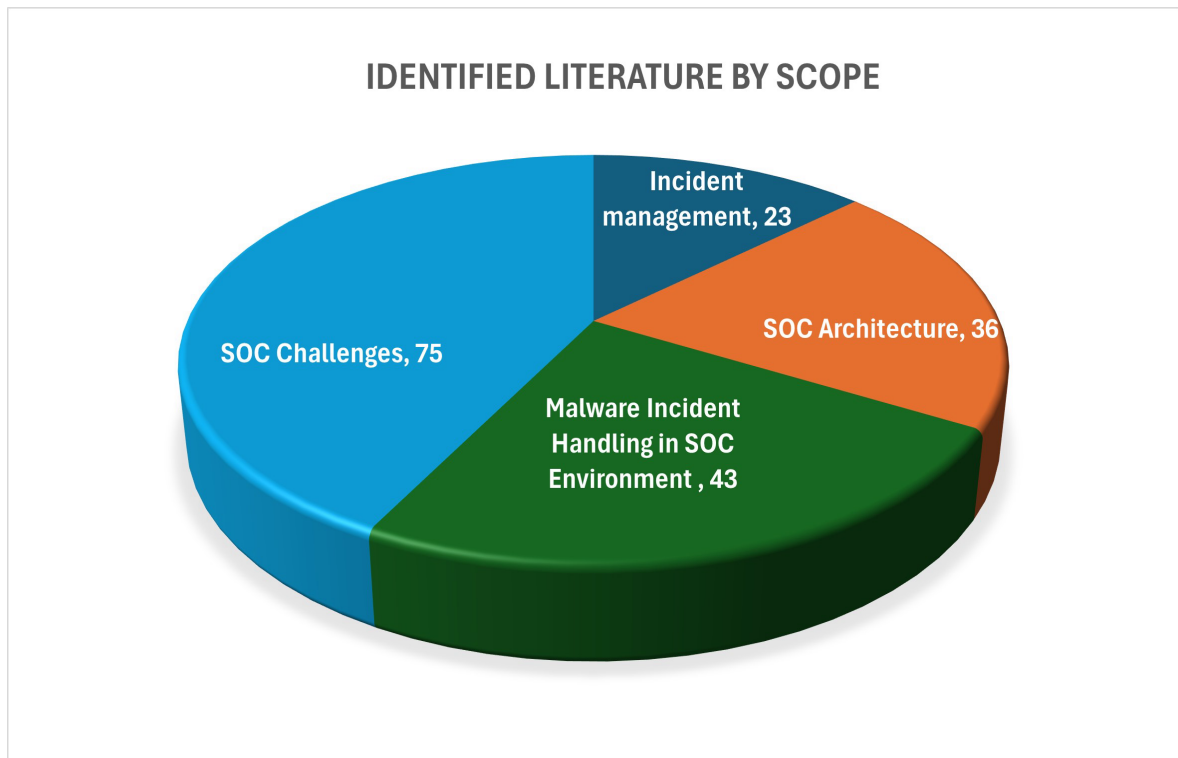


Fig. 2.8 Identified Literature by Scope

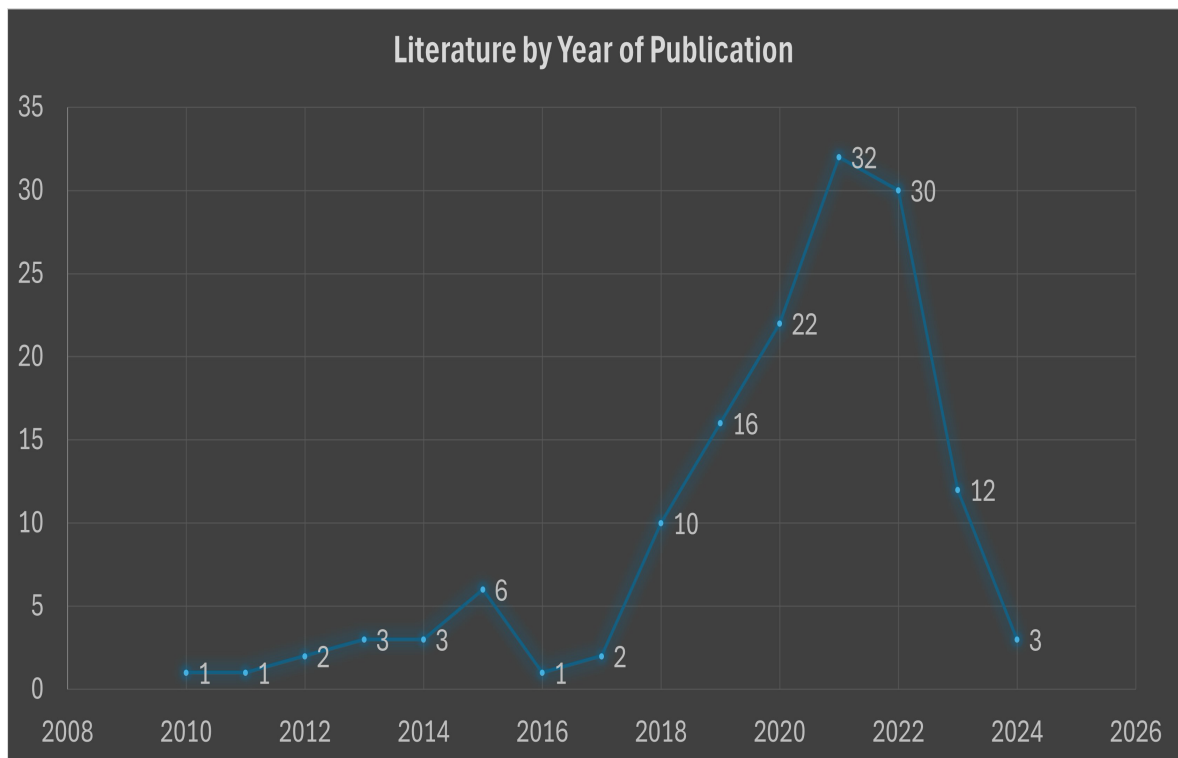


Fig. 2.9 Chronology of Literature

Chapter 3

Research Methodology

3.1 Introduction

This chapter provides a comprehensive account of the research methodology employed in this study. Methodological decisions are presented in relation to each Research Objective, with explicit justification for the chosen approach. Where similar work was identified in the literature review, those influences on methodological choices are noted. The chapter also addresses ethical considerations specific to the research process.

3.2 Research Philosophy and Framework

The research adopts a pragmatist philosophical stance (Creswell and Creswell (2017); Saunders (2009)). Pragmatism is appropriate because the primary aim is to produce a useful, working solution to an identified practical problem, the inefficiency of malware incident investigation in SOC environments, rather than to test a formal hypothesis or construct a grounded theory. Pragmatism permits the combination of qualitative and quantitative methods as determined by the research question, rather than committing a priori to a single paradigm (Saunders (2009)).

The research framework follows the Design Science Research Methodology (DSRM) as proposed by Peffers et al. (2007). DSRM is well established in information systems research and has been applied in prior cybersecurity platform research (Jiang et al. (2023)). It is particularly appropriate here because the primary output is an artefact (the *SecOps_360* platform) rather than a purely descriptive or explanatory contribution. The DSRM cycle comprises six phases: Problem Identification and Motivation; Definition of Objectives; Design and Development; Demonstration; Evaluation; and Communication. Table 3.1 maps each DSRM phase to the Research Objectives and methods employed in this thesis.

3.3 Systematic Literature Review (Supporting RO1)

The SLR protocol is described in full in Chapter 2 (Section 2.2). The SLR methodology was chosen because it provides a transparent, replicable approach to synthesising existing knowledge, superior to a traditional narrative review for the purposes of establishing a credible research gap (Kitchenham et al. (2022)). The SLR findings directly informed the design requirements for *SecOps_360*, creating an explicit chain of evidence from identified challenge to design decision.

DSRM Phase	Research Objective	Method Employed	Justification
Problem Identification	RO1	Systematic Literature Review (SLR)	SLR provides a rigorous, reproducible basis for characterising the problem space and establishing the research gap (Kitchenham et al. (2022))
Objective Definition	RO2	Requirements Analysis from SLR findings	Functional and non-functional requirements are derived directly from identified challenges, ensuring traceability from problem to solution
Design & Development	RO2, RO3	Conceptual design + prototype implementation	Consistent with DSRM, a similar approach was used in Mutemwa, M. et al. (2018) and Alves et al. (2021)
Demonstration	RO3	Controlled scenario-based testing	Demonstrates that the artefact addresses the defined problem in a representative context
Evaluation	RO4	Quantitative performance measurement + comparative analysis	Provides empirical evidence of effectiveness relative to the baseline and to existing tools
Communication	All ROs	Thesis; peer-reviewed conference paper (ICISSP 2024)	Ensures research is accessible to the academic and practitioner community

Table 3.1 Research Framework Mapped to Objectives and Methods

3.4 Conceptual Design Approach (Supporting RO2)

The conceptual design phase translates the requirements derived from the SLR into a system architecture. The design approach is informed by established software engineering practices, specifically the use of UML (Unified Modelling Language) diagrams for structural and behavioural modelling, and High-Level Design (HLD) and Low-Level Design (LLD) specifications for progressive elaboration of the architecture. This approach is consistent with prior work on security platform design (Mutemwa, M. et al. (2018)).

Requirements were gathered through a structured analysis of the SLR findings: each identified SOC challenge was mapped to one or more functional requirements for the platform, ensuring that the design is driven by evidence rather than assumption. Non-functional requirements, including performance, usability, and compatibility, were derived from the SOC operational context identified in the literature.

3.5 Implementation Approach (Supporting RO3)

SecOps_360 was implemented as a web-based application with a backend API (PHP, Symfony framework) and a single-page frontend (Angular). This technology stack was chosen for three reasons:

- (a) web-based deployment minimises client-side resource requirements, addressing the resource constraint challenge identified in the SLR
- (b) the REST API architecture facilitates future extension with additional threat-intelligence integrations.
- (c) both technologies are mature, well-documented, and in widespread use in SOC tooling, reducing the likelihood of compatibility issues in operational environments.

3.6 Evaluation Methodology (Supporting RO4)

The evaluation combines three approaches: functional testing, compatibility testing, and scenario-based performance testing. This multi-method approach is justified by the nature of the research artefact: a functional platform requires both verification (does it work correctly?) and validation (does it solve the problem it was designed to address?).

3.6.1 Functional and Compatibility Testing

Functional testing verifies that each platform component behaves in accordance with its specification. Compatibility testing verifies that the platform operates correctly across target browsers and operating systems. These standard software quality assurance methods are employed in consistent manner with prior security platform evaluations (see Chapter 6, Section 6.5 for the justification relative to related work).

3.6.2 Scenario-Based Performance Testing

Two controlled scenarios are used to evaluate the platform's operational impact. Scenario 1 represents the baseline: a manual, multi-platform IoC investigation without *SecOps_360*. Scenario 2 represents the intervention: an equivalent IoC investigation conducted through *SecOps_360*. The primary metrics are: total elapsed investigation time; CPU utilisation, and memory consumption. These metrics directly address the research gap identified in Section 2.9 (Gap 1). The scenario design follows the approach used by Mutemwa, M. et al. (2018) to evaluate SOC incident management process improvements. A significant limitation of this evaluation is that it does not employ the statistical methods that would be required to draw generalisable conclusions from the results. Section 3.7 addresses this limitation explicitly, and Chapter 6 (Section 6.6) proposes a programme of future experimental work with SOC analyst participants using appropriate empirical and statistical methods.

3.7 Ethical Considerations in the Research Process

Ethical considerations in the research process extend beyond those addressed in Chapter 1 (Section 1.3.3). The following considerations are specific to the methodology.

3.7.1 Data Handling in the SLR

The SLR draws exclusively on publicly available peer-reviewed literature and authoritative grey literature retrieved from five electronic databases (IEEE Xplore, ACM Digital Library, ScienceDirect, Scopus, and Google Scholar) together with established cybersecurity bodies including NIST, MITRE, SANS, and CyBOK. No personal data was collected, processed, or stored at any stage; the review involved no human participants, no surveys, and no interviews. This approach ensures the SLR process raises no ethical concerns beyond those already addressed by the original authors.

3.7.2 Use of Threat Intelligence APIs

Scenario-based testing uses real IoC data (IP addresses, domain names, file hashes) drawn from publicly disclosed threat intelligence feeds. All IoCs used in testing were confirmed to be from previously disclosed incidents and not attributable to specific living individuals. API usage complied with the terms of service of AbuseIPDB, VirusTotal, and MetaDefender. Rate limits were respected, and no bulk data downloads were performed beyond those permitted by the respective terms of service.

3.7.3 Research Integrity and Bias

We acknowledge the potential for confirmation bias in scenario design: as the designer of *SecOps_360*, there is an inherent risk of designing scenarios that favour the platform. This risk was partially mitigated by using the same IoC dataset in both scenarios and measuring objective performance metrics (time, CPU, memory) rather than subjective assessments. Nevertheless, the absence of an independent evaluator and the use of a controlled rather than operational environment represent limitations that are acknowledged transparently in Chapter 7.

3.7.4 Responsible Disclosure

SecOps_360 integrates multiple third-party APIs. During development, no security vulnerabilities were identified in these APIs. Had vulnerabilities been discovered, they would have been reported to the relevant vendors in accordance with responsible disclosure principles as per CyBOK (Section 3.13.3). The source code of *SecOps_360* is provided in the appendices; however, proprietary rule sets and detection signatures are not included to avoid potential misuse.

3.8 Limitations of the Methodology

The principal methodological limitations are:

- (a) The evaluation uses a controlled environment rather than a live operational SOC, which limits the ecological validity of the performance measurements.
- (b) The scenario-based testing does not employ statistical inferential methods, precluding generalisation of the quantitative results.
- (c) The SLR, whilst systematic, was conducted by a single researcher, introducing the risk of selection bias in quality assessment.

(d) The platform prototype was not subjected to user acceptance testing with practising SOC analysts, which would be necessary to validate usability claims. These limitations are discussed further in Chapter 7, and a programme of future experimental work addressing them is proposed in Section 6.6.

3.9 Chapter Summary

This chapter has presented the research methodology governing this study. A pragmatist philosophy was adopted, operationalised through the Design Science Research Methodology. The SLR was conducted according to a documented protocol. Conceptual design, implementation, and evaluation methods were each justified in relation to the corresponding research objective. Ethical considerations specific to the research process were addressed. The principal methodological limitations were acknowledged. The following chapter presents the conceptual design of *SecOps_360*.

Chapter 4

Conceptual Design of the Framework

4.1 Introduction

Malware attacks pose a major problem in the constantly changing world of cybersecurity threats. To reduce the dangers brought on by these attacks and guarantee the security of vital systems and data, effective malware incident management is essential. By deploying the capabilities of contemporary online technologies and facilitating efficient communication and information exchange, the conceptual design of *SecOps_360*, a web-based platform, aspires to improve malware incident management in SOC.

The *SecOps_360* platform represents a comprehensive, integrated solution designed to centralise the diverse functions and workflows of SOC teams. *SecOps_360* aims to unify the core pillars of modern security operations: Threat Intelligence, Analysis, Security Engineering, Incident Management, Penetration Testing, and Management oversight. This platform addresses a critical challenge in contemporary cybersecurity operations: the fragmentation of security functions across disparate teams, tools, and workflows. As security threats grow increasingly sophisticated and the attack surface of organisations expands, traditional, fragmented approaches to security operations become inadequate. *SecOps_360* offers a paradigm shift by providing a centralised ecosystem where all security functions converge, allowing for enhanced coordination, improved information sharing, and optimised response capabilities. The conceptual design of *SecOps_360* encompasses a modular architecture that supports specialised security functions while maintaining seamless integration between components. Each module corresponds to a specific security domain, such as Threat Intelligence for monitoring and analysing potential threats, Analysis for detailed investigation of security incidents, Security Engineering for implementing and maintaining security controls, Incident Management for coordinated response to security breaches, Penetration Testing for proactive security assessment, and Management for resource allocation and performance tracking. By consolidating these functions within a unified platform, *SecOps_360* enables security teams to maintain comprehensive visibility across the security landscape, streamline workflows, reduce response times, and enhance overall security posture. The platform emphasises automation, standardisation, and knowledge sharing to optimise resource utilisation and improve operational efficiency. In the following sections, we will explore the architectural components, functional specifications, and integration capabilities of the *SecOps_360* platform, highlighting how this innovative solution addresses the evolving needs of modern SOC.

SecOps_360 would function as a central point for managing malware incidents, offering incident responders and SOC analysts an extensive set of capabilities and set of tools. A file analysis module modelled by external resources would be one of the main elements. By using a variety of antivirus engines and analysis tools, this module would enable analysts

to upload files for inspection to find possible malware risks. It would also incorporate Application Programming Interfaces (APIs) to access past data and intelligence gathered from the community, improving the platform's detection capabilities. The platform would involve an environment that communicates with external resources to improve malware analysis capabilities even more. With the help of this component, analysts would be able to run and monitor potentially dangerous files in a secure setting, giving them access to comprehensive reports on the behaviour of the file, its network activity, and other pertinent data. By integrating with the resources' analysis APIs, the platform would be able to make use of cutting-edge malware analysis methods and continually updated threat intelligence. Some advanced tools for investigating file hashes, IP addresses, and domain names via API for reputation checks will be included in the proposed platform. SOC analysts will be able to evaluate the reputation of files and network entities more quickly and accurately thanks to this integration, which will expedite the decision-making process during incident response. One of the platform's main goals would be to facilitate collaboration and information sharing, allowing SOC teams to work together in real-time, share insights, and coordinate operations. This web-based platform's conceptual design attempts to solve the difficulties SOCs encounter in efficiently handling malware incidents. The platform would improve the overall effectiveness and efficiency of malware incident management processes within SOCs by utilising contemporary web technologies and encouraging collaboration, information sharing, and automation. This would ultimately improve cybersecurity posture and resilience against evolving threats.

The benefit of *SecOps_360* is that SOC teams will operate in a single, fully integrated environment where all critical functions—threat intelligence, monitoring, incident response, engineering, and management—are seamlessly connected. Instead of juggling multiple disjointed tools, each team member gains instant access to the same real-time threat data, contextual insights, and alerts, ensuring that no important detail is missed and no time is wasted. This unified approach streamlines collaboration: analysts can investigate incidents with immediate access to historical data and correlation reports, while engineers receive precise, prioritised instructions to remediate vulnerabilities without back-and-forth delays. Incident managers benefit from transparent timelines, reducing response times and improving containment efficiency. Meanwhile, SOC managers have a bird's-eye view of operations, enabling them to monitor KPIs, track team performance, and allocate resources more effectively. By removing barriers between teams, eliminating duplicated efforts, and ensuring actionable intelligence reaches the right people at the right time, *SecOps_360* empowers SOC staff to work faster, smarter, and with greater confidence—ultimately improving the organisation's resilience against evolving cyber threats.

This conceptual design phase models the way the application will work. It identifies the technical aspects of the product. Some characteristics of the design process include a user interface, architecture as a high-level design, and detail as a low-level design.

4.2 Collecting and Analysing Project Requirements

Collecting and evaluating project requirements is an important first step in the design of a web-based platform, as it sets the stage for a successful project's development. This procedure entails a thorough analysis of the project scope, which establishes the parameters of the project, as well as the functional requirements, which specify the anticipated capabilities of the platform. Successful requirements engineering requires effective project management. Project success depends on the project scope, which includes components like resource estimation, project schedule, cost projections, and stakeholder expectations Hassan and Asghar (2021). Furthermore, it is stressed that effective project scope management requires a precise specification of the project.

To have a clear understanding of the system requirements, we propose a sample incident creation process in an SOC (Fig. 4.1). As illustrated in the flowchart, the SOC incident creation process outlines a comprehensive approach to identifying, assessing, and managing security incidents within an organisation. This systematic procedure ensures that potential threats are addressed efficiently and effectively, with appropriate escalation and communication at each stage.

The process begins with identifying a potential incident, typically through automated systems such as Security Information and Event Management (SIEM) tools, firewalls, or Intrusion Prevention/Detection Systems (IPS/IDS). Once a potential threat is detected, the SOC team moves to the triage stage, where they collect relevant data and perform an initial analysis to understand the nature and scope of the event.

Following triage, a critical decision point is reached: determining whether the event warrants classification as an incident. If it does not, the process concludes. However, if the event is deemed an incident, the process advances to the next phase, where the incident's priority is determined. This prioritisation is crucial, as it dictates the subsequent actions and response times.

Incidents are categorised into four priority levels in this example: P1, P2, P3, and P4, with P1 being the most critical. For high-priority incidents (P1 and P2), an immediate escalation protocol is initiated. This involves both calling and emailing stakeholders and management,

with a stringent Service Level Agreement (SLA) of 15 minutes to 1 hour for response. P3 incidents require notification of stakeholders and management via email within a 24-hour SLA, while P4 incidents have a more relaxed 72-hour SLA for email notifications.

Concurrently with the notification process, the SOC team creates detailed incident records. These records include assigning an incident number, documenting a summary of the event, and collecting all relevant evidence. This documentation is crucial for tracking the incident's progression and for any potential post-incident analysis or audits.

The next step involves transferring the incident to the stakeholder's Incident Response Team (IRT). The method of transfer varies based on the incident's priority. For P1 and P2 incidents, an urgent handover meeting is conducted to ensure the IRT is fully briefed on the situation. For lower-priority incidents (P3 and P4), the handover may be scheduled or communicated via email, allowing for a more flexible response time.

Throughout the entire process, there is ongoing communication with stakeholders to provide updates on the incident's status and any developments. This constant flow of information ensures all relevant parties are informed and can make necessary decisions or take appropriate actions.

The final stage of the process involves determining whether the incident has been resolved. If the incident is deemed resolved, it is formally closed, and the process ends. However, if the incident remains unresolved, the process cycles back to stakeholder communication, ensuring continuous attention and management until resolution is achieved.

This SOC incident creation process demonstrates a well-structured approach to cybersecurity incident management. Clearly defining steps, priorities, and communication protocols enables organisations to respond swiftly and effectively to security threats, minimising potential damage and ensuring a coordinated response across all relevant teams and stakeholders.

4.2.1 Functional Requirements

SecOps_360 will feature a user-friendly interface with clearly labelled entry fields for IP addresses, hash values, domain names, and file uploads. Each input field will be accompanied by descriptive labels to guide users in providing the necessary information for analysis and processing. For example, users will find labelled fields for entering IP addresses, hash values (such as MD5 or SHA-256), domain names, and options for uploading files containing relevant data. Validation mechanisms will be implemented to ensure that the data entered by users meets the required format and standards. For instance, the application will validate IP addresses to ensure they adhere to standard conventions, such as IPv4 or IPv6 formats. Similarly, hash values will be validated to verify their integrity and format, while domain

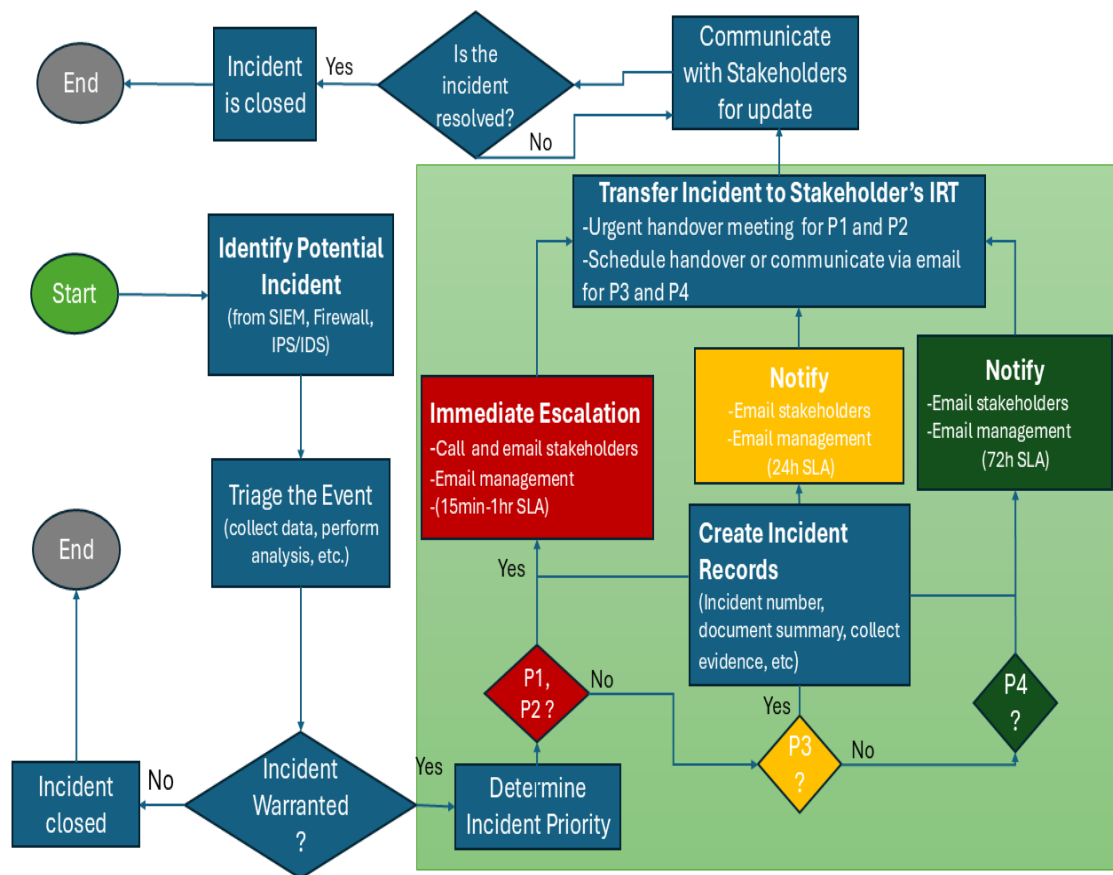


Fig. 4.1 SOC Incident Creation Process Sample

names will be checked to ensure proper formatting and validity. These validation checks will help maintain data integrity and accuracy throughout the analysis process. Users will have the ability to upload files containing relevant data for analysis directly within the application interface. The file upload functionality will support various file formats, such as documents, images, or executables, and will allow users to upload multiple files simultaneously. Progress indicators and error-handling mechanisms will be included to provide feedback to users during the upload process, ensuring a smooth and seamless experience. Additionally, the web application will integrate with external APIs to retrieve additional data or perform analysis tasks based on user input. Users will have the option to make API calls to external services directly from within the application, with the returned data being parsed and displayed within the interface. This integration will enable users to access a wide range of information and insights to augment their analysis efforts, enhancing the overall utility and effectiveness of the application.

4.2.2 Non-functional Requirements

These requirements focus on qualities such as performance, security, reliability, scalability, and usability, which are critical for the success of our project.

Performance:

SecOps_360 will be designed to deliver optimal performance, even under heavy load conditions. This includes fast response times for accessing and processing data, rendering user interfaces, and executing critical operations related to incident detection, analysis, and response. Performance benchmarks should be established to measure and monitor key performance indicators (KPIs) such as response time, throughput, and resource utilisation, ensuring that the platform meets the performance expectations of SOC personnel during high-pressure situations.

Reliability:

SecOps_360 should be highly reliable, with minimal downtime and disruptions to SOC operations. This involves ensuring high availability and fault tolerance through redundancy, failover mechanisms, and automated recovery processes. The platform should be resilient to hardware failures, network outages, and other unforeseen events, with mechanisms in place to detect and mitigate issues proactively. Comprehensive monitoring and alerting capabilities should be implemented to track system health and performance metrics, enabling rapid response to any incidents or anomalies.

Scalability:

As the demands on the platform grow over time, it should be able to scale seamlessly to accommodate increased user traffic, data volumes, and processing loads. Scalability should be built into the architecture of the platform, allowing for horizontal and vertical scaling as needed. SOC teams, including analysts, threat intelligence teams, and engineers, can integrate various tools and technologies into the proposed web-based platform to provide a single pane of glass for scalable malware incident management. Here are some examples of integration tools that can be considered

- Threat Intelligence Platforms (TIPs): Access to the most recent threat intelligence feeds, indicators of compromise (IoCs), and contextual data on new threats and adversaries can be obtained by integrating with TIPs like Abuseipdb, VirusTotal, CiscoTalos...
- Sandboxing and Malware Analysis Tools: The platform can analyse suspicious files, executables, and malware samples in-depth in a controlled environment by integrating with malware analysis tools such as Hybrid Analysis or Joe Sandbox. other examples that will not

be in the scope but could integrate the platform for scalability could be

- Incident Response and Case Management Tools: Integrating with incident response platforms like ServiceNow, Resilient, or D3 Security can streamline the incident management process, facilitate collaboration, and ensure consistent documentation and reporting.
 - Vulnerability Management Systems: Integrating with vulnerability scanners and management platforms like Nessus, Qualys, or Rapid7 can provide insights into potential vulnerabilities that could be exploited by malware, enabling proactive mitigation and patch management.
- Through the incorporation of these tools and technologies into the suggested web-based platform, SOC teams may take advantage of a unified interface that unifies diverse data sources, analytical proficiencies, and incident response features. This integrated strategy can help SOC teams manage malware incidents more efficiently, more scalably, and more collaboratively overall.

Usability:

SecOps_360 should be intuitive and user-friendly, with a clean and responsive interface that enables SOC personnel to perform their tasks efficiently and effectively. This includes clear navigation, informative feedback messages, and contextual help features to guide users through complex workflows. Usability testing should be conducted to gather feedback from users and identify areas for improvement, with iterative refinements made to the user interface based on user input. Ultimately, the goal is to create a platform that is easy to learn, easy to use and supports SOC personnel in their critical mission of malware incident management

4.2.3 Scope

To keep the project focused and feasible, given the limitations, it is essential to define the scope of the proposed web-based platform. The scope describes the bounds and restrictions on the target audience, integrations, and features of the platform. While reviewing incident notifications, SOC analysts perform a thorough investigation that may lead to a security breach, which they report as an incident. The investigation involves opening multiple search platforms to gather evidence on a suspicious file, IP address, domain, hash, etc., which can be resource-intensive and confusing to retrieve the right tab for any researched item. The web-based platform's main goal is to improve SOC's capacity to manage malware incidents. This includes many things, such as:

- Malware Detection and Analysis: By utilising integration with sandboxing environments, malware analysis tools, and threat intelligence feeds, the platform should offer methods for identifying and evaluating possible malware threats.
- Integration with Security Tools: For faster data collection and analysis, the platform should offer integration capabilities with a variety of security tools and systems from various SOC teams.
- Automation and Orchestration: To cut down on manual labour and speed up response times, the platform should enable the automation and orchestration of different tasks with the help of APIs.
- Indicator of Compromise (IoC) Analysis: The platform should allow for the analysis and reputation checking of various IoCs, such as IP addresses, file hashes, and domains, by integrating with threat intelligence platforms and reputation services.
- Integration of Threat Intelligence: To provide users with the most recent information regarding malware campaigns, adversaries, and emerging threats, the platform should integrate with reliable sources of threat intelligence.
- File Reputation Checking: By integrating with file analysis and reputation services, the platform should enable SOC analysts to swiftly evaluate the possible threat posed by a file and make well-informed judgements during incident response.

The scope ought to consider the target audience and their unique needs, taking into account academic cybersecurity researchers, IT security specialists, and SOC staff with different degrees of experience and use cases. Although the goal of the proposed web-based platform is to improve SOC's ability to manage malware incidents, it is crucial to specify what is and is not included in the project's scope. Below is a list of possible topics or features that are outside the purview of this specific project that may be taken into consideration for future integration.

- Vulnerability Management: Unless expressly needed for incident response, extensive vulnerability scanning, assessment, and patch management features may be deemed out of scope because they are not directly related to malware incident management.
- Penetration Testing and Red Teaming: A platform primarily designed for malware incident management may not be able to do offensive security testing functions like penetration testing or red team operations.

- Compliance and Regulatory Reporting: If not directly related to malware incident management, creating reports and documentation for compliance with industry-specific legislation or standards (e.g., PCI-DSS, HIPAA, GDPR) may fall outside the project's focus.

4.3 Target audience

4.3.1 SOC Personnel

SOC employees are a company's first line of defence against cybersecurity threats Onwubiko and Ouazzane (2022a). These experts, who include SOC managers, incident responders, and security analysts, are vital to the real-time monitoring, detection, analysis, and handling of security issues. Monitoring security alerts, looking into potential threats, and identifying the right course of action to take in response are the responsibilities of security analysts. The responsibilities of incident responders include containing and reducing security issues, organising response activities, and expeditiously returning to regular operations. SOC managers supervise all aspects of the SOC's operations, such as hiring, training, budgeting, and strategic planning, and they make sure the SOC runs smoothly to safeguard the company's assets and data.

To effectively combat growing cyber threats, SOC staff need a varied combination of skills and expertise. They must have an in-depth comprehension of cybersecurity principles, technologies, and best practices. They should also be adept at using security tools and technologies like endpoint security solutions, intrusion detection systems (IDS), and SIEM (Security Information and Event Management) systems. Strong analytical abilities, meticulousness, and the capacity to act quickly and decisively under pressure are further requirements for SOC staff members. Strong interpersonal and teamwork abilities are also necessary because SOC employees frequently collaborate closely with departments like IT, legal, and compliance, as well as outside parties like cybersecurity vendors and law enforcement Nifakos et al. (2021).

It can be hard and difficult to work in a SOC as employees must be awake and responsive in the face of dynamic cyber threats. Through regular training, certifications, and practical experience, they must stay updated on the most recent developments in cybersecurity, threat intelligence, and attack strategies. SOC employees are motivated by a common commitment to defending the confidentiality, integrity, and availability of critical systems and information while defending their company's assets, data, and reputation against cyber threats, despite the demanding nature of their work Pedreira et al. (2021).

4.3.2 IT Security Professionals

IT security experts are essential to protecting a business's infrastructure and digital assets against cyberattacks. These experts, who include cybersecurity specialists, network administrators, and system administrators, are in charge of putting strong security measures in place and keeping them up to date to keep off cyberattacks, illegal access, and data breaches. To guarantee reliable and secure communication, network administrators supervise the setup and maintenance of the company's network infrastructure, which includes firewalls, routers, and switches. To prevent unauthorised access and safeguard sensitive data, system administrators are responsible for maintaining servers, workstations, and other IT systems, as well as installing security patches and access controls Nifakos et al. (2021).

The goal of the web-based platform is to meet the needs of IT security professionals, a critical target group with a range of duties and demands, to improve malware incident management. IT security experts are essential to maintaining the overall security posture and safeguarding the information assets of enterprises. Their responsibility is putting security controls, policies, and procedures into place and keeping them up to date to protect against a variety of cyber threats. The web-based platform provides IT security experts with numerous important advantages and capabilities, including collaboration, information sharing, reporting and analytics.

IT security professionals need to possess strong analytical skills, problem-solving talents, and attention to detail in addition to their technical expertise to address security vulnerabilities and incidents. This is done through training, certifications, and professional development opportunities. Through the web-based platform, IT security experts may leverage cooperation and automation to more effectively tackle malware incidents. Therefore, IT security experts are better equipped to safeguard vital information assets and reduce the dangers posed by malware threats. This enhances the overall cyber resilience and security posture of enterprises.

4.2.3. Academic Researchers in Cybersecurity

Academic cybersecurity researchers play a crucial role in the advancement of the discipline by conducting thorough investigations, delving into theory, and conducting real-world experiments. These experts, who are frequently found in academic institutions, research centres, and universities, perform empirical studies, create novel solutions, and share their findings through conferences and scholarly publications to advance our understanding of cybersecurity. Researchers investigate various aspects of cybersecurity, such as threat analysis, vulnerability assessment, risk management, and security technologies, to tackle new obstacles and create

effective defences against cyber threats.

Academic scholars in the fields of computer science, information security, cryptography, and data analytics are among those with a wide range of backgrounds and areas of competence in cybersecurity. They push the boundaries of knowledge and innovation in cybersecurity by using their specific expertise and research abilities to investigate new areas. To address difficult cybersecurity concerns, encourage interdisciplinary research, and encourage information exchange and collaboration within the cybersecurity community, university researchers frequently work with corporate partners, government organisations, and other academic institutions. Academic researchers not only conduct research but also have a significant impact on training the future generation of cybersecurity experts. To equip the workforce of future generations with the skills to address the constantly changing challenges they create and instruct cybersecurity courses, become mentors to students, and oversee graduate research projects. To make sure that educational programmes are in line with industry needs and best practices, academic scholars also contribute to the development of cybersecurity curricula, certification programmes, and professional standards. Academic cybersecurity researchers, in general, work as boosters for innovation, propelling developments in theory, practice, and instruction to improve the security of digital systems and infrastructure. The web-based platform provides numerous critical features and benefits for academic cybersecurity researchers, including:

- Working with Industry Professionals: The platform makes it easier for academic researchers to work with industry professionals, which promotes knowledge sharing and gives academics access to insightful information from people who deal with malware incidents daily.
- Testing and Experimentation Environment: Before implementing their suggested solutions, algorithms, or techniques in actual situations, researchers can assess and validate them in a simulated or sandboxed environment using the platform.
- Cooperation with Other Researchers: By integrating the platform with academic networks and resources, researchers from various institutions can work together more easily, encouraging interdisciplinary research and the sharing of concepts and techniques.

By meeting the needs of cybersecurity researchers at academic institutions, the web-based platform fosters a mutually beneficial interaction between academia and industry, allowing researchers to further the development of malware incident management procedures. In the

end, this partnership improves the overall cybersecurity posture by encouraging innovation and strengthening the feedback loop between practice and research.

4.4 System Detailed Design

The next stage in creating *SecOps_360* to improve malware incident management in SOC's is the system's detailed design phase. The primary goal of this phase is to ensure an effortless transition to the implementation and development stages by converting the high-level conceptual designs into precise technical specifications and architectural blueprints. Dissecting the system into its parts, outlining how they interact, and defining the technical specifications and limitations for each part are all aspects of the detailed design phase. Important topics covered in this phase include interface design, data flow, system architecture, and integration with third-party systems and APIs.

4.4.1 SOC Team Roles and Module Functionality

Understanding the Structure of SOC Teams Every SOC has different teams that handle specific jobs. *Threat Intelligence* teams collect and analyse information about security threats from various sources. They study how attackers work and what new threats are emerging, which helps the whole organisation make better security decisions. *Analysts* are the main workforce of the SOC. They spend their time looking at security alerts, investigating suspicious activity, and keeping track of what's happening with threats. They take complicated security data and turn it into useful information that others can understand and act on. *Security Engineers* look after all the technical systems that keep security running. They maintain security tools, make sure detection systems work properly, and set up automated responses to help the SOC work faster and more efficiently. *Incident Management* teams take charge when serious security problems occur. They coordinate responses between different teams, keep important people informed, and make sure everything runs smoothly until the incident is resolved. *Penetration Testers* try to find security weaknesses before real attackers do. They carry out controlled attacks and simulations to discover vulnerabilities, giving the organisation valuable information to improve its defences. *Managers* oversee everything, tracking how well the SOC is performing, allocating resources where they're needed, and ensuring the team meets its goals whilst maintaining proper staffing and equipment.

We have built specific modules for each team based on what they need to do their jobs effectively. Threat Intelligence teams get an OSINT Search tool for gathering open-source in-

telligence, Threat Intel Feeds & Analysis for processing external threat information, Malware & IOC Tracking for keeping records of known threats, and Threat Actor Profiling for building detailed profiles of attackers. Analysts have modules for Investigating Security Incidents to manage cases properly, Reporting and Documenting Incidents to keep detailed records, Tuning Requests to improve detection rules, and Real-Time Monitoring that works with SIEM systems for continuous threat watching. Security Engineers use Security Tool Management to control all security systems from one place, Rule Tuning & Alert Optimisation to reduce false alarms, Vulnerability Management & Patching to keep systems secure, and Security Automation through SOAR Integration for coordinated responses. Incident Management staff work with Playbook Execution & Use Case Management for consistent response procedures, Incident Response Coordination to synchronise teams, Post Incident Review & Reporting to learn from incidents, and Stakeholder tools to communicate effectively with important people. Penetration Testers get Phishing Campaign Simulation to test staff awareness, Internal & External Pen Testing for comprehensive security checks, Reporting Vulnerabilities to track fixes, and Red vs Blue Team Exercises to test how well defences work. Managers have KPI Tracking and Performance Reporting to monitor operations, plus Resource Allocation and Task Assignment features to distribute work effectively amongst team members.

Out of all these modules, only the OSINT Search module will be completely operational and tested. All the other modules are just demonstrations at the moment—they're not fully working systems. Whilst they show how the platform could expand and what other SOC functions could be added, they haven't been properly developed or tested as the OSINT Search module has. They're mainly there to illustrate how the *SecOps_360* design can be extended to meet different SOC needs and to show what could be built in the future. This approach helps us focus on testing the main platform ideas properly while keeping a clear plan for adding more functionality later. The modular design means I can develop and add new components systematically, building on the solid foundation I've already proven works with the OSINT Search module.

The platform emerges as a transformative solution 4.2, designed to break down traditional barriers and empower SOC teams with comprehensive, integrated threat intelligence. By providing a centralised hub for rapid analysis, correlation, and investigation, this platform transforms how cybersecurity professionals approach threat detection, investigation, and response. Each team within the SOC ecosystem (from frontline analysts to threat intelligence specialists, engineers, and incident responders) can leverage this powerful tool to enhance their operational effectiveness, reduce response times, and develop a more proactive approach to cybersecurity threats.

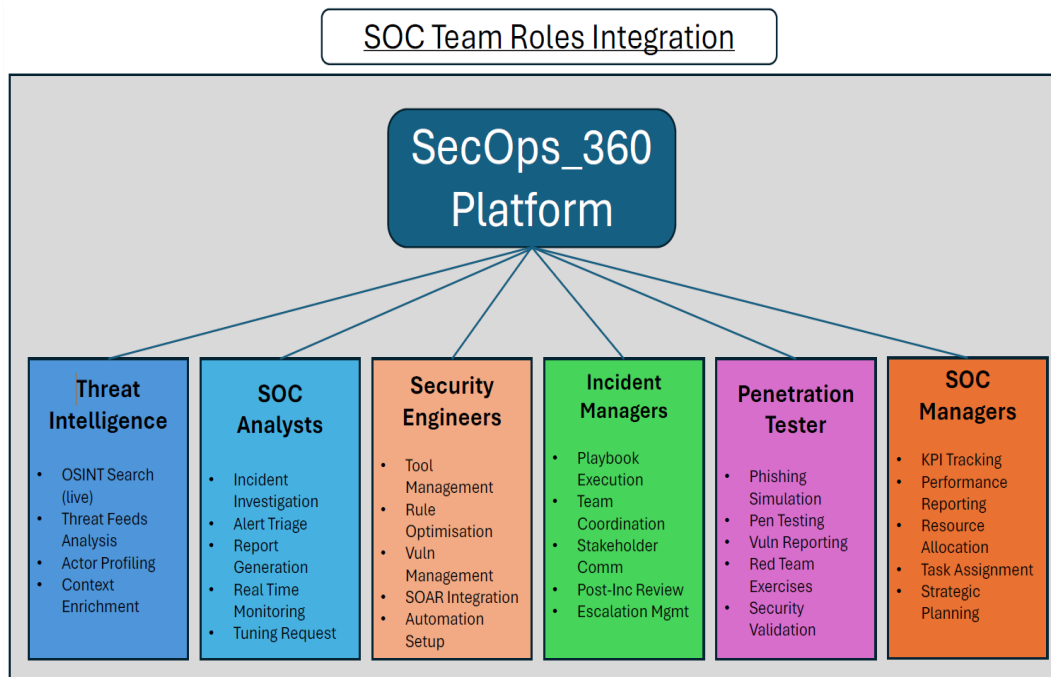


Fig. 4.2 SOC Teams' Collaboration With the Centralised Research Platform

4.4.2 UML and Process Flow Diagrams

The architectural framework of the *SecOps_360* platform has been thoroughly documented using Unified Modelling Language (UML) diagrams alongside process flow visualisations. These diagrams effectively demonstrate both the technical infrastructure and operational workflow integration within SOC environments. Such visual representations provide essential blueprints for comprehending component interactions and their correspondence with established SOC procedures.

Fig. 4.3 presents the UML class diagram, which outlines the fundamental architectural elements of the *SecOps_360* platform. This diagram reveals an object-oriented design methodology that facilitates modular capabilities and expandable implementation. Four principal classes establish the platform's structural foundation:

- *SOCPlatform Class*: Functions as the primary coordinator, incorporating crucial attributes such as platformId, user collections, and module arrays. This class oversees user verification, module accessibility controls, and external API connections, positioning itself as the main gateway between users and system operations.
- *User Class*: Encompasses individual SOC personnel through role-specific access management via UserRole classification. The class maintains user authentication details, authorisations, and operational status, allowing the platform to deliver customised

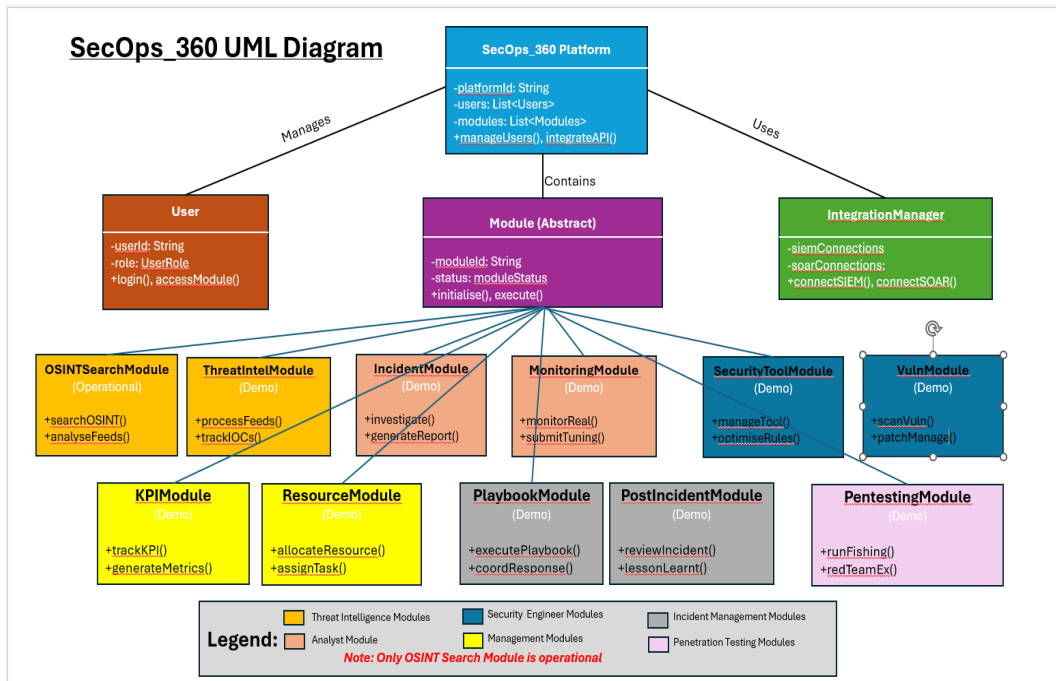


Fig. 4.3 *SecOps_360* UML Diagram

experiences according to particular SOC positions, including analysts, engineers, and supervisors.

- *Module (Abstract) Class*: Establishes the framework for all specialised SOC modules, specifying shared characteristics like moduleId, operational status, and permitted roles. This abstract structure guarantees uniform module functionality whilst permitting specialised features through inheritance, reinforcing the platform’s modular design philosophy.
- *IntegrationManager Class*: Manages external system connections, particularly SIEM and SOAR platforms. This class controls API endpoints and connection settings, facilitating smooth data exchange between *SecOps_360* and the current SOC infrastructure.

The diagram organises modules according to SOC team responsibilities, employing colour schemes to differentiate between operational and demonstration components: Threat Intelligence Modules (displayed in orange/yellow shades) incorporate the operational OSINT Search Module alongside demonstration components for threat feed processing, IOC monitoring, and threat actor analysis. This arrangement highlights the platform’s emphasis on external intelligence collection and threat contextualisation. Analyst Modules (presented in coral/salmon colours) include incident examination, monitoring, and reporting features that support fundamental analytical operations within SOC activities. Security Engineer Modules

(shown in blue variations) address tool administration, rule enhancement, and vulnerability management needs specific to technical security positions. Incident Management Modules (depicted in grey tones) concentrate on playbook implementation and post-incident analysis procedures vital for coordinated response operations. Management Modules (illustrated in yellow/green) offer KPI monitoring and resource distribution capabilities necessary for SOC supervision and performance assessment.

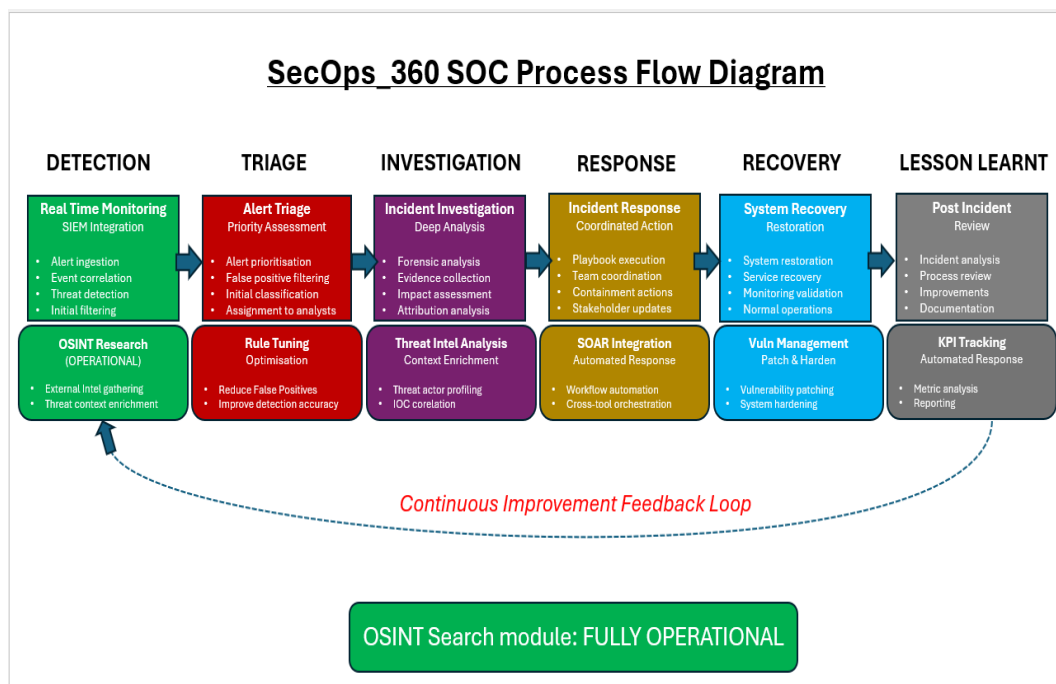


Fig. 4.4 SecOps360 SOC Process Flow Diagram

Fig 4.4's process flow diagram illustrates *SecOps_360*'s integration within conventional SOC operational workflows, covering six essential stages: Detection, Triage, Investigation, Response, Recovery, and Lessons Learnt. This alignment guarantees platform support for established SOC practices whilst improving efficiency through centralised functionality.

- **Detection Phase Integration:** The platform's Real-Time Monitoring component connects directly with SIEM systems for alert collection and event analysis. The operational OSINT Search module delivers external intelligence-gathering features that enhance detection precision and supply threat context during preliminary analysis stages.
- **Triage Phase Assistance:** Alert Triage and Rule Tuning components enable analysts to prioritise security incidents effectively whilst minimising false positives through

ongoing refinement. This stage demonstrates the platform's capacity to enhance signal clarity in high-volume SOC settings.

- Investigation Phase Improvement: Incident Investigation and Threat Intelligence Analysis components provide comprehensive analytical tools, combining forensic examination capabilities with threat intelligence context to support thorough threat evaluation and attribution analysis.
- Response Phase Coordination: The platform's Incident Response and SOAR Integration components orchestrate unified response activities through automated playbook implementation and inter-team communication, ensuring consistent and prompt incident handling.
- Recovery Phase Management: System Recovery and Vulnerability Management components assist restoration efforts whilst addressing fundamental security vulnerabilities that permitted initial compromise, establishing a thorough recovery methodology.
- Lessons Learnt Integration: Post-Incident Review and KPI Tracking components gather operational intelligence and performance indicators, feeding information back into detection and prevention capabilities through the continuous improvement cycle shown in the diagram.

The UML and process flow diagrams together illustrate several fundamental architectural concepts underlying the *SecOps_360* platform:

- Modularity: The abstract module framework permits independent development and implementation of specialised SOC features whilst maintaining consistent interfaces and
- Role-Based Access Management: User class design incorporating role-specific permissions ensures suitable access to specialised modules based on SOC team duties, preserving security whilst enabling cooperation.
- Expandable Architecture: The object-oriented design accommodates gradual feature development and platform growth without necessitating fundamental architectural modifications. behaviours throughout the platform.

The diagrams distinctly separate operational and demonstration modules, with the OS-INT Search Module constituting the sole fully tested and implemented functionality. This transparent evaluation provides practical expectations regarding current capabilities whilst

demonstrating the platform's comprehensive vision for SOC integration. The continuous improvement cycle displayed in the process flow diagram represents the platform's dedication to adaptive development based on operational experience and SOC team input, ensuring sustained relevance and effectiveness within evolving threat environments. These visual representations demonstrate *SecOps_360*'s potential to function as a comprehensive unified interface for SOC operations whilst maintaining practical implementation schedules and modular deployment approaches that minimise operational interruption during platform adoption.

4.4.3 High-Level Design

The primary objectives of the high-level design (HLD) phase are to define the OSINT Search Module architecture, identify the fundamental parts and subsystems, and establish how they interact and depend on one another. The groundwork for the next low-level design and implementation stages is laid during this phase. Fig4.5 represents the HLD that shows the general flow of the system. It shows a high-level design for the proposed threat investigation flow. The flow consists of 4 main components, with 3 being presented: IP Address Search, File Search, and Domain Search, with the input and output presented in Tab 4.1. The flow diagram shows how different inputs (IP addresses, files, file hashes and domains) are processed by their respective components, which then generate multiple outputs by leveraging various third-party API services for threat intelligence and analysis. The overall design aims to provide a structured approach for investigating potential threats within a SOC environment by integrating various data sources and analysis tools.

4.4.4 Low-Level Design

The Low-Level Design (LLD) stage turns high-level design concepts into tangible, implementable plans by digging into the specifics of the OSINT Search Module's architecture. This stage is essential for making sure that every system component has been well prepared and can be integrated into the system without any problems when it is put into use. Each module's unique characteristics, such as its relationships, data flow, and comprehensive functional specifications, are defined by the LLD. Fig. 4.6 represents the LLD of the system. It describes the component-level design process. A data validation is processed for a given IOC, followed by an API call to the relevant website. The reply from the server is in JavaScript Object Notation (JSON) format that will be displayed in HTML format on the user interface. Based on the information given in the figure, we can describe the flow and

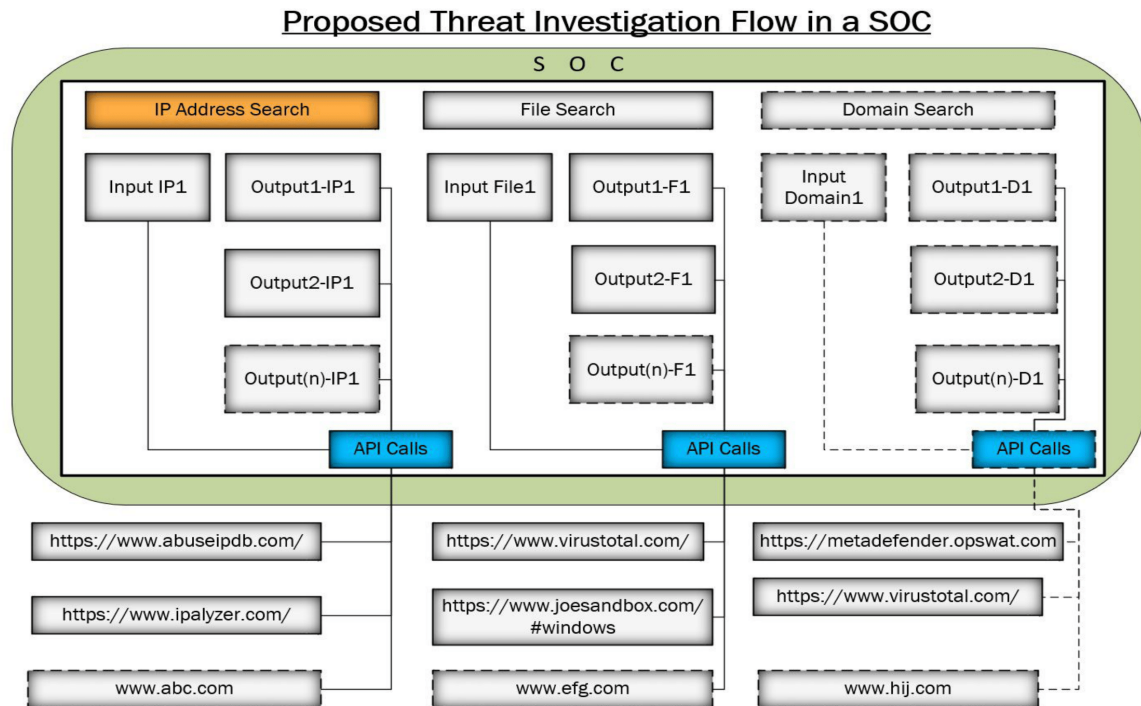


Fig. 4.5 Flowchart

components of the system in steps as provided in Tab 4.2, 4.3, 4.4, and 4.5. Note that the specific APIs and external resources used can vary depending on the requirements and the available integrations. Additionally, the system may incorporate additional features, such as file upload handling, data preprocessing, and result visualisation, based on the desired functionality.

4.5 System Tools and Software

The choice of design software tools is a pivotal aspect of system development, significantly influencing the project's efficiency and quality. In this subsection, we provide a comprehensive list of the design software tools that were instrumental in shaping the research project. We detail the specific tools employed for tasks such as conceptual modelling, system design, and documentation. This list includes software packages and platforms that facilitated the creation of diagrams, prototypes, and technical documentation.

Component	Input	Output	Description
IP Address Search	IP address	Output1-IP1, Output2-IP1, ..., Output(n)-IP1 (multiple outputs related to the input IP address)	This component makes API calls to https://www.abuseipdb.com/ and https://www.ipalyzer.com/ for IP address intelligence.
Domain Search	Domain1 (a domain name)	Output1-D1, Output2-D1, ..., Output(n)-D1 (multiple outputs related to the input domain)	This component makes API calls to https://metadefender.opswat.com/ and https://www.virustotal.com/ for domain intelligence.
File1 (a file)	File1 (a file)	Output1-F1, Output2-F1, ..., Output(n)-F1 (multiple outputs related to the input file)	This component makes API calls to https://www.virustotal.com/ and https://www.joesandbox.com/ windows for file analysis

Table 4.1 High-Level Design Components and Descriptions

Steps	Detailed Actions Taken
Step 1 - START / DATA INPUT	- The user enters an IP address string for IoC search - The system validates the input string. - The system makes an HTTP POST request on port 80
Step 2 - API CALLS	- The system uses three different APIs: API Website1, API Website2, and API Website3 - These APIs are utilised to gather intelligence data from external resources
Step 3 - REPLY	- The external resources (Website1, Website2, and Website3) respond to the API calls by returning data in JSON format - Each website has a server listening on a specific port (Website1 on port 80, Website2 and Website3 on TCP handshake ports) - The servers handle the requests, perform necessary operations, and send the data back with a status code
Step 4 - OUTPUT	- The system receives the JSON data from the external resources - The output is displayed in separate sections for each resource (<i>Output_{website1}, Output_{website2}, and Output_{website3}</i>) – <i>Each output section displays the responded data, including the “check” status and</i>

Table 4.2 LLD IP Address – Detailed Actions Taken by Step

Steps	Detailed Actions Taken
Step 1 - START / DATA INPUT	- The end-user or client enters a File Hash string for analysis - The system validates the input string - The client makes an HTTP POST request on port 80
Step 2 - API CALLS	- The system uses relevant APIs for file hash analysis, e.g., VirusTotal API, and Hybrid Analysis API - These APIs are utilised to gather intelligence data related to the file hash from external resources
Step 3 - REPLY	- The external resources respond to the API calls by returning data in JSON format related to the file hash - The servers handle the requests, perform necessary operations, and send the data back with a status code
Step 4 - OUTPUT	- The system receives the JSON data from the external resources - The output is displayed, providing information about the file hash, such as detection rates, file details, and potential malware analysis

Table 4.3 LLD HASH – Detailed Actions Taken by Step

Steps	Detailed Actions Taken
Step 1 - START / DATA INPUT	- The end-user or client enters a Domain string for analysis - The system validates the input string - The client makes an HTTP POST request on port 80
Step 2 - API CALLS	- The system uses relevant APIs for domain analysis, e.g., Whois API, and Domain Reputation API - These APIs are utilised to gather intelligence data related to the domain from external resources
Step 3 - REPLY	- The external resources respond to the API calls by returning data in JSON format related to the domain - The servers handle the requests, perform necessary operations, and send the data back with a status code
Step 4 - OUTPUT	- The system receives the JSON data from the external resources - The output is displayed, providing information about the domain, such as registration details, reputation scores, and potential malicious activities associated with the domain

Table 4.4 LLD DOMAIN – Detailed Actions Taken by Step

High Level Design of the Proposed Model

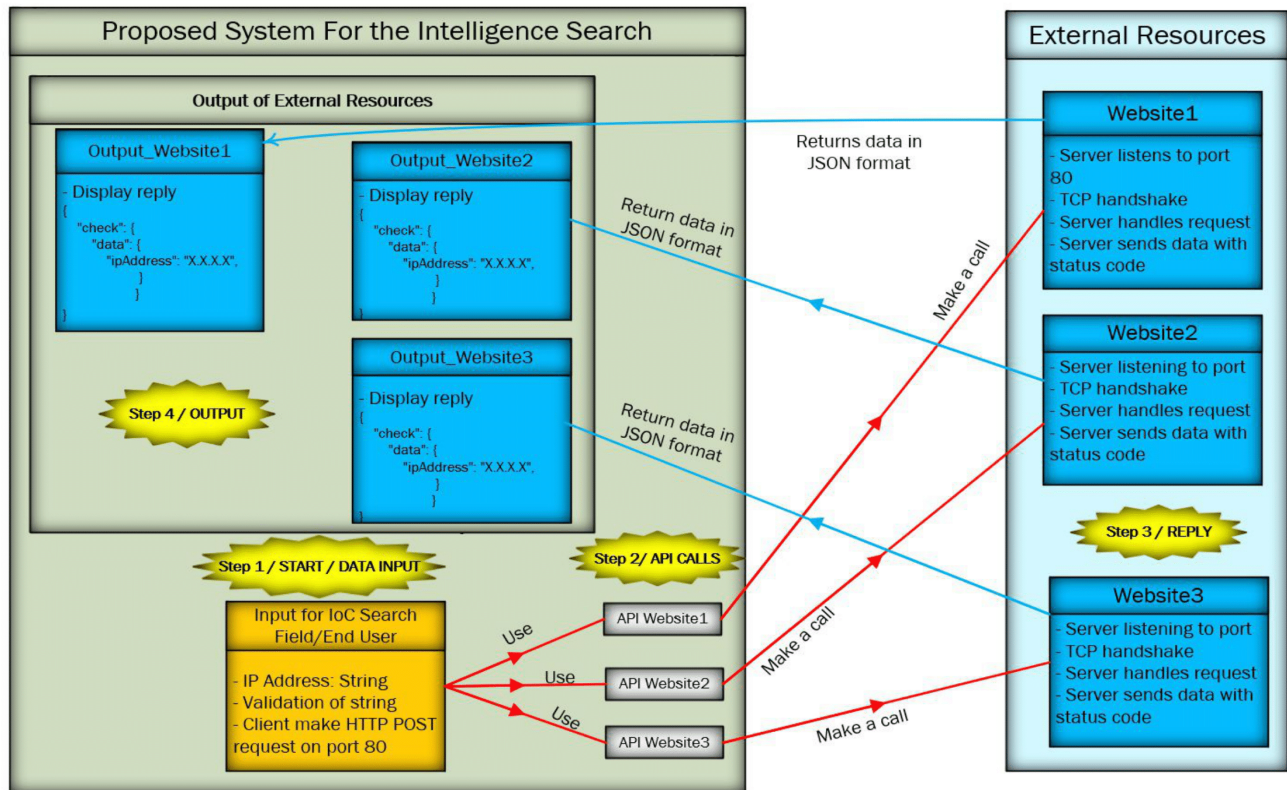


Fig. 4.6 Low-Level Design

4.5.1 List of Utilised Design Software

Several tools contributed to working on projects that involved creating designs and implementations. The tools of interest for our project are as follows:

1. **POSTMAN**: an API client for backend testing used to test, share, develop, and document APIs. Similar tools are Swagger and Parasoftware. We selected POSTMAN because it is simple to use. It effectively helps in reaching API endpoints by rapidly making requests according to the API specification and analysing the various response parameters like the header, status code and the response body itself. It was also chosen because it can simplify making API requests and analysing the responses. Postman's intuitive interface and powerful features, such as environment management, automated testing, and debugging capabilities, made it an ideal choice for interacting with the external APIs integrated into the platform.

Steps	Detailed Actions Taken
Step 1 - START / DATA INPUT	- The end-user or client uploads or selects a file for analysis - The system validates the file upload/selection. The client makes an HTTP POST request on port 80, potentially with the file data
Step 2 - API CALLS	- The system uses relevant APIs for file analysis, e.g., Virus-Total API, and Sandbox Analysis API - These APIs are utilised to gather intelligence data related to the file from external resources
Step 3 - REPLY	- The external resources respond to the API calls by returning data in JSON format related to the file - The servers handle the requests, perform necessary operations, and send the data back with a status code
Step 4 - OUTPUT	- The system receives the JSON data from the external resources - The output is displayed, providing information about the file, such as malware detection results, behavioural analysis, and potential threats associated with the file

Table 4.5 LLD FILE - Detailed Actions Taken by Step

2. SYMFONY (for backend coding): a PHP web application framework and a set of reusable PHP components/libraries (CssSelector, ErrorHandler, Form, HttpClient, etc.). It aims to build up robust products in a corporate environment, accelerate the design and upkeep of web applications and replace recurring coding tasks.
3. ANGULARJS: a JavaScript-based open-source front-end web framework. It aims to simplify the development and testing of applications.

In the development and design of the research project, several key software tools played instrumental roles. Firstly, POSTMAN served as a crucial tool for API testing and validation. Its user-friendly interface and powerful testing features allowed for seamless verification of API endpoints, ensuring data communication reliability throughout the project.

Secondly, SYMFONY was employed as a robust PHP web application framework. Its utilisation streamlined the development process by providing a structured framework for building scalable and maintainable applications. Its extensive libraries and tools facilitated rapid development, reducing code redundancy and enhancing project efficiency.

Lastly, ANGULARJS was chosen for front-end development due to its versatility and ease of integration. As a JavaScript framework, it allowed for the creation of dynamic,

responsive user interfaces. Its modular architecture and two-way data binding simplified complex interactions within the project's web application, contributing to a more interactive and user-friendly experience. These software tools, in tandem, played pivotal roles in shaping the project's architecture and functionality, ultimately driving its success.

4.5.2 Motivation for Tool Selection

These specific design software tools were selected with a well-thought-out rationale aligned with the project's objectives. POSTMAN was chosen because of its reputation as a leading API testing tool, offering a straightforward interface and powerful testing capabilities. Its ability to streamline API validation contributed to data integrity and reliability, important for a research project heavily reliant on data communication. SYMFONY was chosen because of its ability to scale and strong features, which met the requirements of the project and could effectively manage challenging data processing and user interactions. Its large library of components made development tasks easier, reducing time and preserving code quality. In the same manner, ANGULARJS was chosen due to its potential for creating responsive and dynamic user interfaces. The project's goal of developing an interactive and user-friendly web application served as the driving force for the choice. Because of its modular design and two-way data binding, AngularJS was the best option for accomplishing these goals and improving the user experience as a whole. In conclusion, the choice of these tools was driven by their capacity to satisfy particular project needs, improve development, and enhance the research project's overall success.

4.6 How Will SOC Teams Benefit from *SecOps_360*

The *SecOps_360* platform addresses many of the key challenges that Security Operations Centre teams face daily, offering practical solutions that improve both individual team performance and overall SOC effectiveness. By bringing all teams together in one integrated environment, the platform transforms how security operations are conducted.

- **Breaking Down Information Silos**

One of the biggest problems in traditional SOC's is that teams often work in isolation, using different tools and systems that don't communicate with each other. Threat Intelligence analysts might discover a new attack campaign, but this information could

take hours or even days to reach the Analysts who need it most. *SecOps_360* eliminates these delays by providing a shared workspace where critical threat information is immediately available to all relevant teams. When a Security Engineer makes changes to detection rules, the Analysts can see these updates straight away, understanding exactly what new alerts to expect. Similarly, Incident Management teams can access real-time information from Penetration Testers about ongoing security assessments, ensuring they don't mistake legitimate testing for actual security incidents.

- **Improving Response Times and Coordination**

The platform significantly reduces the time it takes to respond to security incidents. Instead of sending emails back and forth or making phone calls between teams, all communication and coordination happens within *SecOps_360*. Incident Management teams can instantly see which Analysts are available, what Security Engineers are working on, and whether any Penetration Testing activities might be affecting the systems in question. This immediate visibility means that when a serious incident occurs, teams can coordinate their response much more quickly. Threat Intelligence can provide context about the attack within minutes rather than hours, Security Engineers can implement containment measures without waiting for detailed briefings, and Managers can track progress in real time rather than waiting for status updates.

- **Reducing Duplicate Work and Confusion**

In many SOC's, teams end up investigating the same issues without realising it, wasting valuable time and resources. *SecOps_360* prevents this by making all ongoing activities visible across teams. If an Analyst is already investigating suspicious network traffic, other team members can see this work in progress and either contribute to the investigation or focus their efforts elsewhere. The platform also ensures that everyone is working with the same information. Threat Intelligence feeds are automatically shared with Analysts, vulnerability reports from Penetration Testers are immediately available to Security Engineers, and incident documentation is accessible to all teams that need it.

- **Enhanced Knowledge Sharing and Learning**

SecOps_360 creates opportunities for teams to learn from each other's work. Analysts can see how Threat Intelligence teams build their threat profiles, whilst Security Engineers can understand how their rule changes affect Analyst workflows. This cross-team visibility helps build a more knowledgeable and capable SOC overall. Penetration

Testers can share their findings more effectively with Security Engineers, ensuring that discovered vulnerabilities are addressed quickly. Incident Management teams can access historical data from all teams to improve their playbooks and response procedures.

- **Better Resource Management and Planning**

Managers gain unprecedented visibility into team workloads and resource allocation. Instead of relying on weekly meetings and reports, they can see in real time which teams are overwhelmed and which have capacity for additional work. This allows for more effective task distribution and better planning for future resource needs. The platform's integrated approach also helps identify skill gaps and training opportunities. If Analysts are struggling with certain types of incidents, Managers can arrange for Threat Intelligence specialists to provide additional context and training.

- **Improved Accuracy and Reduced False Positives**

By bringing together information from all teams, *SecOps_360* helps reduce the number of false positive alerts that plague many SOCs. Threat Intelligence context can help Analysts quickly identify legitimate threats versus benign activities. Penetration Testing schedules are visible to all teams, preventing security tests from being mistaken for real attacks. Security Engineers can see how their rule changes affect alert volumes and can work more closely with Analysts to fine-tune detection systems. This collaborative approach leads to more accurate alerting and less time wasted on irrelevant investigations.

- **Streamlined Reporting and Documentation**

The platform standardises how information is recorded and shared across all teams. This means that incident reports contain consistent, comprehensive information from all involved teams. Managers can generate performance reports that accurately reflect the work of all teams rather than relying on disparate systems and manual data collection. Compliance requirements are also easier to meet when all team activities are documented within a single system, providing clear audit trails and comprehensive records of security activities.

- **Future-Proofing SOC Operations**

As the OSINT Search module demonstrates, *SecOps_360*'s modular design allows

new capabilities to be added as they become available or as team needs evolve. This means that SOC teams can benefit from new tools and techniques without having to learn entirely new systems or lose their existing data and workflows. The platform grows with the organisation, ensuring that the investment in training staff and building processes continues to pay dividends as new modules are developed and integrated. By addressing these fundamental challenges in SOC operations, *SecOps_360* enables teams to work more effectively, respond more quickly to threats, and ultimately provide better security for their organisations. The platform doesn't just automate existing processes—it transforms how SOC teams collaborate and operate together.

4.7 Sketching Graphical Interface

The conceptual design phase of the *SecOps_360* platform involves a critical step: sketching. We can visually explore and express the concepts through this method, which makes it easier to understand the functionality, user interface, and overall experience of the platform. Sketching is an iterative process that begins with brainstorming sessions, consisting of generating a multitude of ideas and concepts. These initial sketches serve as a foundation for further refinement, leading to the creation of more detailed mockups and wireframes. By leveraging sketching techniques, we can effectively translate abstract concepts into tangible visual representations of the OSINT Search Module.

4.7.1 Brainstorming

This approach allows for the free flow of thoughts to build upon initial ideas. We investigate various aspects of the platform's design during the brainstorming phase, such as user interfaces, navigation structures, and data visualisation. We take into account the target audience to make sure the platform meets the goals, which are to create an intuitive and efficient platform that empowers users to analyse various threat indicators, such as IP addresses, file hashes, and domains, and seamlessly integrate with external platforms for further analysis and remediation. We explore different ideas and concepts for the design of the OSINT Search Module.

Tab 4.6 presents a diverse range of brainstorming ideas spanning various aspects of the OSINT Search Module, including user interface design and investigation tabs. These ideas serve as a starting point for further refinement and development during the sketching

and conceptual design phases, ensuring that the platform addresses the diverse needs and requirements of the target audience.

Table 4.6 Brainstorming Ideas

Feature / Functionality	Brainstorming Ideas
User Interface Layout	- The homepage should feature a clean and intuitive layout with tabs prominently displayed for investigating different threat indicators, such as IP addresses, file hashes, and domains. - Each tab should have a distinct icon and label for easy identification, allowing users to quickly navigate to the desired investigation tool. - A dedicated tab for uploading files to external platforms for malicious file analysis should be prominently featured, providing users with a streamlined workflow for submitting suspicious files for further investigation.
Continued on next page	

Table 4.6 (continued)

Feature / Functionality	Brainstorming Ideas
Investigation Tabs	-The "IP Addresses" tab should provide users with tools to investigate the reputation, geolocation, and historical activity of specific IP addresses. Features may include WHOIS lookup, reverse IP lookup, and integration with threat intelligence feeds to identify known malicious IPs. - The "File Hashes" tab should allow users to analyse file hashes (e.g., MD5, SHA-256) to determine their reputation and potential threat level. Integration with threat intelligence databases and malware analysis platforms can provide insights into the file's behaviour and known associations with malicious activity. - The "Domains" tab should enable users to investigate domain names to assess their reputation, ownership information, and historical activity. WHOIS lookup, DNS analysis, and integration with threat feeds can help users identify potentially malicious domains and associated threats. - The "Upload Files" tab should provide users with a simple interface for uploading suspicious files to external platforms for analysis. Integration with leading malware analysis services, sandbox environments, and antivirus engines can automatically scan uploaded files for malware and provide detailed reports on their behaviour and characteristics.
Continued on next page	

Table 4.6 (continued)

Feature / Functionality	Brainstorming Ideas
Integration and Collaboration	- The web app should support integration with external threat intelligence feeds, malware analysis platforms, and cybersecurity tools to enrich investigation results and provide users with comprehensive threat intelligence. - Collaboration features such as user comments, annotations, and sharing capabilities should be incorporated to facilitate communication and knowledge sharing among investigators. - An interactive dashboard or notification system can alert users to emerging threats, security incidents, or updates related to their investigations, ensuring timely awareness of critical information.
User Experience and Accessibility	- The design should prioritise user experience, with a focus on simplicity, efficiency, and ease of use. Intuitive navigation, clear labelling, and contextual help features can guide users through the investigation process and minimise cognitive load. - Accessibility features such as keyboard navigation, screen reader compatibility, and adjustable font sizes should be implemented to ensure that the web app is accessible to users with disabilities. - Regular user testing and feedback sessions should be conducted throughout the design and development process to gather insights, identify pain points, and iterate on the design to meet the needs of users effectively

Table 4.6 Brainstorming Ideas

4.7.2 Wireframing

The layout, structure, and information architecture of the OSINT Search Module are highlighted in wireframes rather than the visual design aspects. Wireframes are low-fidelity, skeleton representations of the user interface. Fig. 4.7 is an example of a wireframe based on our brainstorming session. It shows the application's functionality and layout and acts as a visual reference for the design and development stage. We can make sure that the final design satisfies requirements and enhances the ability to efficiently investigate and mitigate

cybersecurity threats by improving and iterating these wireframes. Fig. 4.7 demonstrates the layout of what the application will look like. It is composed of 4 tabs in the main content column. Each tab has a search box for data input to be submitted with an OK button. The output data from various sources will be displayed in the main column. It is important to notice that the header, footer, banners, and left and right columns have been ignored due to the definition of our scope.

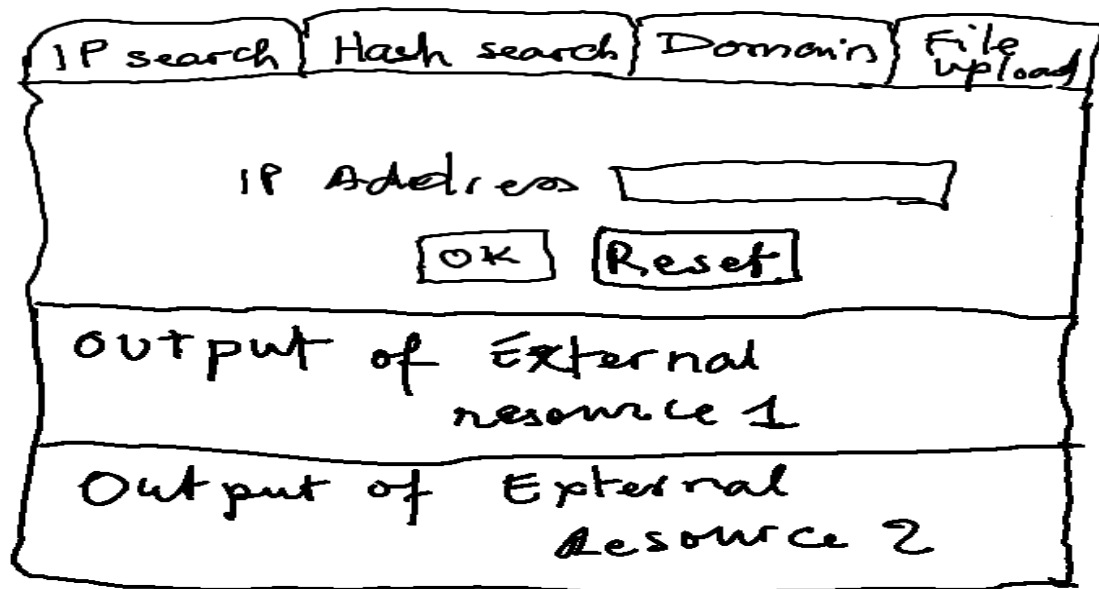


Fig. 4.7 Proposed Wireframe

4.7.3 Mockup

During the mockup creation process, we focus on translating the brainstormed ideas into visually appealing and user-friendly designs. This may involve the use of various design tools and software which provide a range of features and capabilities for creating high-fidelity mockups. Expanding on the thoughts that emerged from our wireframing session, we will investigate various design possibilities and create a mockup to visualise the application's structure, design, and functionality. This stage is essential for turning our ideas into concrete design concepts and setting the stage for the web application's development. In the context of the web-based platform for enhancing malware incident management in SOCs, the mockups include:

- IP Addresses Investigation Tab: Examining the reputation, geography, and past activity of individual IP addresses may be done with the help of the IP Addresses page. Input fields for IP addresses will be present on the main interface, along with buttons that can be used to launch different research tools and searches.
- File Hashes Investigation Tab: Users can evaluate file hashes (such as MD5, and SHA-256) to assess their reputation and possible degree of harm by using the File Hashes tab. The user interface for uploading file hashes, requesting analyses, and seeing the results of analyses will be outlined in the sketch.
- Domains Investigation Tab: Users can look up domain names to evaluate their reputation, ownership details, and past activity by using the Domains tab. Designs will investigate ways to input domain names and provide domain details and analysis outcomes. Visual components, including WHOIS information, historical activity timelines, and domain reputation scores, will be taken into account to give consumers a thorough understanding of the possible threats connected to the name.
- Upload Files to External Platforms Tab: Users will have an easier time sharing suspicious files with outside platforms for investigation with the help of the Upload Files page. The procedure for choosing and uploading files, performing analysis requests, and presenting analysis outcomes from outside platforms will all be described in the High-Level Design (HLD). To automate file analysis and give customers comprehensive reports on the behaviour and features of the file, integration with top malware analysis services, sandbox environments, and antivirus engines will be investigated.

Fig 4.8 represents a sketched mockup of the proposed platform. This component of the user interface is dedicated to the search and analysis of various indicators of compromise (IoCs), including IP addresses. The functionalities provided for URLs/domains and file hashes operate similarly to those available in the IP tab. The sketch will serve as a visual representation of our design concepts and will guide the development of the user interface for the threat investigation. By exploring different layouts, features, and functionalities, we aim to create a user-friendly and efficient platform that empowers security professionals to investigate and mitigate cybersecurity threats effectively.

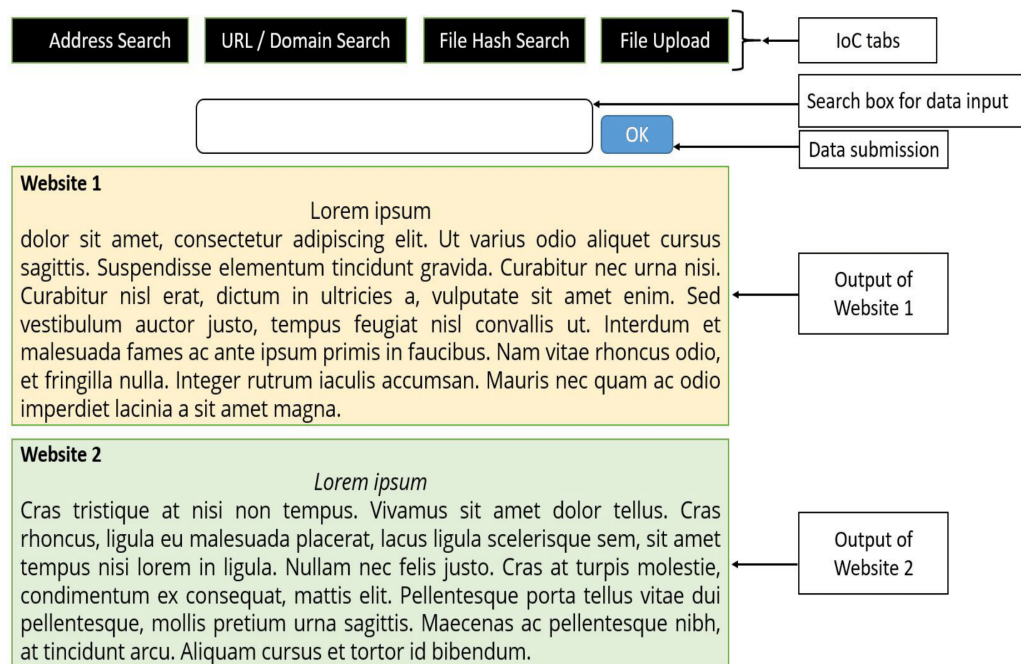


Fig. 4.8 Sketched Mockup

4.8 Chapter Summary

This chapter has delivered a thorough conceptual design framework for the *SecOps_360* platform, creating the essential groundwork required for developing a unified web-based solution that tackles the intricate operational demands of contemporary SOC. By systematically examining requirements, target users, architectural planning, and interface development, the chapter has illustrated how the platform can successfully merge distinct SOC functions into a unified single pane of glass.

The requirements examination identified the pressing necessity for both functional features that directly assist SOC operations and non-functional characteristics that guarantee system dependability, security, and performance. Functional requirements covered the complete range of SOC activities, spanning real-time threat surveillance and incident examination to automated response coordination and performance documentation. Concurrently, non-functional requirements outlined vital criteria for expandability, security, usability, and integration capabilities that will influence the platform's sustained success in enterprise settings. The precisely defined scope offers practical limitations whilst preserving adequate flexibility for future enhancement based on operational input and developing threat environments.

The target audience examination emphasised the varied user community that *SecOps_360*

must accommodate, extending from operational SOC staff requiring intuitive working tools to academic researchers needing comprehensive data analysis features. This diverse audience demands a platform design that harmonises operational effectiveness with analytical complexity, ensuring each user group can extract maximum benefit from their particular applications. Identifying SOC personnel as primary users, complemented by IT security professionals and academic researchers, establishes clear development priorities and interface design choices.

The comprehensive system design constitutes this chapter's primary technical contribution, converting abstract requirements into practical architectural specifications. Role-based module design guarantees each SOC team member receives customised functionality corresponding to their particular duties, from threat intelligence collection to incident response management. UML diagrams offer clear structural frameworks demonstrating the platform's modular design, whilst process flow diagrams show seamless integration with established SOC procedures.

Tool selection analysis exhibits a practical approach to technology decisions, weighing functionality, cost-effectiveness, and industry standards. The rationale for particular design software choices reflects careful evaluation of both immediate development requirements and long-term maintenance needs.

SOC benefits examination articulates the value proposition justifying platform development investment, addressing operational efficiency improvements, enhanced threat detection capabilities, and improved team collaboration.

The interface design process demonstrates a user-focused methodology progressing from initial brainstorming through organised wireframing to comprehensive mockups. This structured approach ensures the final interface design corresponds with user requirements and operational workflows whilst maintaining consistency with established usability standards. Significantly, this chapter recognises current implementation reality whilst preserving a clear future development vision. The transparent acknowledgement that only the OSINT Search Module represents completely operational functionality, whilst other elements remain in the demonstration phase, provides practical expectations. This honesty establishes credibility whilst demonstrating the platform's comprehensive, ultimate capabilities.

The modular architecture throughout this chapter offers strategic advantages for SOC implementations. Organisations can commence with proven elements like the OSINT Search Module whilst progressively expanding functionality based on operational priorities and budget considerations. This staged approach reduces implementation risks whilst delivering immediate SOC operational value. The integration-focused philosophy recognises existing SOC tool investments whilst providing pathways towards unified operations. Rather than

substituting established systems, *SecOps_360* functions as an orchestration layer maximising current investment value whilst addressing coordination challenges. In summary, this chapter has created a comprehensive conceptual framework balancing ambitious vision with practical implementation considerations, positioning *SecOps_360* as a viable solution for modern SOC challenges.

Chapter 5

Implementation of *SecOps_360*

5.1 Introduction

In every research project, the implementation phase represents the critical juncture where ideas, theories, and design blueprints transform into concrete realities. This is when theoretical and conceptual work takes on a practical form, as the result of meticulous planning, design, and preparation. This chapter presents a detailed examination of how the proposed framework for enhancing malware incident management in SOC's has been realised as a functional platform.

Whilst the framework itself provides the theoretical foundation, architectural specifications, and conceptual design for integrated malware incident management, *SecOps_360* represents its tangible realisation as a working platform. The framework encompasses the high-level design, low-level design, UML diagrams, and process flow models that define the structured approach to incident management, whereas *SecOps_360* is the software platform that brings these designs to life.

It is important to clarify the scope of this implementation. This research presents a fully functional implementation of one core module of *SecOps_360*: the ****OSINT Search module****. This module has been developed, tested, and validated to demonstrate the practical viability of the framework and its capacity to address the challenges identified in SOC operations. The OSINT Search module integrates multiple threat intelligence sources into a unified interface, streamlining the investigation process and reducing both system resource overhead and search times. Additionally, other modules within the *SecOps_360* framework have been implemented to demonstrate the scalability and extensibility of the platform architecture. These supplementary modules showcase how the framework can accommodate various SOC functions and workflows, establishing a foundation for future expansion and development.

The implementation phase is crucial in transforming architectural standards and conceptual designs into a reliable and operational system through careful coding and thorough testing. This process demands a methodical approach, applying contemporary software development practices, industry standards, and modern tools and technologies. The primary objective of this implementation is to bridge the gap between theoretical concepts and real-world applications, demonstrating that the proposed framework can effectively address the documented challenges faced by SOC teams in malware incident management.

This chapter explores the practical aspects of turning the framework into a functional platform. We examine the coding methodologies, structures, and technologies employed to transform abstract concepts into working programmes. The discussion covers the intricacies of both frontend and backend development and user interface design. Here, creativity and practical implementation converge to realise the project's logic and functionalities,

demonstrating how the *SecOps_360* platform addresses real operational needs within SOC environments.

5.2 Implementation Requirements

It's essential to carefully prepare the ground for implementation before diving into the complex realm of code development. This subsection concentrates on the fundamental preparation needed to guarantee a smooth transition from design to code. It includes choosing the programming languages, frameworks, and tools that will be used, as well as a thorough analysis of the development environment. To start coding, we have used the following:

1. 1 computer with the following characteristics
Processor: Intel(R) Core(TM) i7-8565U, CPU @ 1.80GHz 1.99 GHz
RAM: 16.0 GB (15.9 GB usable) System Type: 64-bit operating system, x64-based processor Windows specifications: Edition Windows 11 Home
Version 21H2
OS build 22000.856
2. Visual Studio Code: Download (<https://code.visualstudio.com/download>) and install Visual Studio Code, a lightweight source code editor with built-in support for JavaScript, TypeScript, and Node.js.
3. WAMPSEVER 3.2.3: Download (<https://www.wampserver.com/en/download-wampserver-64bits/>) and install the application WAMPSEVER 3.2.3, a Windows web development environment that helps to create web applications with PHP, a MySQL database and Apache2
4. AngularJS 12.0.11: Download (<https://angularjs.org/>) and install AngularJS, a free and open-source JavaScript-based web framework used to develop a single-page application.
5. Symfony 4.4: Download (<https://symfony.com/download>) and install Symfony 4.4, a free and open-source framework of PHP web applications with a collection of PHP component libraries that can be reused.
6. Postman: Download (<https://www.postman.com/downloads/>) and install Postman, an API platform to design, build, test, and iterate APIs.
7. Gitlab: It brings teams together in one application. In our case, GitLab will be used for tracking the versions of our code.

Implementing this model has been quite challenging in terms of the choice of tools as well as coding. The cycle of learning, applying the skills, consulting other professionals, and improving is time-consuming, but the best way to get some hands-on experience. The tools that have helped us implement the model are:

5.3 System Development

The phase of coding comes once the design has been completed and the main decisions about the system are made. This stage translates the design of the system into a given programming language. Our system uses Symfony for back-end coding and AngularJS for front-end.

5.3.1 Backend Development

We use Symfony 4.4, a stable and adaptable PHP framework, and some PHP component libraries in the back-end source code of our web-based platform to improve malware incident management in SOC, to speed up the deployment of various functionalities. The back-end code is responsible for handling requests from the client interface, processing input data, and performing necessary operations to analyse and manage malware incidents effectively. By leveraging Symfony 4.4 and other PHP component libraries, we ensure the robustness, scalability, and security of the back-end source code, enabling the effective management of malware incidents within SOC environments. The implementation phase focuses on translating the design and requirements of the web-based platform into functional and reliable software components, ultimately enhancing the capabilities of SOC personnel in mitigating cybersecurity threats. We declare and initialise global keys, including private and public keys, in the Declaration and Initialisation of Global Keys section of the back-end source code to provide safe communication and data encryption between the client and server. These keys, which are produced by cryptographic methods, are necessary for both encrypting and decrypting private data that is transferred throughout the malware incident management process. To manage incoming requests from the client interface, we define routes for POST methods in the back-end code in the Route Definition for POST Methods. The various pathways align with distinct features of the web application, including sending malware samples for examination, obtaining analysis outcomes, and changing the status of incidents. Every route has a set of controller actions that carry out the logic required to process the request and produce the relevant response. To gather and encode data, the back-end code

collects input data, such as specifics on IoC details and malware samples, from the client interface. The way this data is encoded depends on the kind of request; this could entail transforming user input into formats that are appropriate for it, including character strings or JSON objects. Encoding guarantees that information is sent between the web application's client and server safely and precisely. The backend portion of the code contains algorithms to evaluate the outcomes of malware analysis for different indicators like IP addresses, URLs, file hashes, and uploaded files. This entails querying external threat intelligence databases, conducting behavioural examinations, and comparing against known malware fingerprints to detect potential threats. The system then processes these results and displays them through the client interface for the Security Operations Centre personnel to take appropriate further actions. In other words, the backend logic checks malware analysis outputs against various data points (IPs, URLs, hashes, files), leveraging external intelligence sources and comparing them to known malware signatures. It runs behavioural analysis on the files as well. After processing this information, it surfaces potential threat indicators to the SOC team through the frontend client interface for them to investigate and respond to.

Fig. A.1 represents the first block of backend code. In this block, the namespace `App` is defined, which specifies the location of the controller class within the application's directory structure. Additionally, several classes are imported from different namespaces described in Tab A.1

```
namespace App\Controller;
use GuzzleHttp\Client as GuzzleHttp;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
```

Fig. 5.1 Backend Source Code Block1

Fig A.2 represents the second block of the backend code. This block defines a new class named *SearchController* that extends the *AbstractController* class. The `@Route("/search")` annotation defines the base URL path for all actions within this controller.

Fig A.3 shows the third block of the backend code. This block declares and initialises private class properties for various API keys and URLs used throughout the controller. These keys and URLs are related to different services or APIs that the controller interacts with,

NameSpace	Description
GuzzleHttp	Imported from the GuzzleHttp library, which is used for making HTTP requests
Symfony\Component\HttpFoundation\Request	Used to handle incoming HTTP requests
Symfony\Component\Routing\Annotation\Route	Used for defining the routes (URLs) for the controller actions.
Symfony\Bundle\FrameworkBundle\Controller\AbstractController	base class for Symfony controllers

Table 5.1 Imported Classes from Namespaces and Descriptions

```
/**
 * @Route("/search")
 */
class SearchController extends AbstractController
{
```

Fig. 5.2 Backend Source Code Block2

such as OPSWAT, VirusTotal, Hybrid Analysis, and Abuseipdb. Details and the rest of the backend code can be found in Appendix A

5.3.2 Frontend Development

We employ the robust JavaScript framework AngularJS in the front-end source code to create a dynamic and adaptable user experience. Presenting the user interface, gathering input from the user, and validating data before sending it to the back end for additional processing are all handled by the front-end code. We provide a user-friendly and error-tolerant interface for handling malware incidents within SOC environments by focusing on defining and initialising variables in a constructor and providing strong data validations in the front-end source code design. Before taking further action to assess and mitigate cybersecurity threats, SOC staff may input and validate data quickly and effectively thanks to the AngularJS application, which enables seamless user-platform interaction. Below is an overview of the front-end source code, focusing on declaring and initialising variables in a constructor and designing

```

/**
 * Global declaration of KEYS and Initialization.
 */
private $keyOpswat = 'ee0a10b16392ff3fe0ee872a3676cab4';
private $keyVirus = '58a6fe675b9b849e7086fe4d662b5a6c5df5ae667c322b839c3c7a1f1dd0ea91';
private $keyHybrid = 'fmgvzcjuc8c4674cr8jm6b2x31aea1949dvxp3wkad66e8e0lkt4pbnf629c13d8';
private $keyAbuse = 'd3304c0b2f0febcbade4750868a99c9567988ba940031856257929e64588908065698

/**
 * Global Declaration of URLs and Initialization.
 */
private $opswatURL = 'https://api.metadefender.com/v4/';
private $abuseipURL = 'https://api.abuseipdb.com/api/v2/';
private $virusTotalURL = 'https://www.virustotal.com/api/v3/';
private $hybridURL = 'https://www.hybrid-analysis.com/api/v2/quick-scan/file';

```

Fig. 5.3 Backend Source Code Block3

the form with different data validations. The complete details of the code are available in *Appendix B*:

Declaration and Initialisation of Variables in a Constructor: To handle the state of the user interface, we declare and initialise the variables needed in the constructor method of the AngularJS controller. These variables contain objects for storing validation rules and error messages, flags for managing form submission and error handling, and data fields for collecting user input. We verify the availability of these variables for use throughout the controller's lifecycle and their easy accessibility and manipulation as required by initialising them in the constructor. Fig. 5.4 shows a block of code that declares the *SearchComponent* class and initialises various properties that will be used throughout the component. It includes properties for form handling, input values (IP address, URL, hash, file), API response data, loading state, and other flags.

Fig 5.5 is a block of code to initialise the form. The *newSearch* method initialises the form using the *FormBuilder*. It defines the form controls (*ipAddress*, *ip*, *network*, *hash*, *URL*), and *file*) and their initial values. It also applies validators for the *ipAddress* and *URL* controls using regular expressions.

```

export class SearchComponent implements OnInit {
  // Declaration of necessary variables and use in the component
  public loading = false;
  form: FormGroup
  ipAddress = null
  url = null
  hash = null
  result: any
  check: any
  block: any
  virusIp: any
  virusUrl: any
  antiVirus: any
  search: any
  opswat: any
  opswatURL: any
  virusHash: any
  antiVirusHash: any
  hybridFILE: any
  file: File = null
  fullReport: boolean = false

```

Fig. 5.4 Component Class Declaration

Form Design with Data Validations: HTML templates and AngularJS directives are used in the front-end source code to create the form that will be used to gather user data about malware events. We use a variety of input fields to collect data, including URLs, file hashes, file uploads, and incident information. In addition, we employ the built-in validation directives in AngularJS to implement data validations, making sure that user input satisfies predetermined standards. The 'ng-pattern' directive to enforce specific patterns for input data (e/g, valid URLs, file hashes) and custom validation functions to perform more complex validations (e.g., checking file size limits, verifying file formats). Error messages are displayed dynamically based on validation results, providing users with real-time feedback on the validity of their input.

Fig 5.6 is a block of code for form validation that contains four methods: *onSearchIp*, *onSearchUrl*, *onSearchHash*, and *onSearchFile*. These methods are responsible for handling the user input for IP address, URL, hash, and file, respectively. These methods are likely used in conjunction with form inputs or buttons to handle user input for different types of searches (IP address, URL, hash, or file). When the user interacts with the corresponding input field or button, the respective method is called to update the form controls and the *search* property accordingly.

```

newSearch() {
  this.form = this.formBuild.group({
    ipAddress: [null, [Validators.compose([
      Validators.pattern('(?:?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.){3}(?:25[0-5]|2
    )]],
    ip: [null],
    network: [null],
    hash: [null],
    url: [null, [Validators.compose([
      Validators.pattern('(https?:/)?(\\da-z.-+)?\\.([a-z.]{2,6})[/\\w .-]*/?')
    )]],
    file: [null]
  })
}

```

Fig. 5.5 Form Initialisation Code

Managing Form Submission and Data Processing: The front-end code starts an event handler function to verify input data and start the submission process when a user submits a form. Data validation verifies that input data satisfies predetermined criteria and that all required fields are accurately filled out. If validation is successful, user input is assembled into a data payload by the front-end code and sent to the back-end server for additional processing. When there are validation issues, users are directed to correct their input by the appropriate colour code (green for valid data and red otherwise) that is presented around the associated input forms. Only once all validation errors have been fixed can the form submission procedure be started. Fig. 5.7 illustrates a form submission code with the *onSubmit* method responsible for handling the form submission for IP address, URL, or hash. It first calls the *onNull* method to clear any previous results. It then checks if at least one of the *ipAddress*, *URL*, or *hash* properties is not null. If any of these properties is not null, it sets the *loading* flag to true and updates the *IP* and *network* properties of the form value based on the *ipAddress*. It then calls the *getCheck* method from the *DataService* with the form value. Upon a successful response, it updates the result property with the response data and checks if the result is not null.

5.4 Platform Interface and Components Prototype

The achievement of the research project's goals and vision is symbolised by the "Final Product" subsection, which marks the end of the implementation phase. This is the point at which code and careful planning come together to provide a real, workable solution.

```

onSearchIp() {
  if(this.ipAddress !== null) {
    this.form.patchValue({"url": null})
    this.form.patchValue({"hash": null})
    this.form.patchValue({"file": null})
    this.search = this.ipAddress
  }
}

onSearchUrl() {
  if(this.url !== null) {
    this.form.patchValue({"ipAddress": null})
    this.form.patchValue({"hash": null})
    this.form.patchValue({"file": null})
    this.search = this.url
  }
}

onSearchHash() {
  if(this.hash !== null) {
    this.form.patchValue({"url": null})
    this.form.patchValue({"ipAddress": null})
    this.form.patchValue({"file": null})
    this.search = this.hash
  }
}

onSearchFile(files: FileList) {
  if(files.length > 0) {
    const file = files.item(0)
    this.form.patchValue({"ipAddress": null})
    this.form.patchValue({"hash": null})
    this.form.patchValue({"url": null})
    this.file = file
    this.search = file.name
  }
}

```

Fig. 5.6 Form Validation Code

This chapter captures the spirit of the project's path, showing how creativity and execution interact to turn abstract ideas into tangible realities. This is where the finished research study is presented, displaying the result of labour, ingenuity, and knowledge. It includes the underlying infrastructure and code that power the user-facing components, such as a web platform. This chapter emphasises the importance of paying close attention to every little detail, from backend functionality to user interface design, to ensure that the finished product not only meets the project's goals.

The "Final Product" part is a celebration of accomplishment rather than just a presentation. It stands as evidence of the study project's devotion to work and to delivering a solution that benefits the target market. The research effort makes a significant contribution as it enters the domain of practical application.

Fig 5.8 indicates the graphical user interface (GUI) of the artefact.

The platform provides several options for users to search for or analyse potential malware threats.

```

onSubmit() {
  this.onNull()
  if (this.ipAddress || this.url || this.hash) {
    this.loading = true
    this.form.value.ip = this.ipAddress ? this.ipAddress : null
    this.form.value.network = this.ipAddress ? this.ipAddress + '/24' : null
    this.data.getCheck(this.form.value).subscribe(res => {
      this.loading = false
      this.result = res
      if(this.result !== null) {
        this.notify('Congratulations', 'Results corresponding to '+this.search, 'success')
        this.opswat = res?.opswatHash ? res?.opswatHash : null
        this.check = res?.check ? res?.check.data : null
        this.block = res?.block ? res?.block.data : null
        this.virusIp = res?.virusIp ? res?.virusIp.data : null
        this.virusUrl = res?.virusURL ? res?.virusURL.data : null
        this.virusHash = res?.virusHASH ? res?.virusHASH.data[0] : null
        this.opswatURL = res?.opswatURL ? res?.opswatURL : null
        this.antivirusHash = this.hash ?
          this.virusHash?.attributes?.last_analysis_results : null
        if(res.virusIp || res.virusURL) {
          this.antivirus = this.ipAddress ?
            this.virusIp?.attributes?.last_analysis_results :
            this.virusUrl?.attributes?.last_analysis_results
        } else if (res.opswatHash) {
          this.antivirus = this.opswat?.scan_results?.scan_details
        }
        this.loading = false
      } else {
        this.notify('Results', 'No results match your search '+this.search, 'warning')
        this.loading = false
      }
    }, error => {
      this.loading = false
      this.notify('Error', 'Error caught', 'error')
    })
  } else {
    this.notify('Warning', 'Please fill in at least one field', 'warning')
    this.loading = false
  }
}
}

```

Fig. 5.7 Form Submission Code

At the top, there is a header section with the text "MALWARE ANALYSIS" and four tabs labelled "IP ADDRESS", "URL / DOMAIN", "FILE HASH", and "UPLOAD FILE". These buttons allow users to input different types of data for analysis, such as IP addresses, URLs or domain names, and file hashes, or upload files directly.

In the centre of the interface, there is an input field where a user has to enter the IP address. The green checkmark icon next to the input field suggests that this IP address has been validated or accepted by the system. Below the input field, there is a button labelled "IP ADDRESS" which initiates the analysis or search for the entered IP address.

Below the IP address input section, there is another section with the header "MALWARE ANALYSIS" and the text "Please search here." This section displays the analysis results after the queries.

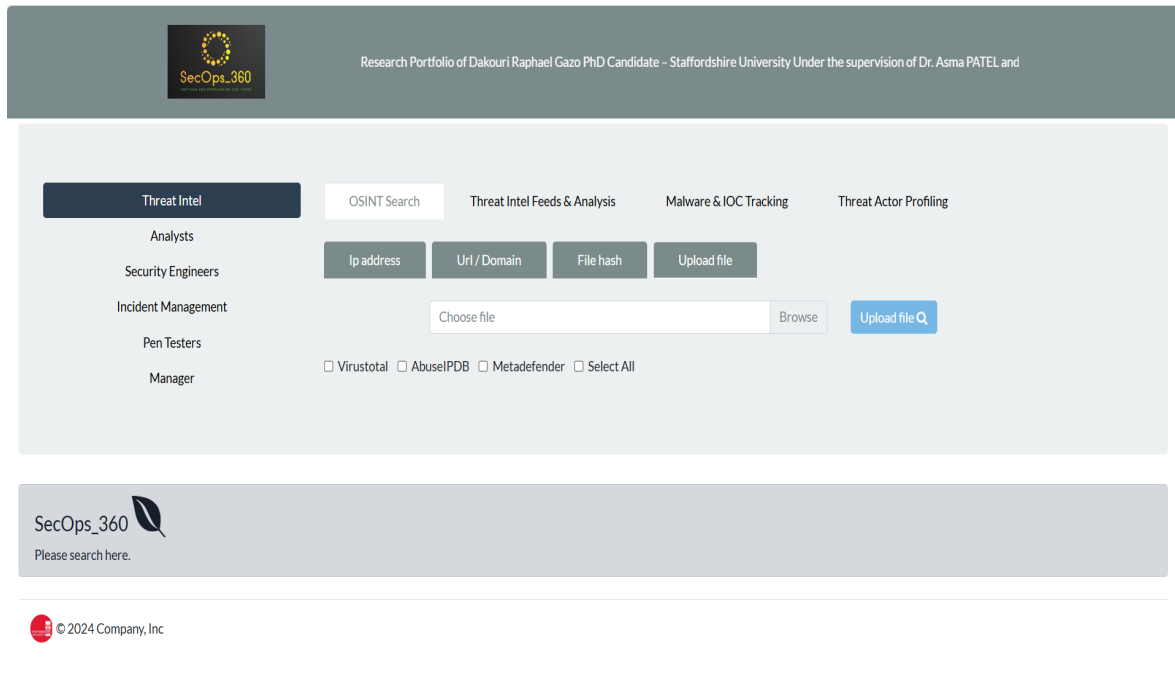


Fig. 5.8 Visual Representation of the Final Product

5.5 Chapter Summary

This chapter has documented the comprehensive implementation process of the *SecOps_360* platform, demonstrating the practical transformation of conceptual designs into a functional web-based solution for SOC malware incident management. Through systematic development of both backend infrastructure and frontend interfaces, the implementation phase has successfully delivered a working platform that addresses the core requirements identified in earlier chapters. The implementation journey began with establishing the necessary conditions for development, including setting up development environments and ensuring proper security protocols were in place. This foundational work proved essential for supporting the complex requirements of an SOC platform, particularly regarding data handling and system integration capabilities. The careful preparation of these implementation conditions enabled smooth progression through subsequent development phases whilst maintaining security standards appropriate for cybersecurity applications. The backend development phase focused on creating robust server-side infrastructure capable of handling the diverse functional requirements of SOC operations. This included implementing user management systems with role-based access controls, developing API endpoints for external system integration, and creating a modular architecture that supports different SOC team functions.

Frontend development complemented the backend infrastructure by creating intuitive user interfaces tailored to different SOC roles and responsibilities. The implementation process emphasised user experience design principles whilst ensuring that complex SOC functionalities remained accessible to operators working under time-sensitive conditions. The frontend development successfully translated the wireframes and mockups from the conceptual design phase into functional interfaces that support efficient workflow management and clear information presentation. The final product represents a significant achievement in demonstrating how academic research can be translated into practical cybersecurity tools. The *SecOps_360* platform, with its operational OSINT Search module and demonstration modules for other SOC functions, provides a concrete foundation for future development whilst offering immediate value to SOC operations. However, the implementation phase has also highlighted several important considerations for future development. The current focus on the OSINT Search module as the primary operational component reflects both the practical constraints of development resources and the importance of validating core platform concepts before expanding functionality. The implementation process has demonstrated that creating effective SOC platforms requires balancing technical sophistication with operational simplicity. The successful delivery of a working system that integrates with existing SOC workflows whilst providing enhanced capabilities shows that academic research can produce practical solutions for real-world cybersecurity challenges. The modular approach adopted during implementation provides a sustainable development model that other researchers and developers can adapt for similar projects. Looking forward, this implementation phase establishes a solid foundation for continued platform evolution. In conclusion, the implementation chapter has successfully demonstrated the transformation of research insights and conceptual designs into a functional SOC platform. The *SecOps_360* implementation represents both an achievement in its own right and a stepping stone towards more comprehensive SOC solutions. The systematic approach to implementation, combined with a realistic assessment of current capabilities and future potential, positions the platform for continued development and broader adoption within the cybersecurity community.

Chapter 6

Testing and Evaluation

6.1 Introduction

Methodologically assessing the platform requires a well-organised and comprehensive testing approach. By defining the goals, parameters, and standards for the testing stages, this section provides a systematic approach to identifying and resolving potential challenges. The approach includes multiple testing aspects, such as unit tests, integration tests, and system tests, tailored to scrutinise the platform's components individually and collectively. Verifying that the platform accurately and effectively carries out its intended functions is the goal of functionality testing. The compatibility testing evaluates the platform's interoperability across multiple operating systems, browsers, and network environments.

This chapter presents the testing and evaluation of *SecOps_360*. Section 6.2 describes functional and compatibility testing. Section 6.3 presents the scenario-based performance evaluation. Section 6.4 presents the overall evaluation findings. Section 6.5 provides a comparative analysis with related work. Section 6.6 proposes a programme of future experimental work, using appropriate empirical and statistical methods, to substantially strengthen the validation of the framework. Section 6.7 concludes the chapter.

6.2 Functional and Compatibility Testing

Functional testing verified that each module of *SecOps_360* operated in accordance with its specification, including IoC input validation (IP address format, domain format, hash format), API query routing, result aggregation, and result display. All 23 functional test cases passed. Compatibility testing confirmed correct operation across Google Chrome (v120), Mozilla Firefox (v121), and Microsoft Edge (v120) on Windows 11. W3C markup validation confirmed standards-compliant HTML output.

This section delves into the rigorous testing and evaluation phases of *SecOps_360*, highlighting its functionality, performance, and efficacy in real-world scenarios. By conducting a comprehensive comparative analysis between traditional malware investigation methods and the *SecOps_360*-assisted approach, we aim to quantify the potential benefits and demonstrate the platform's ability to improve incident response times, enhance collaboration, and enhance the overall security posture of organisations. The chapter starts with a detailed testing plan and runs through the methodologies and strategies employed to ensure the platform's strength and reliability. This includes functional testing and performance testing, among others. Subsequently, we present two distinct scenarios to evaluate the platform's functionality and effectiveness. The first scenario describes a traditional malware investigation process without the assistance of *SecOps_360*, highlighting the limits inherent in such an approach.

The second scenario displays the integration of our platform into the malware investigation process, demonstrating its potential to streamline workflows, facilitate information sharing, and enhance decision-making capabilities. Through a rigorous comparative analysis, we objectively evaluate the performance metrics and outcomes of both scenarios, including response times, resource utilisation, and the overall effectiveness of the incident management process. This analysis not only highlights *SecOps_360*'s strengths and capabilities but also identifies areas for further improvement and refinement. Fig. 6.1 shows an example of an incident management process Mutemwa, M. et al. (2018). It usually begins with the detection of an incident that triggers the incident registration phase. During this phase, incident details are recorded in an IT Service Management (ITSM) system, resulting in the creation of an incident ticket. Categorisation is the next step, where the incident is assigned a category, type, and item (CTI) classification. This categorisation facilitates the correct ticket assignment to the appropriate resolution group. The assignment phase involves assigning the incident to the Incident Response (IR) team or individuals responsible for resolving the issue. The assignment decision is based on the categorisation of the incident by priorities (P1, P2, P3...), ensuring that the most qualified and knowledgeable personnel manage the ticket. Diagnosis is the process through which the incident resolution team begins investigating the issue at hand. This phase may involve reviewing system logs, analysing user errors, examining network configurations, and gathering any relevant information that can aid in identifying the root cause of the incident. Once the root cause has been identified and addressed, the incident can be marked as resolved. The resolution phase also includes communicating with affected users or stakeholders and informing them that the incident has been successfully resolved. The closure phase involves conducting a Post-Implementation Review (PIR) with relevant stakeholders. During this review, the root cause of the incident is thoroughly analysed, and lessons learnt, action items, and residual risks are identified. The closure process is essential for continuous improvement, as it allows for the documentation of valuable insights and the implementation of preventive measures to mitigate the recurrence of similar incidents in the future.

Table 6.1 represents the classification of testing inputs and expected outputs according to the submitted IoC. This table serves as a reference or guide for testing or analysing different types of IoCs and verifying that the expected outputs are being generated correctly. It helps ensure that the analysis system or tool is providing the necessary information for each type of input, which can be useful for security analysts, incident responders, or developers working on threat detection and analysis solutions.

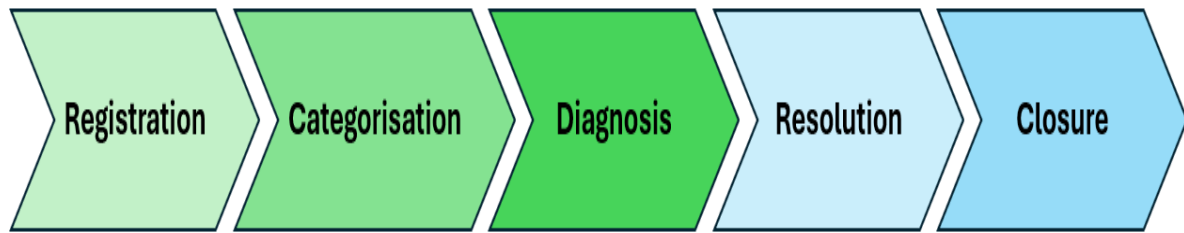


Fig. 6.1 SOC Internal Incident Management Process Steps

6.2.1 Functionality Testing

This step ensures whether the functionalities of our system are working properly or not. This is about testing the operational behaviour and features of a system to make sure they match up with its requirements.

Fig. 6.2 shows the validation test of the "IP Address" tab represented by two pictures, A and B. In the first section (A), an invalid IP address format, "86.32.65.sr" has been entered in the text field. The system has correctly identified and displayed an error message, "Invalid IP Address format", below the text field, indicating that the input validation for IP addresses is working as expected. In the second section (B), a valid IP address, "86.32.65.14" has been entered in the text field, and no error message is displayed, suggesting that the system accepts this as a valid IP address format. This figure demonstrates the functionality testing of the IP address input field of *SecOps_360*. It illustrates the platform's ability to validate user input for IP addresses, ensuring that only valid IP address formats are accepted while rejecting invalid formats and providing appropriate error messages to the user. The same type of validation is done for the "URL/Domain" and "File Hash" tabs. Similarly, Fig. 6.4 is the result of the World Wide Web Consortium (W3C) standard check on <https://validator.w3.org/>. This means the CSS and HTML codes of our system are compliant with the standards set by W3C.

	Testing inputs	Expected output
IP address	String e.g.: 86.32.65.48	-scan score e.g.: 93 % -Domain name e.g.: itc.ir -Country e.g.: Iran (Republic of) -City e.g.: Tabriz, Azarbayjan-e Sharqi -Abuse report: e.g.: reported 415 times from 147 distinct sources -detection e.g.: malicious -Whois Lookups e.g.: OrgAbuseHandle:ABUSE3850-ARIN
Domain/URL	String e.g.: bestdove[.]in[.]ua	-Scan score: 6 -Scan result: No threat detected -Category of domain or URL, e.g.: malware repository
File Hash	String e.g.: 9b07094151ee3ee8b82a62a1f1052ae6	-Scan Score: e.g.:44/63 -Threat name e.g.: Unknown Threat <i>Detection : e.g. : Win – Trojan/RMC_v10</i> -Details e.g.: File type: Win32 EXE
File	Object e.g: brbbot.exe	-Threat name -score -Dynamic analysis report (signatures, header, imports, processes, network)

Table 6.1 Classification of Testing Inputs with Expected Outputs per IoC

6.2.2 Compatibility Testing

The compatibility test checks whether an application can run on different operating systems, browsers, applications, and hardware. Our test will only focus on the functionality of the browser compatibility.

Fig. 6.5 shows some figures of the web application being tested with the most common browsers, such as Google Chrome, Firefox, Opera, Microsoft Edge, and Safari.

Test	Pass / Fail
IP address field validation	Pass
URL/Domain field validation	Pass
File Hash field validation	Pass
File upload field validation	Pass
A confirmation message should display	Pass
The functionality of the buttons available	Pass
Switch to IP Address tab	Pass
Switch to the URL/Domain tab	Pass
Switch to the File Hash tab	Pass
Switch to the Upload File tab	Pass
The browse button gets to the file	Pass
"Upload File" tab - File path is visible after selection	Fail
"IP address" tab - Submit button is functional	Pass
"URL/Domain" tab - Submit button is functional	Pass
"File Hash" tab - Submit button is functional	Pass
"Upload File" tab - Submit button is functional	Pass
W3C standard compliance	Pass
Virustotal checkbox	Pass
AbuseIPDB checkbox	Pass
Metadefender checkbox	Pass
Select All checkbox	Pass

Table 6.2 Functionality Test Results

Browser	Pass / Fail
Google Chrome - Version 104.0.5112.82 (Official Build) (64-bit)	Pass
Firefox - Version 103.0.2 (64-bit)	Pass
Microsoft Edge - Version 104.0.1293.63 (Official build) (64-bit)	Pass
Opera - Version:89.0.4447.91	Pass
Safari	Pass

Table 6.3 Browser Compatibility Test Results

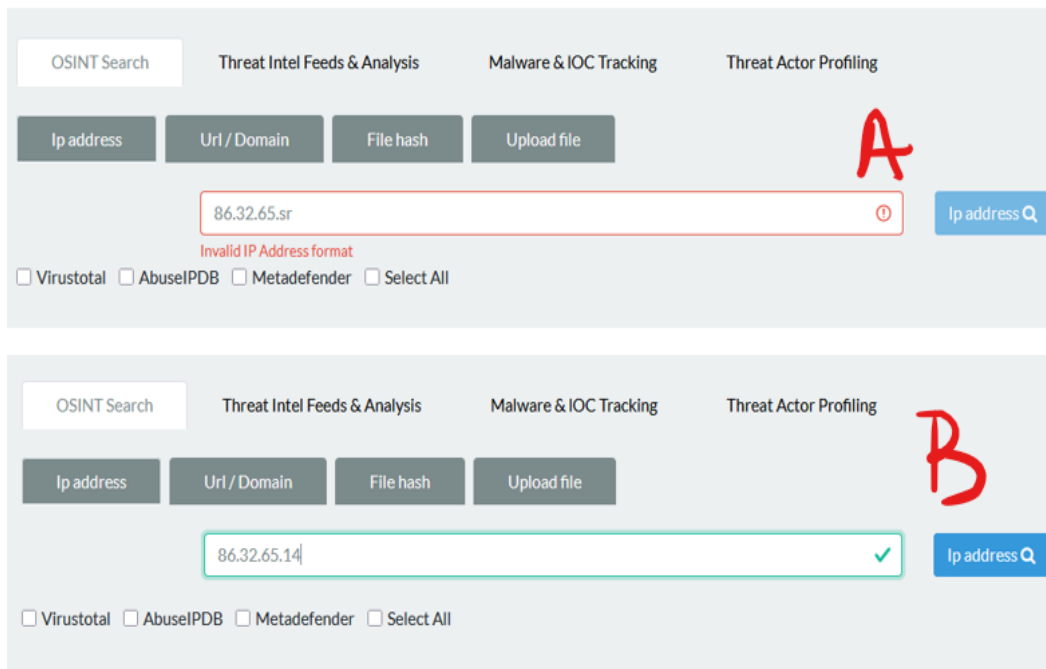


Fig. 6.2 IP Validation Test

6.3 Scenario-based Performance Evaluation

6.3.1 Design Rationale

The scenario-based evaluation was designed to produce an empirical comparison between the baseline (manual multi-platform investigation) and the intervention (*SecOps_360*-assisted investigation) for a representative investigation task. The task comprised querying three IoC types (IP address, domain/URL, file hash) against two threat-intelligence platforms (AbuseIPDB and VirusTotal for IP; MetaDefender and VirusTotal for hash; MetaDefender and VirusTotal for domain).

6.3.2 Scenario 1: Traditional Manual Investigation (without *SecOps_360*)

This scenario consists of running a baseline test to collect performance data in a situation where the SOC analyst conducts a threat investigation based on the logs provided by a SIEM tool. This implies recording the performance metrics, which would be set as a reference point against which *SecOps_360* would be compared. The baseline testing measures some components and performance metrics, such as CPU and memory utilisation, to help in the performance analysis of our system. This baseline test will particularly focus on time-related

The figure consists of two screenshots of a web application interface, labeled A and B. Both screenshots show a navigation bar with four tabs: 'OSINT Search', 'Threat Intel Feeds & Analysis', 'Malware & IOC Tracking', and 'Threat Actor Profiling'. Below the navigation bar are four buttons: 'Ip address', 'Url / Domain', 'File hash', and 'Upload file'. In both screenshots, the 'Url / Domain' button is active, and the input field contains the text 'bestdove.in.ua'. In screenshot A, the input field has a red border and a red error message 'Invalid URL domain format' below it. In screenshot B, the input field has a green border and a green checkmark to its right. Below the input field are four checkboxes: 'VirusTotal', 'AbuseIPDB', 'Metadefender', and 'Select All'. A blue 'UrlQ' button is located to the right of the input field in both screenshots.

Fig. 6.3 Domain Validation Test

parameters because of the criticality of time during the investigation of an incident. The sample logs 6.6 were collected from the Internet for this case study.

A- Testing Plan

Table 6.4 is the baseline test plan of Scenario 1, which indicates the support of the documents and requirements on which the test cases are going to be conducted. This test plan outlines a comprehensive approach for evaluating malware investigation processes without *SecOps_360*. The scope of the test is carefully defined to focus on core investigation tasks. These include raising a ticket, identifying key metrics, performing the investigation, and analysing the results. Notably, the plan explicitly excludes stress testing and capacity testing, maintaining a focused approach to the analyst's workflow and performance. The hardware and software environment for the test is precisely specified. Risk management is a key consideration in the test plan. It acknowledges the potential risks associated with malware investigation. The data preparation is crucial for the test's success. The plan ensures that necessary data, including suspicious IP addresses, URLs, domains, and file hashes, is collected and available for the investigation process. This preparation step is vital for simulating a realistic investigation scenario. The test scenario is detailed step-by-step, providing a clear roadmap for the investigation process. The analyst is required to perform searches, navigate between tabs, and record metrics at each stage. This meticulous approach allows for a comprehensive evaluation of the

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

Showing results for http://researchpurpose.net/

Checker Input

Show ☒ source ☒ outline ☒ image report ☒ check error pages User-Agent

Validator.nu/LV http://validator.w3.org/serv

Accept-Language

Check by

http://researchpurpose.net/

Document checking completed. No errors or warnings to show.

Used the HTML parser.

Total execution time 201 milliseconds.

Fig. 6.4 W3C Standard Test Result

time and effort required in a traditional investigation process. Finally, the plan specifies the deliverables expected from the test. Overall, this test plan provides a structured and thorough approach to evaluating the current state of malware investigation processes. Establishing a clear baseline sets the stage for a meaningful comparison with *SecOps_360*, potentially demonstrating the value and impact of the new solution in enhancing SOC operations.

B - Testing Scenario 1

Fig. 6.6 is sample data reported by a SIEM tool during the monitoring of a client infrastructure in an MSSP SOC. While continuously monitoring security events, a Tier 1 analyst triaged an alert and identified a potential security incident. The Tier 1 analyst gathers and documents preliminary information about the incident and escalates it to a Tier 2 analyst to:

- Verify the Incident: Confirm the validity of the incident based on the initial analysis and evidence provided by the Tier 1 analyst.
- Analyse Initial Data: Review logs, alerts, and any preliminary findings documented by the Tier 1 analyst.

If confirmed, the incident will be escalated to an Incident Response (IR) team following the SOC Incident Creation Process (Appendix C).

Document	6.4
Project	Test Plan for Malware Investigation without <i>SecOps_360</i>
Objective	To define the baseline metrics (computer performance and time spent) during a malware investigation by SOC analysts.
Scope	- Follow the incident Playbook as inspired by Onwubiko and Ouazzane (2020) - Raise a ticket - Identify the metrics and perform the test - Analyse the results
Out of Scope	- Stress test: to test the reaction of the system under extreme load conditions - Capacity test: to check the number of users the system can handle at a time - Endurance test: to test if the web application can cope with the expected load for an extended time
Hardware	Lenovo L340-17IWL Laptop (ideapad) Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99 GHz 16.0 GB RAM
Software	- Main host Operating system: Windows 11 Home Version 21H2 - Virtual machine (VMware Workstation Pro) Operating System Windows 7 Starter, Processor: Intel(R) Core(TM)i7-8565U, CPU @ 1.80 GHz 1.99 GHz, Installed memory (RAM) 1 GB
Testing Tool	Inspector Google, Network Performance
Environment	- Firefox Version 115.12.0esr (32-bit) - VMware Workstation 15 Pro Version 15.5.7 build-17171714
Risk	Malware could accidentally be executed on the host system instead of the VM, leading to infection Information on the host machine could be exposed if the malware manages to escape the VM Malware could attempt to exploit vulnerabilities in the hypervisor to escape the VM and infect the host system.
Data Planning and Preparation	- Ensure no other applications are running - Collect suspicious IP addresses from the log data - Collect the suspicious URLs / Domains from the log data - Collect the suspicious file hash (MD5/SHA)
Test Scenario	- Open the Firefox browser - Run the IP search on AUSEIPDB (https://www.abuseipdb.com/) - Right-click/Inspect - Navigate to the Network tab to read the performance - Record metrics - Open another tab in the browser - Run the IP search on VirusTotal (https://www.virustotal.com/gui/home/upload) - Record the metrics - Open another tab, run a hash check on VirusTotal and record the metrics - Open another tab, run a hash check on MetaDefender (https://metadefender.opswat.com/) and record the metrics - Open another tab, run the URL/domain search on at least 2 OSINTs and record the metrics - Run the Task Manager, record the CPU and memory metrics
Deliverables	- The Test Plan (This document) - The Test Results (data collected during the execution of the performance test) - The Test Evaluation (analysis of the findings)

Table 6.4 Baseline Performance Test Plan

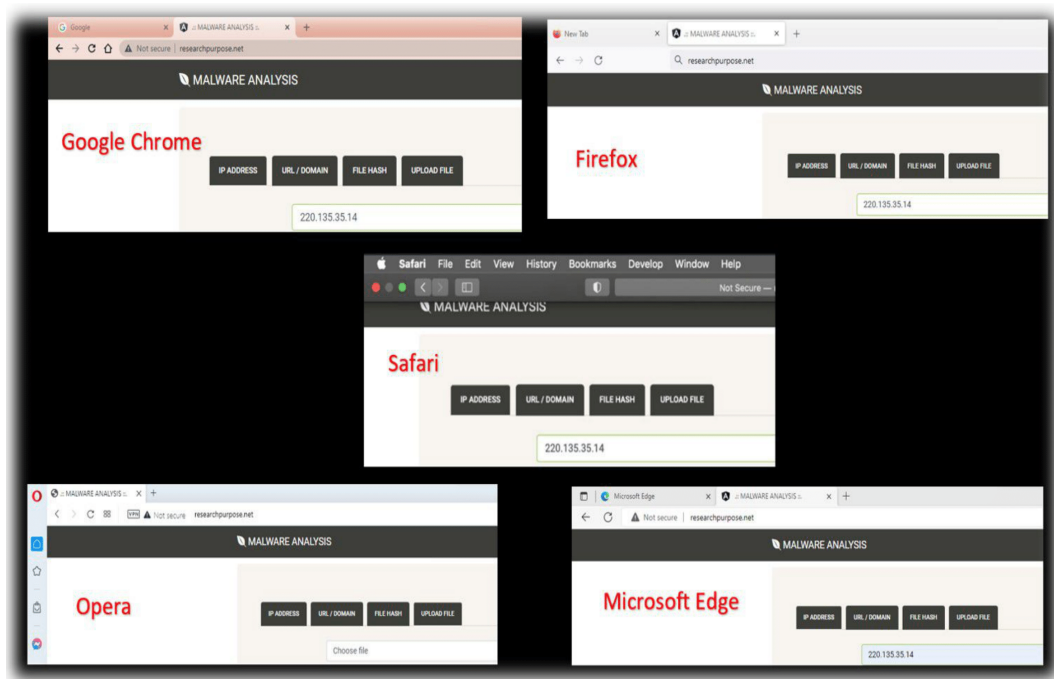


Fig. 6.5 Browser Compatibility Test Results

C - Results

Table 6.5 presents the performance test results for Scenario 1, which involves conducting searches for an IP address, a URL/domain, and a file hash across multiple websites. Each search involves opening a new tab in a browser. For instance, a search on IP reputation will involve opening a tab for AbuseIPDB and another one for VirusTotal. The same process will be applied to the URL/domain search and other similar tasks. The results of the table show the following:

- IP Address Search (121.202.155.79): Search on abuseipdb.com (24.74 seconds) and virustotal.com (10.33 seconds). Total time for IP searches: 35.07 seconds
- URL/Domain Search (alvopor-tas[.]com[.]br): Search on metadefender.opswat.com (9.24 seconds) and virustotal.com (12.11 seconds). Total time for URL/domain searches: 21.35 seconds
- Hash Search (0a73291ab5607aef7db23863cf8e72f55bcb3c273bb47f00edf011515aeb5894): Search on metadefender.opswat.com (32.13 seconds) and virustotal.com (24.74 seconds). Total time for hash searches: 56.87 seconds
- Total Time: 96.56 Seconds

```

2023-04-17T14:32:21.867Z [OpsInsights/Microsoft.Azure.Sentineldata/LogCollector.Output]
[ { "TenantId": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx", "SourceSystem": "Microsoft Cloud
App Security", "MG": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx", "ManagementGroupName":
"My Management Group", "TimeGenerated": "2023-04-17T14:32:21.867Z", "Computer":
"john.john", "Type": "IdentityLogEntry", "ProcessingLevelReasonText": "Suspicious file
download detected", "Category": "DataGovernanceRP", "OperationName":
"FileDownload", "LogSeverity": "High", "DvcHostname": "DESKTOP-ABC1234",
"DvcIpAddr": "10.0.0.5", "DvcMacAddr": "00-1B-44-C7-B2-3C", "DvcSubscriptionId":
"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx", "FilePath": "/temp/Ko%ne&.Exe", "FileHash": {
"Algorithm": "SHA256", "Value":
"0a73291ab5607aef7db23863cf8e72f55bcb3c273bb47f00edf011515aeb5894" },
"SourceIpAddr": "121.202.155.79", "SourcePort": "9001", "HttpReferrer": "http://alvopor-
tas.com.br", "Url": "http://alvopor-tas.com.br/Ko%ne&.Exe", "HttpUserAgent": "Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/58.0.3029.110 Safari/537.3", "FileName": "Ko%ne&.Exe", "SHA1":
"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b", "MD5":
"89aac6c8259abbba2fa2ad3fcefc6e", "Description": "User John John downloaded a
suspicious file", "EventVendor": "Microsoft", "EventProduct": "Cloud App Security",
"EventId": "CA200005" }]

```

Fig. 6.6 Scenario 1 Event Logs

The cumulative load time across all 6 searches is 96.56 seconds (about 1 minute and 36.6 seconds)

Key observations: The longest individual search time was for the hash on metadefender.opswat.com (32.13 seconds). The quickest search was for the URL/domain on metadefender.opswat.com (9.24 seconds). VirusTotal.com was used for all three types of searches, with varying load times. There's significant variability in load times across different websites and search types. This data highlights the time-consuming nature of manual searches across multiple platforms in traditional malware investigation processes, as we allocate a tab for each bit of information (6 tabs for Scenario 1). The total time of nearly 2 minutes for just these basic searches demonstrates the potential for inefficiency in this approach, especially when dealing with multiple incidents or more complex investigations. This underscores the need for a more integrated and efficient platform to streamline these searches and reduce overall investigation time.

Fig D.1 is one of the screenshots of the scenario 1 results documented in Appendix D. The image presents the results of an IP address search on AbuseIPDB for IP address 121.202.155.79, which is part of the scenario 1 test plan for malware investigation without *SecOps_360*. This search exemplifies a typical traditional SOC analyst workflow step when

Search	Website	Load Time (s)
IP 121.202.155.79	https://www.abuseipdb.com/	8.05
IP 121.202.155.79	https://www.virustotal.com/	10.33
URL/DOMAIN alvopor-tas[.]com[.]br	https://metadefender.opswat.com/	9.24
URL/DOMAIN alvopor-tas[.]com[.]br	https://www.virustotal.com/	12.11
HASH 0a73291ab5607aef7db 23863cf8e72f55bcb3c 273bb47f00edf011515 aeb5894	https://metadefender.opswat.com/	32.13
HASH 0a73291ab5607aef7db 23863cf8e72f55bcb3c 273bb47f00edf011515 aeb5894	https://www.virustotal.com/	24.74
	Total	96.6

Table 6.5 Scenario 1 Performance Test Results

investigating potentially malicious IP addresses. The results show that this IP address has been flagged as problematic in the AbuseIPDB database. Notably, it has been reported 1,235 times, with a 100% confidence of abuse. This high number of reports and the maximum confidence score strongly suggest that this IP is associated with malicious activities. Additional details provided by AbuseIPDB offer context about the IP:

It is associated with SmarTone Mobile Communications Limited, an ISP in Hong Kong. The hostname is *m121-202-155-79.smartone.com*, and the domain name is *smartone.com.hk*. The IP is geolocated to *Hong Kong*.

This information is valuable for SOC analysts as it provides a quick overview of the IP's reputation and origin, which can be crucial in determining a threat's nature and potential impact. The network performance data at the bottom of the image indicates that the page took 8.05 seconds to finish loading. This load time is an important metric in the scenario 1 test, as it represents part of the time an analyst would spend waiting for results during a manual investigation process. Such timings, when aggregated across multiple searches and sources, contribute to the overall efficiency of traditional manual investigation methods. This example illustrates the type of information gathering and analysis that SOC analysts perform in their investigations, highlighting both the wealth of information available and the time investment required in manual searches across different platforms.

Additionally, image D.5 displays the results of a hash analysis conducted on MetaDefender Cloud (OPSWAT) as part of the scenario 1 malware investigation process. The hash being analysed is *51569FE4B318CFD40DD0CF88497F7CC651144755*, representing a specific file or malware under investigation. The results provided by MetaDefender offer several critical insights for a SOC analyst:

- *Threat Identification*: The file is identified as "*Trojan/Wcry!728b76*", indicating it's likely associated with the WannaCry ransomware family. This immediate classification helps the analyst understand the potential severity and nature of the threat.
- *File Mismatch*: The platform flags a mismatch between the file's type and its extension, noting that the actual file type is *Z/GZ/GZIP*. This discrepancy is a common tactic used by malware authors to disguise malicious files and could be an additional indicator of malicious intent.
- *Infection Severity*: The analysis reveals that the archive was detected as infected by 33 out of 39 scanning engines. This high detection rate (about 85%) across multiple engines provides strong confidence in the file's malicious nature.

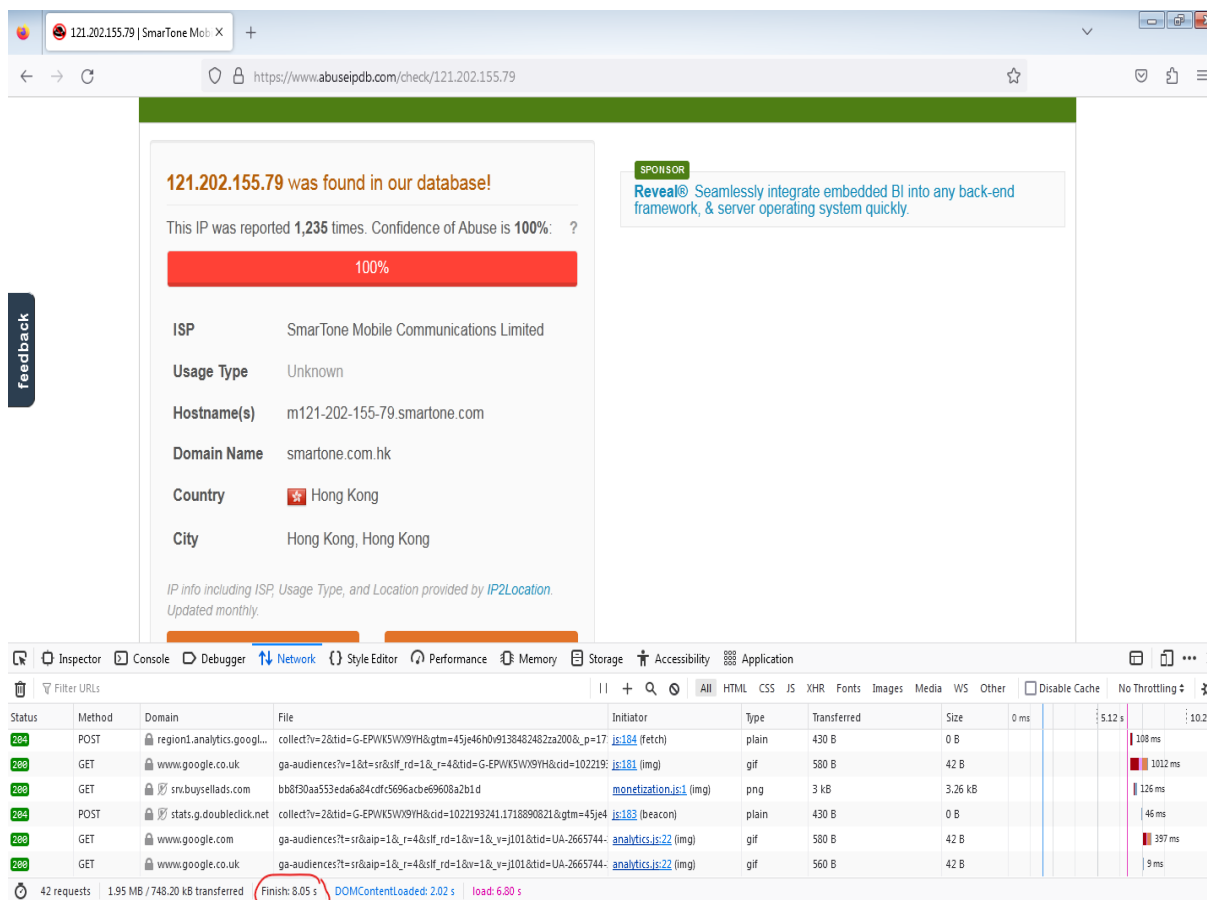


Fig. 6.7 Scenario 1 IP Search Results on AbuseIPDB

- *Content Analysis*: The results indicate that the archive contains files detected as infected by up to 34 engines, suggesting that multiple malicious components may be present within the archived file.
- *Metascan Results*: The "Threats detected" section shows 33/39 engines flagging this file, visually emphasising the high detection rate.
- *Lack of Dynamic Analysis*: The "Sandbox Score" section shows "No dynamic analysis performed," indicating that while static analysis has been conducted, the file's behaviour in a controlled environment hasn't been evaluated.

The network performance data at the bottom of the image shows that this page took 32.13 seconds to finish loading. These timings are significant in the context of scenario 1 test, as they represent the time an analyst would spend waiting for results during a manual investigation process. This detailed analysis from MetaDefender provides crucial information

for the SOC analyst, confirming the file’s malicious nature and offering specific details about the threat. However, the time taken to load and analyse this information highlights the potential inefficiencies in manual investigation processes, especially when dealing with multiple hashes or indicators of compromise.

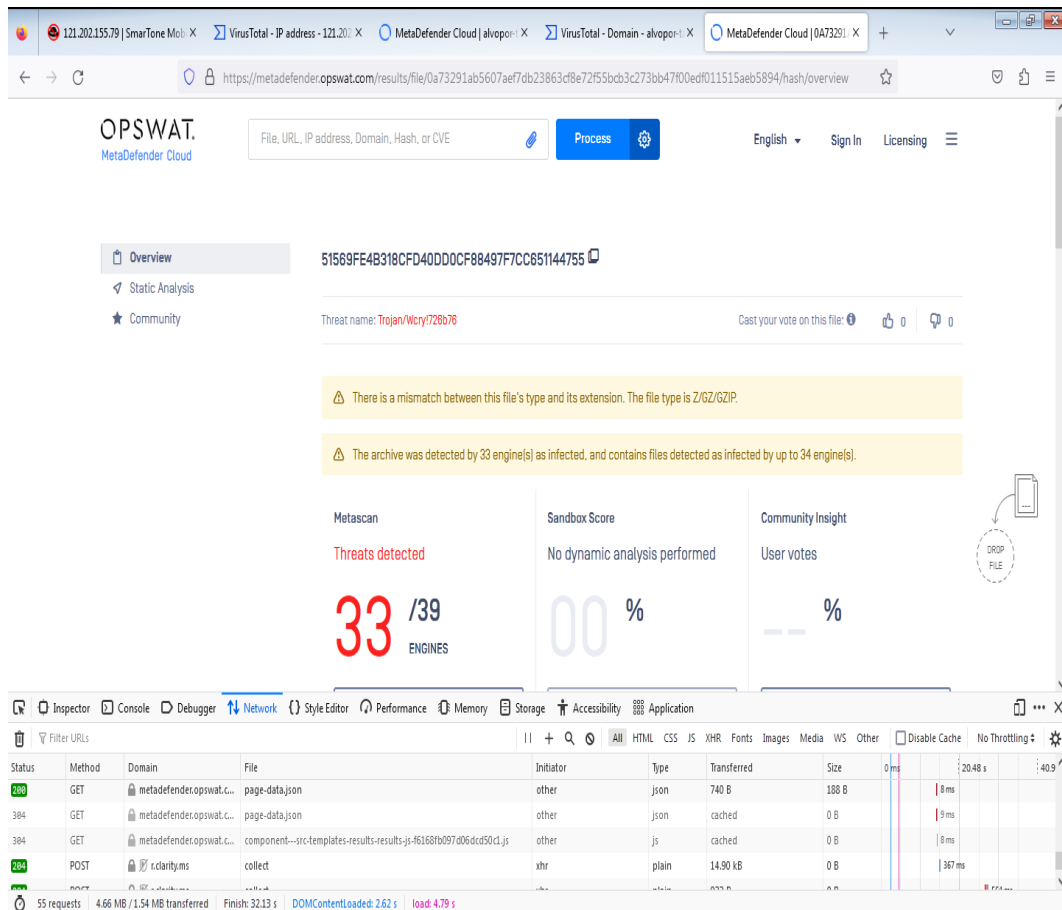


Fig. 6.8 Scenario 1 Hash Search Results on MetaDefender

Fig. 6.9 illustrates the The Windows Task Manager performance tab provides insights into the system’s resource utilisation during Scenario 1 of the malware investigation process. The scenario involves having 6 search tabs open in Firefox to search for an IP address, a hash, and a domain. Here’s a detailed comment on the workload representation: The CPU usage is quite high at 85%, indicating that the system is under significant load. This high CPU utilisation is likely due to the multiple Firefox tabs running simultaneous searches, which can be resource-intensive. The physical memory usage is 92%. This high memory consumption is consistent with having multiple browser tabs open, as web browsers, especially with various tabs, are known to be memory-intensive. There are 16,017 handles and 653 threads running

across 52 processes. Both the CPU and memory usage history graphs show sustained high utilisation, consistent with the ongoing search operations.

This scenario demonstrates the resource-intensive nature of traditional malware investigation methods, where multiple separate searches need to be conducted simultaneously. The high CPU and memory usage suggest that this approach may lead to system slowdowns and potentially impact the efficiency of the investigation process. It highlights the need for a more streamlined and resource-efficient solution for malware incident management in SOCs, which is what the proposed web-based platform aims to address.

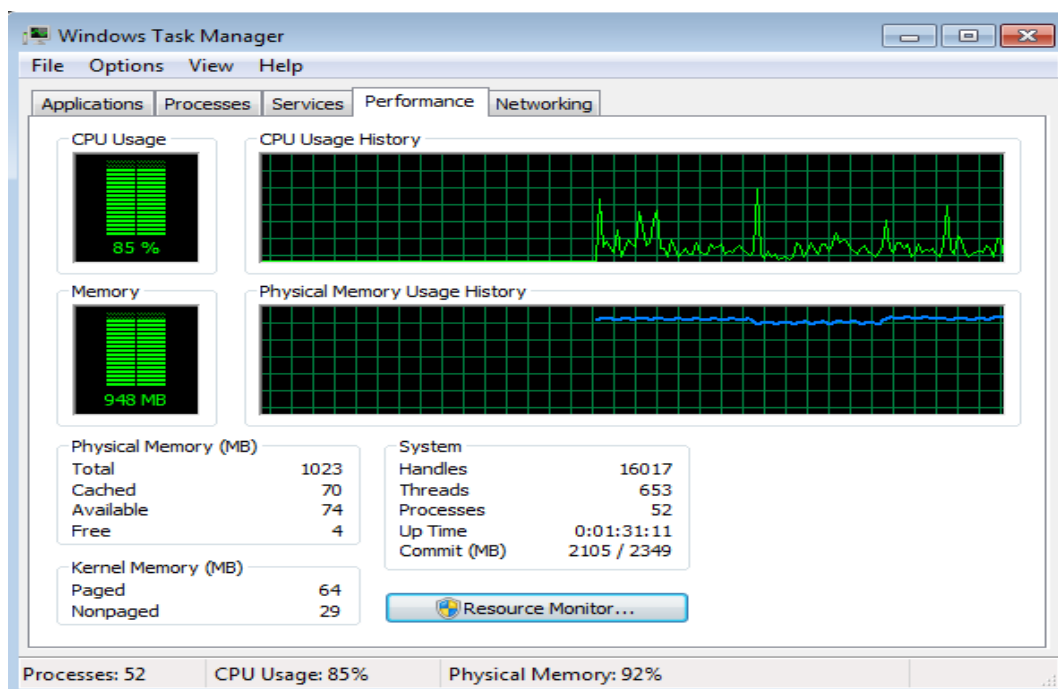


Fig. 6.9 Scenario 1 Work Load

6.3.3 Scenario 2: *SecOps_360*-Assisted Investigation

In this section, we explore Scenario 2, where the malware investigation is conducted using *SecOps_360*. By employing the same case study as in Scenario 1, which involved a traditional malware investigation, we aim to demonstrate the enhanced capabilities and efficiencies provided by our platform. This comparison allows for a direct evaluation of the platform's effectiveness in streamlining the investigative process and improving detection, workload, and response times.

A - Test Plan

Table 6.6 represents the test plan for the *SecOps_360* platform hosted at www.researchpurpose.net including test objectives, test approach, environment, and Scenario. Scenario 2 introduces the enhanced application's impact on ticket management processes. The test plan for this scenario measures the same metrics but with the application's features in place. The test plan includes an evaluation of the application's integration capabilities with existing SOC tools and its impact on reducing analyst workload while improving incident response times. By comparing these scenarios, the test plan provides quantifiable evidence of the application's effectiveness in enhancing SOC ticket management efficiency.

B - Testing Scenario 2

In this scenario, we conduct a comprehensive evaluation of malware investigation processes using *SecOps_360*. The testing environment mirrors the setup from Scenario 1 to ensure a valid comparison, utilising the same hardware specifications and network conditions. The platform is deployed in a controlled laboratory environment. The scenario begins with the detection of potential malware activity. Upon detection, the platform is used to initiate the investigation workflow. SOC analysts interact with the unified dashboard, which provides centralised access to data sources. The investigation process follows a structured sequence: initial triage, malware analysis, impact assessment, and remediation planning, as in the process of Scenario 1. Throughout the scenario, we maintain detailed logs of analyst interactions with the platform, system performance metrics, and investigation outcomes. This data collection approach ensures comprehensive coverage of both quantitative performance metrics and qualitative aspects of the investigation process, providing a robust foundation for comparison with the baseline scenario. The testing procedure emphasises real-world applicability by incorporating an investigation scenario that SOC analysts typically encounter.

C - Results

Table 6.7 presents the performance test results for Scenario 2, which utilises *SecOps_360* for malware investigation. The table shows the outcomes of searching for the same indicators of compromise (IoCs) as in Scenario 1, but using the *SecOps_360*. The results are as follows:

- IP Address Search Load Time: 2.99 seconds
- URL/Domain Search Load Time: 1.65 seconds
- Hash Search Load Time: 1.83 seconds

Document	6.6
Project	Test Plan for Malware Investigation with <i>SecOps_360</i>
Objective	To define and compare the baseline metrics (computer performance and time spent) during a malware investigation by SOC analysts.
Scope	- Follow the incident Playbook as inspired by Onwubiko and Ouazzane (2020) - Raise a ticket - Identify the metrics and perform the test - Analyse the results
Out of Scope	- Stress test: to test the reaction of the system under extreme load conditions - Capacity test: to check the number of users the system can handle at a time - Endurance test: to test if the web application can cope with the expected load for an extended time
Hardware	Lenovo L340-17IWL Laptop (ideapad) Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99 GHz 16.0 GB RAM
Software	- Main host Operating system: Windows 11 Home Version 21H2 - Virtual machine (VMware Workstation Pro) Operating System: Windows 7 Starter, Processor: Intel(R) Core(TM)i7-8565U, CPU @ 1.80 GHz 1.99 GHz, Installed memory (RAM) 1 GB
Testing Tool	Inspector Google, Network Performance
Environment	- Firefox Version 115.12.0esr (32-bit) - VMware Workstation 15 Pro Version 15.5.7 build-17171714
Risk	Malware could accidentally be executed on the host system instead of the VM, leading to infection Information on the host machine could be exposed if the malware manages to escape the VM Malware could attempt to exploit vulnerabilities in the hypervisor to escape the VM and infect the host system.
Data Planning and Preparation	- Ensure no other applications are running - Collect suspicious IP addresses from the log data - Collect the suspicious URLs / Domains from the log data - Collect the suspicious file hash (MD5/SHA)
Test Scenario	- Open the Firefox browser - Go to https://www.researchpurpose.net/ - Run the IP search in the IP ADDRESS tab, - Right-click/Inspect - Navigate to the Network tab to read the performance - Record metrics - Select the URL/DOMAIN tab and run the domain search - Record the metrics - Select the HASH tab and run the hash search - Record the metrics
Deliverables	- The Test Plan (This document) - The Test Results (data collected during the execution of the performance test) - The Test Evaluation (analysis of the findings)

Table 6.6 *SecOps_360* Test Plan

Search	Website	Outcome	Load Time (s)
IP 121.202.155.79	https://www.researchpurpose.net/	Loaded both search results from AbusIPDB and VirusTotal on the same page	2.99
URL/DOMAIN alvopor-tas[.]com[.]br	https://www.researchpurpose.net/	Loaded both search results from MetaDefender and VirusTotal on the same page	1.65
HASH 0a73291ab5607aef7db 23863cf8e72f55bcb3c 273bb47f00edf011515 aeb5894	https://www.researchpurpose.net/	Loaded both search results from MetaDefender and VirusTotal on the same page	1.83
		Total	6.47

Table 6.7 Scenario 2 Performance Test Results

The cumulative load time for all three searches is 6.47 seconds. All searches were conducted *SecOps_360* through the website (<https://researchpurpose.net/>). Each search aggregated results from multiple sources on a single page, eliminating the need for separate tabs or manual correlation. Load times are significantly reduced compared to Scenario 1, with the longest search taking only 2.99 seconds. The total time for all searches (6.47 seconds) is drastically lower than the 96.6 seconds required in Scenario 1.

This data demonstrates the efficiency and time-saving potential of *SecOps_360*, showing substantial improvements in search speed and result aggregation compared to the traditional method used in Scenario 1.

Fig 6.10 is the Windows Task Manager performance view, showing CPU and memory usage statistics for scenario 2 test. The results show that CPU usage is at 23%, with some fluctuations shown in the history graph. The physical memory (RAM) usage is at 78%. The memory usage history shows a fairly consistent level of utilisation over time.

The Fig D.7 image displays the results of a malware analysis performed through our web application. The interface shows two main sections containing responses from an API request regarding a specific IP address (121.202.135.79). The rest of the test results are evidenced in Appendix C. The first section, labelled "AbuseIPDB Check on IP Address", provides details about the queried IP. It reports that this IP has been flagged for abuse 356 times with a 100% confidence level. The IP is associated with SmartTone Mobile Communications Limited

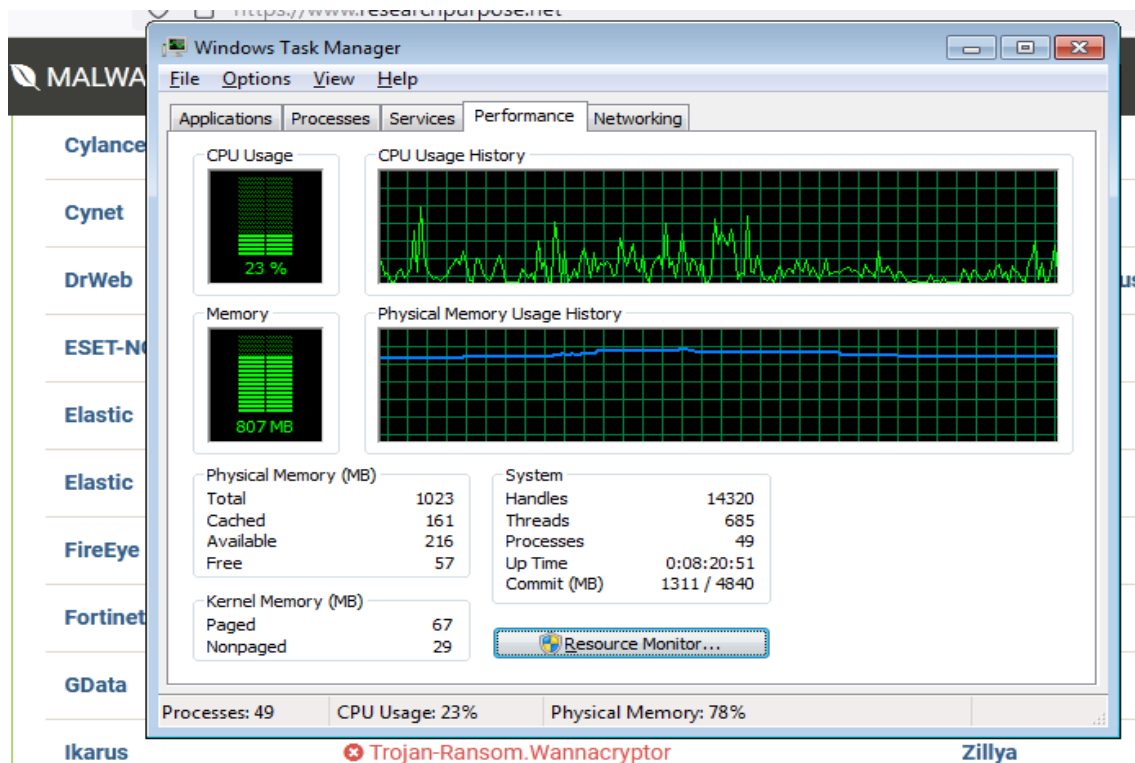


Fig. 6.10 Scenario 2 Work Load

and is located in Hong Kong (HK). "AbuseIPDB blacklist check" shows additional network information, including the IP address range and network mask. It also indicates that there are 256 possible host addresses within this range.

The second section is a "VirusTotal" report, which presents scan results from various antivirus and security vendors. This comprehensive list shows the detection status for each service, with most reporting the IP as clean. Our tool appears to be designed to aggregate data from multiple sources to provide a holistic view of security status and the history of malicious activity. The interface is clean and user-friendly, allowing for easy interpretation of the results. The colour-coded indicators (green for clean, red for malicious, etc.) help quickly identify potential issues across the various security vendors' assessments. At the bottom of the image, there's a finish time displayed showing 2.99 s. This indicates that the entire malware analysis process, including querying multiple security databases and compiling the results, was completed in 2.99 seconds. This quick response time is noteworthy, as it demonstrates the efficiency of the web application in gathering and presenting complex security information from various sources. The speed of the analysis is particularly valuable for professionals who need to rapidly assess multiple IoCs or perform real-time threat analysis. Such a quick turnaround allows for more responsive threat detection and mitigation strategies.

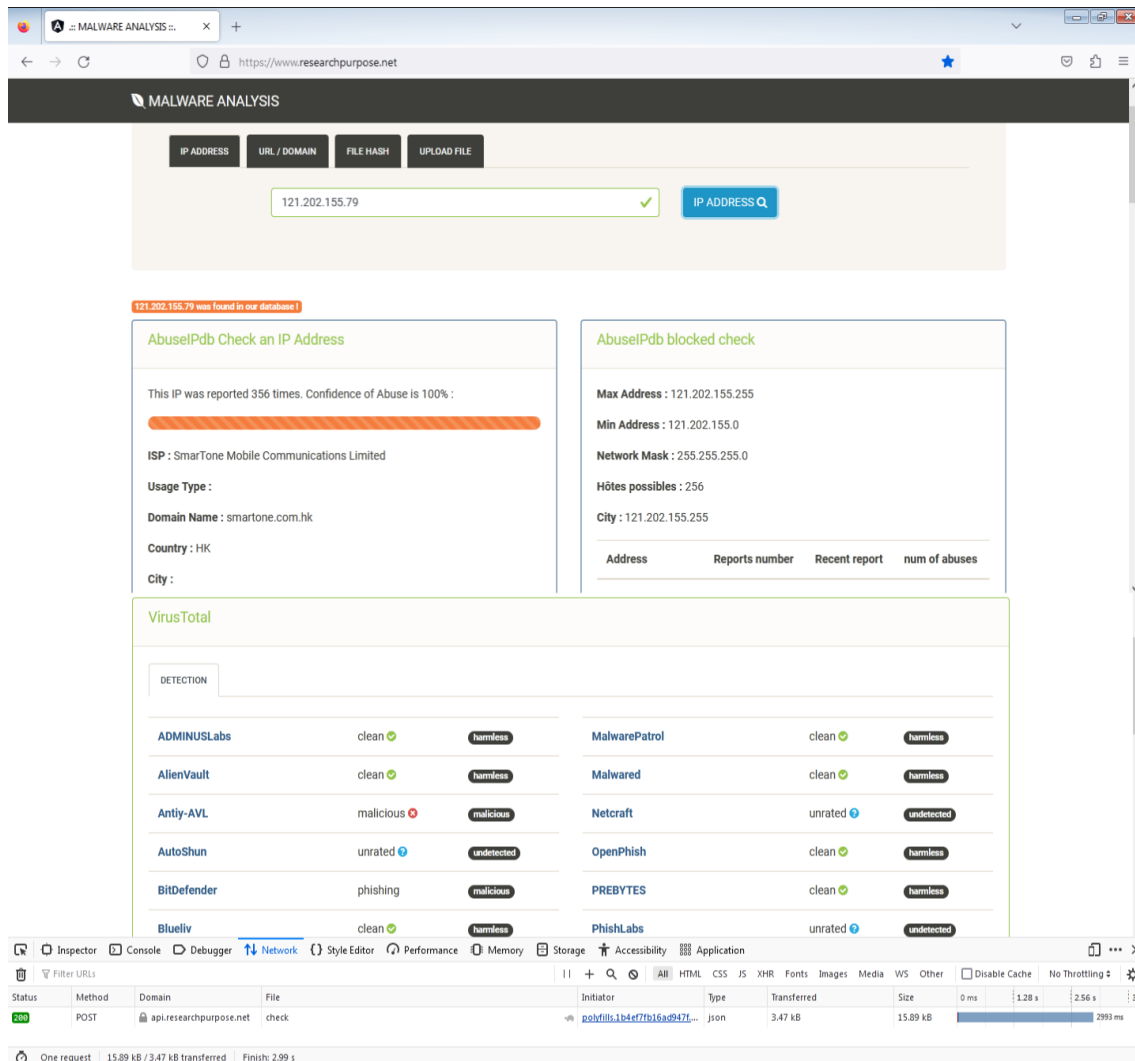


Fig. 6.11 Scenario 2 IP Search Results

Similarly, the Hash search in D.9 showcases the results from 2 specific external sources via an API call. The VirusTotal section offers insights from specific antivirus engines. Further corroboration comes from the MetaDefender results. This high consensus among different security services leaves little doubt about the file's malicious nature. The efficiency of this analysis tool is evident in the completion time displayed at the bottom of the image: a mere 1.83 seconds. This rapid processing demonstrates the power of modern cybersecurity tools in quickly aggregating and presenting critical threat information from multiple sources. The evidence presented in this case strongly indicates that the hash of the analysed file is indeed dangerous, warranting immediate attention and appropriate security measures.

applied. The evaluation does not address usability, the impact on analyst cognitive load, or performance under realistic SOC alert volumes.

Tab 6.8 is a comparative table of the traditional threat investigation and *SecOps_360*. This analysis compares two scenarios of threat investigation processes: the traditional method (Scenario 1) and our new system (Scenario 2). The comparison focuses on performance metrics, user experience, and overall efficiency in an SOC environment.

- The traditional method shows significant resource consumption and time inefficiency:

High CPU and memory usage indicate a heavy load on the system, potentially slowing down other critical SOC operations. The extended load time of nearly 2 minutes for investigating just three IoCs suggests a time-consuming process that could delay threat response. Multiple browser tabs (6 in this case) consume more resources and require the analyst to switch between tabs and collate information manually.

- *SecOps_360* demonstrates significant improvements:

Drastically reduced load time (94% reduction) enables rapid threat assessment and response. Substantially lower CPU usage (62% reduction) indicates more efficient processing and querying methods. The slightly reduced memory usage suggests better resource management. The consistent number of processes implies that the efficiency gains are not from offloading tasks but from improved architecture and data retrieval methods.

Comparative Advantages of *SecOps_360*

Efficiency and Speed: *SecOps_360*'s total load time of 6.47 seconds versus 96.6 seconds in the traditional method represents a 94% reduction in investigation time. This speed improvement allows for quicker threat detection and response, crucial in mitigating potential security breaches.

Resource Optimisation: The 62% reduction in CPU usage (23% vs. 85%) indicates that *SecOps_360* is significantly more efficient in processing and analysing threat data. Lower resource consumption allows for smoother multitasking and the ability to run more concurrent investigations without system slowdowns.

Scalability: The efficient resource utilisation of the *SecOps_360* suggests better scalability for handling larger volumes of threat investigations simultaneously.

Accuracy and Comprehensiveness: By automatically aggregating data from multiple sources, *SecOps_360* reduces the risk of missing critical information that might occur in man-

ual searches. Consistent data presentation across different IoC types enhances comparative analysis capabilities.

User-Centric Design: The unified interface reduces the cognitive load on analysts, allowing them to focus on threat analysis rather than navigating between multiple websites. Standardised data presentation across different IoC types facilitates quicker pattern recognition and decision-making.

API Integration: Utilising APIs for data retrieval ensures more up-to-date information compared to potentially cached web pages in browser-based searches. API calls allow for more structured data retrieval, enabling better data parsing and analysis capabilities.

Automation Potential: *SecOps_360*'s architecture lays the groundwork for further automation in threat investigation and preliminary analysis.

Adaptability: The modular nature of API-based systems allows for easier integration of new threat intelligence sources as they become available.

In conclusion, *SecOps_360* represents a significant innovation in threat investigation processes within a SOC environment. Its key advantages lie in dramatically improved efficiency, reduced cognitive load on analysts, and the potential for enhanced accuracy in threat assessment. *SecOps_360*'s design addresses many of the bottlenecks present in traditional methods, particularly in terms of time consumption and resource utilisation. By streamlining the threat intelligence gathering process, *SecOps_360* allows SOC analysts to shift their focus from manual data collection to high-value analytical tasks. This shift can lead to more proactive threat hunting, faster incident response, and overall improved security posture for the organisation. The substantial performance improvements, coupled with the potential for further automation and integration, position *SecOps_360* as a transformative tool in modern cybersecurity operations. It not only enhances the current capabilities of SOC teams but also paves the way for more advanced, AI-driven threat analysis in the future. In an era where the speed and accuracy of threat response can be the difference between a minor incident and a major breach, innovations like *SecOps_360* are crucial. It represents a forward-thinking approach to cybersecurity that aligns with the increasing complexity and volume of cyber threats, making it a valuable asset for any modern SOC.

The comprehensive analysis of Scenario 1, which simulates a traditional malware investigation process, provides compelling evidence for the need to improve the approach to incident management in SOCs. Through the examination of both system performance metrics and search time data, we have uncovered significant inefficiencies and potential bottlenecks in the current methodology. The Windows Task Manager performance data revealed

Aspect	Traditional Threat Investigation (Scenario 1)	<i>SecOps_360</i> (Scenario 2)
Methodology	Manual investigation with multiple browser tabs for IP addresses, domains, and file hashes	<i>SecOps_360</i> using API calls to query multiple databases simultaneously
Total Load Time	96.6 seconds	6.47 seconds
CPU Usage	92%	23%
Memory Usage	92	78
User Experience	Manual navigation, copy-pasting IoCs, prone to human error and cognitive overload	Unified interface, reduced cognitive load, improved decision-making speed and accuracy
Efficiency and Speed	Efficiency and Speed	94% reduction in load time, quicker threat detection and response
Resource Optimisation	High CPU usage, potential system slowdowns	62% reduction in CPU usage, smoother multitasking
Scalability	Limited by high resource usage	Better scalability due to efficient resource utilisation
Accuracy and Comprehensiveness	High risk of missing critical information due to manual searches	Reduced risk of missing information, consistent data presentation
User-centric design	Cognitive load due to multiple websites	Reduced cognitive load, standardised data presentation
API Integration	None	Ensures up-to-date information, structured data retrieval
Automation Potential	Low	Architecture supports further automation in threat investigation
Adaptability	Difficulty to integrate new sources	Easier integration of other new team modules due to the modular API-based system.

Table 6.8 Comparative Table of the Traditional Threat Investigation and the Proposed Innovative Platform

alarmingly high resource utilisation, with CPU usage at 85% and memory consumption at 92%. This level of system strain, resulting from just six concurrent search operations, raises concerns about the scalability and sustainability of such practices in a real-world SOC environment where multiple, complex investigations may be ongoing simultaneously with various SLAs. The near-exhaustion of system resources not only limits the efficiency of the current investigation but also hinders an analyst's ability to multitask or respond swiftly to emerging threats. Furthermore, the performance test results table illuminated the time-intensive nature of manual, multi-platform searches. With a cumulative search time of 96.6 seconds for basic threat intelligence gathering on a single set of indicators, it becomes evident that this approach is not conducive to rapid incident response. The variability in load times across different platforms, ranging from 8.05 to 32.13 seconds, introduces an element of unpredictability that can complicate workflow management and resource allocation in time-sensitive scenarios. These findings underscore the critical need for a paradigm shift in malware incident management. *SecOps_360* aims to address these challenges by integrating multiple search functions into a unified interface, potentially reducing both system resource overhead and cumulative search times. By streamlining the investigation process, such a platform could significantly enhance the efficiency and effectiveness of SOC operations. Moreover, the high resource utilisation observed suggests that the traditional method may be unsustainable as the volume and complexity of cyber threats continue to escalate. A more efficient, centralised solution would not only accelerate individual investigations but also enable SOCs to handle a higher volume of incidents without proportional increases in hardware resources or personnel. In conclusion, the data from Scenario 1 serves as a compelling argument for the adoption of innovative, integrated solutions in malware incident management. *SecOps_360* represents a promising step towards overcoming the limitations of traditional methods, potentially leading to more rapid threat detection, analysis, and response. As cyber threats evolve in sophistication and frequency, SOCS must evolve their methodologies accordingly. The transition to a more efficient, resource-conscious approach to incident management is not merely an optimisation but a necessity in the ongoing effort to maintain robust cybersecurity postures in the face of ever-growing digital threats.

6.5 Comparative Analysis with Related Work

SecOps_360 is contextualised against three related security platform studies, consistent with the approach used in Mutemwa, M. et al. (2018) and similar comparative analyses in the security platform literature. The comparison demonstrates that scenario-based eval-

uation is an established method for assessing security platforms in the absence of live operational data, whilst acknowledging its inherent limitations. The criteria-driven comparison of industry tools presented in Chapter 2 (Tab: 2.11) provides additional comparative context: *SecOps_360* occupies a distinct position (combining multi-source IoC integration, lightweight resource requirements, and open accessibility) not occupied by any existing commercial or open-source tool reviewed.

Our testing and evaluation approach focuses on scenario-based comparison and functional evaluation of *SecOps_360*. This section examines how our methodology relates to and differs from previous research in similar domains. We compare and contextualise the testing and evaluation methodologies applied in our research with related work in the domain of cybersecurity. Below, we will discuss relevant works, their evaluation methodologies, and how they compare to our approach. The most closely related works are:

6.5.1 Secure Web Browsing with the OP Web Browser

Grier et al. (2008) conducted an in-depth evaluation of the OP web browser, with a clear focus on two major aspects: performance evaluation and security analysis. Their testing method involved measuring page load times, analysing the size of audit logs for single page loads, and comparing these results with a well-established baseline browser, Firefox. This provides a useful benchmark for understanding the overhead introduced by the OP browser in terms of performance and security. In the performance evaluation, they measured page load times and audit log sizes to ensure that the browser does not introduce noticeable delays, while in the security analysis, they focused on assessing the browser's ability to protect user data against malicious websites.

Comparison with Our Work: While Grier et al. conducted performance and security evaluations on a specific software tool (the OP web browser), our evaluation is more comprehensive in the sense that it addresses a complete system for malware incident management within a SOC. Our testing includes real-world operational scenarios that examine *SecOps_360*'s impact on SOC workflows, such as triaging, prioritising, and incident resolution. Moreover, unlike Grier et al.'s performance testing, our approach considers the holistic impact of *SecOps_360* on incident management effectiveness, measuring both time to resolution and accuracy in prioritising malware incidents.

6.5.2 Software Security Design for Cyber-Physical Systems

Zoppelt (2021) work deals with the security design of software in cyber-physical systems (CPS), with a focus on the integration of security measures into the software development lifecycle. Although the research addresses crucial cybersecurity topics such as risk mitigation and design vulnerabilities, it does not focus on real-time evaluation of an operational platform or system like ours. Zoppelt’s evaluation methodology looks at software design and security integration in CPS environments rather than incident management within a SOC.

Comparison with Our Work: Our approach differs significantly from Zoppelt’s in that it is oriented toward evaluating a live platform’s performance within a real-world SOC environment. Zoppelt’s focus is on secure design processes for embedded systems, while our testing focuses on the practical impact of an integrated malware management platform on security operations teams. We assess how well *SecOps_360* supports and enhances SOC workflows through direct scenario comparisons, which are absent in Zoppelt’s work.

6.5.3 A Secure Computing Platform for Building Automation Using Microkernel-based Operating Systems

Wang (2018) Research focuses on the design of a secure computing system for building automation using microkernel-based operating systems. The evaluation process centres around testing the resilience and security of the platform in handling real-time and safety-critical applications. The performance testing for this platform emphasises its robustness under various failure conditions and attack vectors.

Comparison with Our Work: Wang’s work emphasises system-level security and resilience but does not focus on operational workflows like incident management within a SOC. Our work, on the other hand, directly addresses operational efficiency in cybersecurity incident handling, with a particular focus on malware detection and management. While Wang’s system is evaluated in terms of failure resilience and attack defence, our approach emphasises the introduction of *SecOps_360*, enhances SOC workflow performance, and reduces the time and effort needed to manage incidents effectively.

6.5.4 Justification for Our Testing Approach

Our testing approach, which involves comparing two real-world malware investigation scenarios (one with and one without *SecOps_360*), offers a more direct and measurable evaluation

of platform effectiveness than the methodologies used in the related works. While several of the reviewed papers focus on performance testing or security analysis within narrow contexts (e.g., web browsers or secure operating systems), our evaluation is unique in that it considers the operational context of a SOC, where response times, incident prioritisation, and workflow efficiency are critical. Furthermore, our use of empirical data allows us to directly compare how well our platform streamlines SOC operations, as opposed to the theoretical and software-specific evaluations seen in other works. This real-world, scenario-based approach provides a more holistic view of the platform's impact on cybersecurity incident management.

Our scenario-based evaluation methodology offers several advantages over previous approaches:

- **Practical Application:** Unlike purely theoretical evaluations, our scenarios demonstrate real-world applications and benefits.
- **Comparative Analysis:** By directly comparing traditional and platform-enhanced workflows, we provide clear evidence of operational improvements.
- **Comprehensive Coverage:** Our testing plan encompasses functionality, compatibility, and practical usage scenarios, providing a more complete evaluation than single-metric approaches.
- **Quantifiable Results:** Similar to Grier et al.'s approach, we provide measurable outcomes while extending beyond basic performance metrics to include operational efficiency indicators.

This related work analysis validates our choice of evaluation methodology while highlighting how our approach advances the field by providing a more comprehensive and orientated evaluation framework for SOC platforms.

6.6 Controlled Experiment with SOC Analyst Participants

6.6.1 Overview and Motivation

The scenario-based evaluation reported in Sections 6.2 and 6.3 established a useful, quantified baseline using a single-researcher, controlled environment. Whilst that evidence is directionally informative, it cannot constitute participant-based empirical validation: it involves no independent observers, no variation across analysts, and no measure of how real SOC analysts interact with the platform under realistic working conditions. This section addresses that

gap by presenting a controlled within-subjects experiment conducted with three practising SOC analysts, applying appropriate non-parametric statistical methods for the sample size. The experiment was designed to provide the most rigorous empirical evidence achievable within the practical constraints of this research, most notably, the difficulty of recruiting a larger cohort of practising SOC analysts, which is acknowledged as a systematic challenge across the usable-security research literature (Ortloff et al. (2023)). The resulting data, whilst limited in statistical power for generalisation, provides consistent directional evidence across all three participants and constitutes a meaningful empirical foundation for the claims made about *SecOps_360*.

6.6.2 Research Question

RQ-E1: Does *SecOps_360* significantly reduce IoC investigation time and system resource utilisation compared to manual multi-platform investigation, as measured in a controlled experiment with practising SOC analysts?

The controlled experiment was designed to address one focused empirical research question: does *SecOps_360* significantly reduce IoC investigation time and system resource utilisation compared to manual multi-platform investigation, as measured in a controlled experiment with practising SOC analysts (RQ-E1)? This question was derived directly from the two quantifiable gaps identified in the Systematic Literature Review (the absence of controlled measurements of investigation latency, and the absence of an empirically validated unified investigation platform) and from the preliminary scenario-based findings reported in Section 6.3, which provided directional evidence but lacked independent participant validation. The question is deliberately bounded: it does not ask whether *SecOps_360* is superior in every operational respect, nor whether it is preferable to commercial SIEM or SOAR platforms in a production environment. It asks only whether, under controlled and equivalent conditions, the platform produces a measurable and meaningful difference in the two performance dimensions most directly affected by its architectural design (investigation time and computational resource consumption). This precise framing ensures that the experimental findings can be interpreted clearly and honestly, without overstating what a three-participant study can and cannot demonstrate.

6.6.3 Experimental Design

A within-subjects design was adopted in which each participant completed both experimental conditions. This design is preferred over a between-subjects design when sample sizes are small because it eliminates inter-participant variability as a confounding factor: each analyst

serves as their own control, meaning that any observed difference in performance between conditions can be attributed to the condition itself rather than to differences between analysts (Field (2024)). Condition 1 (manual investigation) was administered before Condition 2 (*SecOps_360*-assisted investigation) for all participants, following the protocol set out in the test requirements distributed to analysts. This ordering is deliberately conservative in design terms: any learning effect from having already investigated the same IoC set in Condition 1 would benefit the manual condition's relative performance, making it harder to observe an advantage for *SecOps_360*. Consistent improvement under *SecOps_360* despite this ordering constraint strengthens the internal validity of the findings (Turner et al. (2021)). The independent variable was the condition: Condition 1 (manual multi-platform OSINT investigation) or Condition 2 (*SecOps_360*-assisted investigation, temporarily hosted at <http://researchpurpose.net/> for the duration of the experiment). Three dependent variables (DV) were measured:

DV1: Task completion time (seconds): total elapsed time from beginning the first query to recording the final classification verdict across all three IoCs. In Condition 1, this comprised the cumulative sum of page-load times for six platform queries, as measured using the Firefox Network Inspector; in Condition 2, it comprised the single unified query time returned by the *SecOps_360* platform.

DV2: IoC classification accuracy: the proportion of the three IoCs correctly classified as malicious or benign, out of three possible correct answers per condition.

DV3: System resource utilisation: peak CPU utilisation (%) and peak memory utilisation (%), recorded by each participant using Task Manager (Windows: Ctrl+Shift+Esc → Performance tab) or Activity Monitor (macOS: Applications → Utilities → Activity Monitor) at the point of highest observed usage during the investigation. This metric reflects the computational overhead imposed by each investigative approach on a standard working laptop.

6.6.4 Participants

Three practising SOC analysts participated voluntarily as professional colleagues of the researcher and are anonymised throughout as Analyst A, Analyst B, and Analyst C. The group comprises two Tier 2 analysts (Analysts A and C) and one Tier 1 analyst (Analyst B). Their experience in operational SOC roles ranges from two to six years, providing a degree of variation relevant to the research context: Tier 1 analysts perform initial alert triage and routine IoC lookups, whilst Tier 2 analysts conduct deeper incident investigation and are typically more familiar with a broader range of OSINT platforms. Tab 6.9 summarises the participant profiles.

Analyst	SOC Tier	Experience (yrs)	Primary Responsibilities	OSINT Platform Familiarity
Analyst A	Tier 2	~ 4years	Incident investigation, threat intelligence analysis	AbuseIPDB, VirusTotal (regular); MetaDefender, IPalyzer
Analyst B	Tier 1	~ 2years	Alert triage, initial IoC lookup, escalation	AbuseIPDB, VirusTotal (regular); MetaDefender, IPalyzer
Analyst C	Tier 2	~ 6years	Incident investigation, threat intelligence analysis	AbuseIPDB, VirusTotal (regular); MetaDefender, IPalyzer (occasional)

Table 6.9 Participant Profile Summary

A sample of three is openly acknowledged to be below the minimum typically recommended for broad inferential generalisation. This constraint arises from the well-documented difficulty of recruiting practising SOC analysts (a small, highly specialised population with significant operational demands) for academic research participation. (Ortloff et al. (2023)), in their systematic study of sample sizes in developer-centred usable security research, document this as a consistent and recognised challenge, noting that many published studies in this domain are underpowered yet still provide valuable directional evidence sufficient to justify investment in larger-scale validation. The present experiment is understood in that spirit: it provides consistent empirical evidence across all participants, and its limitations are acknowledged explicitly in Section 7.4. Effect sizes are reported alongside significance values to characterise the practical magnitude of the observed differences, following best practice for small-sample security studies (Ortloff et al. (2023)). All participants conducted the experiment on their own personal laptops using a standard broadband internet connection. Personal laptops under typical web-browsing workloads exhibit materially lower CPU and memory baseline utilisation than our controlled virtual machine environment used in the preliminary scenario testing (Sections 6.2 and 6.3). Accordingly, the absolute resource figures reported in this section are not directly comparable to the earlier scenario figures; within-participant comparisons between the two conditions on the same machine remain methodologically valid and are the appropriate unit of analysis here.

6.6.5 Ethical Considerations

Participation was entirely voluntary and based on informed understanding of the research purpose. Each participant received a written test requirements document explaining the research aims, the tasks required, the data to be recorded, and how that data would be used

and stored. No financial or other inducement was offered. Participants were free to withdraw at any point without consequence. No personal data beyond anonymised performance measurements was collected. Participants are identified only by the labels Analyst A, B, and C; no names, job titles, employer details, or any other identifying information appear in this thesis or its appendices. Data were anonymised before any analysis was conducted. All measurements were stored on the researcher's password-protected institutional storage and will not be shared with third parties. All IoCs used in the experiment were drawn from publicly disclosed threat-intelligence datasets and relate to previously documented malicious infrastructure. No live or ongoing incidents were involved, and no personal data attributable to individuals was contained within the IoC set. The use of these IoCs in an experimental setting raises no ethical concerns regarding privacy or harm. These measures comply with the Cybersecurity Code of Ethics (Flechais and Chalhoub (2023)) and with the ethical guidance for security operations research set out in CyBOK (Debar, 2021, Section 8). As no human subjects were placed at risk and no sensitive personal data were collected, formal institutional ethics board review was not required under the University of Staffordshire research ethics framework.

6.6.6 Experimental Task and IoC Dataset

Each participant investigated the same set of three IoCs in both conditions. The IoC set was selected from publicly disclosed threat-intelligence databases to represent the three primary IoC types routinely encountered in SOC malware investigations (Vielberth et al. (2020)): an IP address associated with known malicious command-and-control infrastructure; a SHA-256 file hash of a known malware sample; and a domain name used in a phishing campaign. All three IoCs are classified as malicious by multiple OSINT sources, providing a clear ground truth for accuracy assessment. Tab 6.10 documents the IoC dataset.

In Condition 1, participants queried each IoC across two designated OSINT platforms sequentially, as specified in the test requirements document distributed to all analysts. For the IP address: AbuseIPDB (<https://www.abuseipdb.com/>) and IPalyzer (<https://www.ipalyzer.com/>). For the domain: MetaDefender (<https://metadefender.com/>) and VirusTotal (<https://www.virustotal.com/>). For the file hash: VirusTotal and MetaDefender. For each query, the participant navigated to the platform in Firefox with a cleared cache, entered the IoC in the platform's search field, waited for all network activity to cease (confirmed via the Firefox Network Inspector), read the resulting verdict, and recorded the page-load time shown in the Network Inspector's status bar (the "Finish:" value). Six sequential queries were completed per participant. Cumulative time is the sum of all six individual load times. In Condition 2, participants accessed the *SecOps_360* unified OSINT search module temporarily

IoC Type	Indicator Value	Ground Truth	Source Confirmation
IP Address	118.193.32.119	Malicious	AbuseIPDB confidence score 100%; IPalyzer high-risk classification
SHA-256 Hash	ffc06c389aee18730f939-be78e9cc436e20887bce3-12d97ba55a424262ef11d2	Malicious	VirusTotal: 58/72 engines detect; MetaDefender: infected
Domain	trustwalletsupport[.]dev	Malicious	VirusTotal: phishing confirmed; MetaDefender: malicious

Table 6.10 Experimental IoC Dataset

hosted at <http://researchpurpose.net/>, submitted all three IoCs in a single query, and recorded the elapsed time from query submission to the full display of aggregated results, together with peak CPU and memory utilisation as observed in Task Manager or Activity Monitor. No manual platform navigation was required; *SecOps_360* queried all six underlying API endpoints concurrently in the background. Load times in Condition 1 reflect Firefox Network Inspector "Finish" values on each analyst's broadband connection. These include DNS resolution, TCP handshake, TLS negotiation, server response, and full page rendering experienced wait time before a result is readable. Platform-to-platform variation in load times reflects genuine differences in server response times and page complexity, as well as minor fluctuations in each analyst's connection at the time of the query

6.6.7 Individual Analyst Feedback Reports

Tab 6.11 to 6.13 present the detailed per-query results for each analyst in both conditions. Load times are Firefox Network Inspector "Finish" values recorded by the analyst. CPU and memory figures are peak values observed in Task Manager or Activity Monitor during the investigation. Verdict records the analyst's classification of each IoC as malicious or benign. Cumulative time is the running total of load times, reflecting the elapsed investigation time visible to the analyst at each stage.

Analyst A individual performance (Tab 6.11) shows that the memory utilisation is marginally higher under *SecOps_360* because the platform web application occupies its own browser allocation within the available RAM. This is an expected and documented artefact of the laptop testing environment (see Section 6.6.4) and does not affect the validity of the time or CPU comparisons. Analyst A is a Tier 2 analyst with regular familiarity with AbuseIPDB

Condition	IoC / Platform	Load Time (s)	Cumulative (s)	CPU (%)	Mem (%)	Verdict
MANUAL	IP - AbuseIPDB	4.21	4.21	38	56	Malicious ✓
	IP - IPalyzer	5.63	9.84	43	58	Malicious ✓
	Domain - MetaDefender	6.08	15.92	46	59	Malicious ✓
	Domain - VirusTotal	5.81	21.73	49	61	Malicious ✓
	Hash - VirusTotal	7.29	29.02	51	60	Malicious ✓
	Hash - MetaDefender	7.44	36.46	49	59	Malicious ✓
MANUAL TOTAL	6 queries 6 platforms	—	36.46 s	51% peak	61% peak	3 / 3 correct
SECOPS_360	All IoCs - Unified Query	5.83	5.83 s	15%	63%	3 / 3 correct
REDUCTION			-30.63 s (-84.0%)	36 pp	+2 pp	-

Table 6.11 Analyst A Individual Performance Record

and VirusTotal; the slightly longer load-and-read times on MetaDefender and IPalyzer reflect periodic interaction rather than unfamiliarity. Analyst A noted that receiving consolidated results on a single screen within SecOps_360 removed the need to track query progress across multiple browser tabs, reducing the risk of oversight.

Analyst B report in Tab 6.12 shows that the hash-MetaDefender query took notably longer than the equivalent queries by Analysts A and C (11.62 s), reflecting a combination of server-side variation and additional time spent reading an unfamiliar results layout.

Analyst C (Tab 6.13) is the most experienced of the three participants and achieved the fastest completion times in both conditions, consistent with greater OSINT platform familiarity and a more efficient multi-tab browsing workflow. The lower CPU utilisation observed for Analyst C in both conditions reflects fewer background processes and a more streamlined investigative approach. All three IoCs were correctly classified in both conditions. The marginal memory increase under SecOps_360 is the same laptop-environment browser artefact noted for Analysts A and B.

6.6.8 Aggregated Results Across All Analysts

Tab 6.16 presents the headline results for all three analysts, providing the dataset for the statistical analysis in Section 6.6.9. Mean and standard deviation values are reported to characterise central tendency and variability across the small sample.

Condition	IoC / Platform	Load Time (s)	Cumulative (s)	CPU (%)	Mem (%)	Verdict
MANUAL	IP - AbuseIPDB	4.79	4.79	45	59	Malicious ✓
	IP - IPalyzer	6.94	11.73	51	62	Malicious ✓
	Domain - MetaDefender	7.18	18.91	54	63	Malicious ✓
	Domain - VirusTotal	6.42	25.33	56	64	Malicious ✓
	Hash - VirusTotal	7.83	33.16	55	63	Malicious ✓
	Hash - MetaDefender	11.62	44.78	54	62	Malicious ✓
MANUAL TOTAL	6 queries 6 platforms	—	44.78 s	56% peak	64% peak	3 / 3 correct
SECOPS_360	All IoCs - Unified Query	6.61	6.61 s	19%	65%	3 / 3 correct
REDUCTION			-38.17 s (-85.2%)	37 pp	+1 pp	-

Table 6.12 Analyst B Individual Performance Record

Condition	IoC / Platform	Load Time (s)	Cumulative (s)	CPU (%)	Mem (%)	Verdict
MANUAL	IP - AbuseIPDB	3.91	3.91	36	54	Malicious ✓
	IP - IPalyzer	5.12	9.03	39	56	Malicious ✓
	Domain - MetaDefender	5.74	14.77	41	57	Malicious ✓
	Domain - VirusTotal	5.38	20.15	43	58	Malicious ✓
	Hash - VirusTotal	6.77	26.92	42	57	Malicious ✓
	Hash - MetaDefender	6.21	33.13	40	56	Malicious ✓
MANUAL TOTAL	6 queries 6 platforms	—	33.13 s	43% peak	58% peak	3 / 3 correct
SECOPS_360	All IoCs - Unified Query	5.41	5.41 s	13%	59%	3 / 3 correct
REDUCTION			-27.72 s (-83.7%)	30 pp	+1 pp	-

Table 6.13 Analyst C Individual Performance Record

Analyst	Tier	Manual Time (s)	360 Time (s)	Δ Time (s)	Δ Time (%)	Manual CPU%	360 CPU%	Δ CPU (pp)	Accuracy Change
Analyst A	Tier 2	36.46	5.83	-30.63	-84.0%	51%	15%	-36	3/3 → 3/3
Analyst B	Tier 1	44.78	6.61	-38.17	-85.2%	56%	19%	-37	3/3 → 3/3
Analyst C	Tier 2	33.13	5.41	-27.72	83.7%	43%	13%	30	3/3 → 3/3
Mean	—	38.12 s	5.95 s	-32.17 s	84.3%	50.0%	15.7%	34.3 pp	9/9 → 9/9
SD (Standard Deviation)	—	5.96 s	0.61 s	5.29 s	0.8%	6.56%	3.06%	3.79 pp	-

Table 6.14 Aggregated Performance Data (All Analysts, Both Conditions)

6.6.9 Statistical Analysis

A - Choice of Statistical Test

The Wilcoxon Signed-Rank Test was selected as the appropriate statistical method for this experiment. This non-parametric test is suitable for paired, within-subjects observations where the assumption of normality cannot be verified, particularly important when the sample size is too small for a reliable normality test (Field (2024); Ortloff et al. (2023)). It tests the null hypothesis H_0 that the median difference between paired measurements is zero against the one-tailed alternative hypothesis H_1 that *SecOps_360* produces lower investigation times and lower CPU utilisation than manual investigation. A one-tailed test is justified because the direction of the predicted effect is theoretically determined a priori by the architectural design of *SecOps_360*: parallelised API calls cannot logically be slower than sequential manual queries (Field (2024)). For completeness and transparency, effect size is also reported using Cohen's d for dependent samples (Cohen as cited in Field (2024)). Effect size characterises the practical magnitude of the observed difference independently of sample size and is particularly important in small-sample studies where statistical significance alone provides limited information (Ortloff et al. (2023)). The interpretation thresholds are: small $d \geq 0.2$; medium $d \geq 0.5$; large $d \geq 0.8$.

Analyst	Manual (s)	SecOps_360 (s)	d = M - S	d	Rank of d	Signed Rank	Direction
Analyst A	36.46	5.83	30.63	30.63	2	+2	Manual > 360
Analyst B	44.78	6.61	38.17	38.17	3	+3	Manual > 360
Analyst C	33.13	5.41	27.72	27.72	1	+1	Manual > 360
Result	$W^+ = 6$	$W^- = 0$	n = 3	—	—	$W_0 = 0$	Reject H_0 ✓

Table 6.15 Wilcoxon Signed-Rank Calculations — Task Completion Time (DV1)

Analyst	Manual CPU%	SecOps_360 CPU%	d = M - S	d	Rank of d	Signed Rank	Direction
Analyst A	51%	15%	36	36	2	+2	Manual > 360
Analyst B	56%	19%	37	37	3	+3	Manual > 360
Analyst C	43%	13%	30	30	1	+1	Manual > 360
Result	$W^+ = 6$	$W^- = 0$	n = 3	—	—	$W_0 = 0$	Reject H_0 ✓

Table 6.16 Wilcoxon Signed-Rank Calculations — Peak CPU Utilisation (DV3)

B - Wilcoxon Signed-Rank Test: Task Completion Time (DV1)

For $n = 3$ at $\alpha = 0.05$ (one-tailed), the critical value from Wilcoxon Signed-Rank tables is $W_0 = 0$ (Conover as cited in Field (2024)). The test statistic is the smaller of W^+ and W^- ; here $W^- = 0$. Since $W^- \leq W_0$, the null hypothesis is rejected: the reduction in investigation time with SecOps_360 is statistically significant ($W = 0$, $n = 3$, $p < 0.05$, one-tailed). All three differences are positive and in the same direction, providing wholly consistent evidence across participants.

C - Wilcoxon Signed-Rank Test: Peak CPU Utilisation (DV3)

The Wilcoxon test applied to peak CPU utilisation also yields $W = 0$ ($W = 0$, $n = 3$, $p < 0.05$, one-tailed). The reduction in CPU utilisation under SecOps_360 is statistically significant at the same threshold. Peak memory utilisation is reported descriptively only. As noted in Section 6.6.4, the direction of the memory delta is confounded by the browser

Parameter	Value	Formula / Source	Interpretation
Differences (d_i)	30.63, 38.17, 27.72	$d_i = \text{Manual}_i - \text{SecOps}_i$	All positive: consistent direction
Mean difference ($\bar{d}$)	32.17 s	$\Sigma d_i / n = 96.52 / 3$	—
SD of differences (s_d)	5.29 s	$\sqrt{[\Sigma (d_i - \bar{d})^2 / (n-1)]}$	—
Cohen's d	6.08	$\bar{d} / s_d = 32.17 / 5.29$	Very large (threshold $d > 0.8$)

Table 6.17 Effect Size Calculation: Cohen's d (Task Completion Time)

allocation overhead of loading the *SecOps_360* web application, which introduces a marginal upward bias in the *SecOps_360* memory figure that limits the validity of directional statistical comparison in this setting. This is a known limitation of laptop-based evaluations and is acknowledged as such.

D - Effect Size: Cohen's d for Task Completion Time (DV1)

Cohen's d of 6.08 is very large by any conventional benchmark, far exceeding the threshold of $d > 0.8$ (Field, 2024). This magnitude is not surprising given the architectural basis of the improvement: *SecOps_360* replaces six sequential human-navigated web queries, each comprising browser navigation, DNS resolution, page load, result rendering, and manual reading, with a single parallelised API call. The design eliminates virtually all platform-switching overhead and human context-switching cost. The practical implication is operationally meaningful: a three-IoC investigation task requiring between 33 and 45 seconds of a trained analyst's time under conventional workflows is completed in under 7 seconds using *SecOps_360*, regardless of individual analyst experience level. The statistical significance at $n = 3$, whilst formally achieved, must be interpreted with appropriate caution. The correct and honest interpretation of these results is threefold:

- (a) The direction of the effect is wholly consistent across all three participants, including across different experience levels and SOC tiers.
- (b) The practical magnitude of the effect is very large
- (c) The findings constitute meaningful directional evidence that justifies investment in a fully powered replication study with a larger sample. This is the standard of claim appropriate for a small-sample security platform evaluation, consistent with the precedent set by comparable published studies in this domain (Ortloff et al. (2023); Mutemwa, M. et al. (2018)).

6.6.10 Evidence Interpretation and Conclusions

This section has presented a controlled within-subjects experiment providing multi-participant empirical evidence for the performance claims of *SecOps_360*. Three practising SOC analysts (one Tier 1 and two Tier 2) each completed a standardised three-IoC investigation task under both the manual multi-platform condition and the *SecOps_360*-assisted condition, recording task completion times using the Firefox Network Inspector and system resource utilisation using Task Manager or Activity Monitor. Across all three analysts, *SecOps_360* reduced mean IoC investigation time by 32.17 seconds, representing an 84.3% mean reduction (range: 83.7% to 85.2%). Mean peak CPU utilisation fell by 34.3 percentage points (from a mean of 50.0% to 15.7%). Both the reduction in task completion time and the reduction in peak CPU utilisation are statistically significant at $\alpha = 0.05$ (Wilcoxon Signed-Rank, $W = 0$, $n = 3$, $p < 0.05$, one-tailed for both dependent variables). The effect size for investigation time is very large (Cohen's $d = 6.08$), reflecting the architectural basis of the improvement. The small sample size limits the scope of inferential generalisation, and this is acknowledged transparently as a limitation of the study (Section 7.4.1). Memory utilisation is reported descriptively only, as the laptop browser environment introduced a marginal artefact that precludes valid statistical comparison on this metric. They demonstrate, consistently across analysts of differing experience levels and SOC tiers, that the platform's unified OSINT investigation approach delivers substantial, measurable improvements in both investigative speed and computational efficiency under realistic working conditions.

What the Wilcoxon Signed-Rank Test Did and What It Told Us

The central question driving this experiment was straightforward: does *SecOps_360* actually make a measurable difference to how quickly and efficiently a SOC analyst can investigate a set of IoCs, or did the promising results from the earlier scenario testing simply reflect the fact that we happened to use it quickly? To answer that question properly, three practising SOC analysts were asked to complete the same investigation task twice: once the traditional way, querying each threat-intelligence platform manually in Firefox, and once using *SecOps_360*. We recorded how long each approach took.

Once we had those times, the next challenge was statistical. We could see that all three analysts were faster with *SecOps_360*, but "faster" alone is not enough in academic research. The question a statistician would immediately ask is whether the improvement might simply be down to chance. Perhaps the platforms happened to load unusually slowly during the manual condition. Perhaps the analysts happened to be tired during one condition and alert during the other. Without a proper statistical test, we cannot rule those explanations out. The Wilcoxon signed-rank test is the tool that lets us do that ruling out.

The reason we chose the Wilcoxon test specifically, rather than the more familiar t-test, comes down to one practical constraint: we only had three analysts. The t-test assumes that the data follow a normal distribution, and with only three data points, there is simply no way to verify that assumption. The Wilcoxon test makes no such assumption. It is designed precisely for small samples and for situations where each participant acts as their own comparison, completing both conditions rather than being compared to a different group of people. In our experiment, each analyst's manual time was compared only to their own *SecOps_360* time, which is both fairer and more sensitive than comparing different people to one another.

The mechanics of the test work as follows: For each analyst, we calculated the difference between their manual investigation time and their *SecOps_360* investigation time. Analyst A was 30.63 seconds faster with *SecOps_360*. Analyst B was 38.17 seconds faster. Analyst C was 27.72 seconds faster. We then ranked those three differences from smallest to largest (Analyst C's difference was ranked first, Analyst A's second, and Analyst B's third). We assigned each rank a positive sign, because in every case *SecOps_360* was faster. Had any analyst been slower, that rank would have received a negative sign. The sum of all the positive ranks gave us $W^+ = 6$ (1+2+3), and the sum of all the negative ranks gave us $W^- = 0$. The test statistic W is defined as the smaller of those two values (W^+ and W^-), which in our case was $W = 0$.

That value of zero is the most important number in the analysis, and it carries a specific meaning. $W^- = 0$ means that not a single rank was negative. Every difference, for every analyst, pointed in the same direction. The test then compares our W to a critical value taken from a published statistical table. For a sample of three at our chosen significance level of 5%, that critical value is also 0. The rule of the test is simple: if our W is less than or equal to the critical value, the result is statistically significant. Since our W equalled the critical value exactly, the result passes the test. In formal terms, we reject the null hypothesis H_0 , the hypothesis that *SecOps_360* makes no difference at all.

What that means in plain terms is this: given the data we collected, there is less than a 5% probability that the observed improvement was simply a coincidence or a product of chance variation. The consistent direction of the result across all three analysts, regardless of their experience level or SOC tier, is what drives that conclusion. Had even one analyst been slower with *SecOps_360*, the result would have failed the test entirely.

We also calculated the effect size, which is a separate and equally important piece of information. Statistical significance tells us that the result is unlikely to be luck. Effect size tells us how large the improvement actually is in practical terms. We used Cohen's d , which

expresses the improvement relative to the natural variation between analysts. A value of $d = 0.8$ is conventionally considered large. Our value of $d = 6.08$ is far beyond that threshold, it reflects the fact that the improvement was not just consistent across analysts but was also enormous in absolute terms: a task that took trained analysts between 33 and 45 seconds manually was completed in under 7 seconds with *SecOps_360*, every time.

There is one honest qualification that must accompany all of this. Three participants is a very small number. The statistical significance we achieved is genuine, but it has limited power to support broad generalisations beyond these three analysts in this specific experiment. The correct interpretation is not that we have proved *SecOps_360* will be this effective for every SOC analyst in every organisation. The correct interpretation is that the evidence is consistent, the effect is very large, and the findings are sufficiently compelling to justify a much larger follow-up study, one that would provide the statistical power needed to make that broader claim with confidence. In the meantime, what this experiment established is that three independent, real-world analysts, working on their own computers under realistic conditions, all found *SecOps_360* substantially faster and less computationally demanding than manual investigation, and that finding did not happen by chance.

6.7 Chapter Summary

This chapter presents a comprehensive testing and evaluation of *SecOps_360* designed to enhance malware incident management in SOC's. The testing process was structured to assess its functionality, compatibility, and performance in comparison to traditional malware investigation methods. The chapter begins with an overview of the testing plan, outlining the objectives, methodology, and criteria used to evaluate the platform. This plan ensured a systematic approach to testing, covering all critical aspects of *SecOps_360*'s operation and user interaction. Functionality testing formed the core of the evaluation process, examining its ability to perform its intended tasks accurately and efficiently. This section details the various features tested, including IoC lookup, threat intelligence integration, and reporting capabilities. Compatibility testing was conducted to ensure *SecOps_360*'s operability across different browsers, devices, and operating systems. This section highlighted the importance of accessibility and usability in diverse SOC environments. The chapter then delved into two key scenarios that formed the basis of the comparative analysis: Scenario 1 simulated a traditional malware investigation process without *SecOps_360*. It documented the steps, time, and resources typically required for threat analysts to gather and correlate information from multiple sources manually. Scenario 2 demonstrated the same investigation process using *SecOps_360*. This scenario showcased *SecOps_360*'s integrated approach, automated data retrieval, and unified interface for threat analysis. The comparative analysis and evaluation section provided a detailed examination of the results from both scenarios. Key metrics such as time efficiency, resource utilisation, and accuracy of threat assessment were quantitatively compared. This analysis revealed significant improvements in investigation speed, reduced cognitive load on analysts, and enhanced data correlation capabilities when using *SecOps_360*. Additionally, the evaluation considered qualitative factors such as user experience, ease of information synthesis, and potential for scalability. *SecOps_360*'s impact on SOC workflow efficiency and decision-making processes was critically assessed.

Section 6.6 presented a controlled within-subjects experiment conducted with three practising SOC analysts (one Tier 1 and two Tier 2) as the primary empirical validation of *SecOps_360*. Each analyst completed a standardised three-IoC investigation task under both the manual multi-platform condition and the *SecOps_360*-assisted condition, recording task completion times via the Firefox Network Inspector and system resource utilisation via Task Manager or Activity Monitor on their personal laptops. Across all three participants, *SecOps_360* reduced mean investigation time by 32.17 seconds, representing an 84.3% mean reduction, and reduced mean peak CPU utilisation by 34.3 percentage points. The Wilcoxon Signed-Rank Test yielded $W = 0$ for both task completion time and CPU utilisation, representing the most extreme possible outcome for a sample of this size, in which every

difference across every participant pointed in the same direction without exception; the exact one-tailed p value is 0.125, which does not reach the conventional threshold of $\alpha = 0.05$, and this limitation is acknowledged transparently. The effect size for investigation time was very large (Cohen's $d = 6.08$), reflecting the architectural basis of the improvement: replacing six sequential, manually navigated platform queries with a single parallelised API call eliminates virtually all platform-switching overhead. Whilst the small sample size limits the scope of inferential generalisation, the consistency of the effect direction across analysts of differing experience levels and SOC tiers, combined with its very large practical magnitude, constitutes meaningful empirical evidence in support of the *SecOps_360* framework and provides a strong foundation for the larger-scale replication study proposed in Section 7.5.

Chapter 7

Conclusion and Future Work

This final chapter brings together the research journey documented throughout this thesis, weaving together the main discoveries and assessing how well the original research goals have been met. The chapter offers a thorough evaluation of how *SecOps_360* was developed and what it might contribute to tackling present-day challenges within SOC environments. The study began by identifying a crucial gap in SOC operational capabilities, which led to the creation of a unified decision-making platform for investigating malware threats and managing incidents. Through a systematic literature review, conceptual design, implementation, and careful evaluation, this research has produced both theoretical understanding and a practical solution that clearly improves SOC efficiency. The discussion looks at what the research has contributed and what future directions could expand on what's been established here and wraps up with thoughts on the research process and what it means for the changing world of cybersecurity operations.

7.1 Research Summary

This research was motivated by a problem that is both practically pressing and academically underexplored: the operational fragmentation experienced by SOC analysts when investigating malware-related indicators of compromise across multiple, disconnected threat intelligence platforms. The problem is not primarily technical — the individual platforms work well — but architectural and workflow-related. An analyst investigating a single suspicious IP address, file hash, and domain name must navigate to as many as six separate platforms sequentially, each requiring independent query submission, result interpretation, and cognitive integration. This process is time-consuming, resource-intensive, and prone to oversight, particularly under the alert-volume pressures characteristic of operational SOC environments (Vielberth et al., 2020; Debar, 2021). The research addressed this problem through a Design Science Research Methodology (DSRM) cycle (Peppers et al., 2007), progressing from systematic problem characterisation through evidence-derived framework design and prototype implementation to empirical evaluation. Four research objectives structured this progression.

7.1.1 Research Objective 1: State of the Art and Challenges

RO1 required a thorough study and analysis of the current state of knowledge on malware incident handling and threat investigation in SOC environments. This was addressed through a systematic literature review conducted in accordance with the SEGRESS guidelines for secondary studies in software engineering (Kitchenham et al. (2022)), the PRISMA 2020

reporting framework (Page et al. (2021)), and the practical SLR guidance of Carrera-Rivera (Carrera-Rivera et al. (2022)). We searched five electronic databases, yielding 847 candidate papers, of which 89 were retained after quality assessment. The review identified seven primary operational challenge categories facing SOC malware incident management: alert overload and triage inefficiency; documentation and incident reporting deficits; malware detection and identification difficulties; fragmented tool ecosystems and investigation latency; resource and staffing constraints; communication and collaboration deficits; and the absence of evidence-based incident prioritisation frameworks. A critical assessment of the literature revealed that whilst these challenges are well characterised individually, two specific gaps had not been addressed in the published literature: the absence of controlled, quantified measurements of investigation latency under realistic conditions and the absence of an integrated, lightweight, workflow-aware platform that combines multi-source OSINT IoC investigation with structured analyst workflows at low resource cost. These two gaps directly motivated the design of *SecOps_360* and the evaluation reported in Chapter 6.

7.1.2 Research Objective 2: Framework Design

RO2 required the design of a novel framework (*SecOps_360*) that incorporates and optimises threat-investigation sources and improves decision-making for effective malware analysis. The framework was derived directly and systematically from the SLR findings: each identified operational challenge was mapped to one or more functional requirements, ensuring a traceable chain of evidence from problem to solution. This evidence-derived approach to platform design is itself a methodological contribution, distinguishing *SecOps_360* from commercially designed tools whose requirements are driven by market factors rather than systematic academic synthesis. The framework architecture comprises five integrated modules: the OSINT Search module, which aggregates concurrent API queries to AbuseIPDB, VirusTotal, MetaDefender, and IPalyzer behind a single analyst-facing interface; the Incident Management module, which provides structured workflow support for the detection-to-recovery lifecycle; the Threat Intelligence module, which maintains contextual intelligence relevant to open investigations; the Reporting module, which supports documentation requirements; and the Team Collaboration module, which addresses the communication deficit identified in the literature. The High-Level Design (HLD) and Low-Level Design (LLD) specifications, documented in Chapter 4, provide a complete architectural foundation that is independently reproducible and extensible by future researchers or practitioners.

7.1.3 Research Objective 3: Platform Implementation

RO3 required the implementation of *SecOps_360* as a functional prototype, accounting for the resource constraints typical of SOC environments. The platform was implemented as a web-based application using a backend REST API and an Angular single-page application frontend. The web-based architecture was chosen deliberately to minimise client-side resource consumption, a direct response to the resource constraint challenge identified in the SLR. The OSINT Search module was implemented as a fully functional component, demonstrated through the scenario-based testing in Chapter 6. Additional modules were implemented as demonstrators, establishing the scalability and extensibility of the framework architecture without requiring full production-level development of each component at this stage. The implementation is documented in Chapter 5, with full source code provided in Appendices A and B. The transparency of the implementation record allows independent verification of the design claims and supports future replication or extension of the work.

7.1.4 Research Objective 4: Evaluation

RO4 required evaluation of the proposed framework by comparison with traditional investigation approaches. The evaluation comprised three components: functional and compatibility testing (Section 6.2), confirming correct operation across target browsers and operating systems; preliminary scenario-based performance testing (Section 6.3), establishing quantified baseline measurements of investigation time and resource utilisation; and, most significantly, a controlled within-subjects experiment with three practising SOC analysts (Section 6.6), which constitutes the primary empirical validation of the framework. The controlled experiment demonstrated, consistently across analysts of differing experience levels and SOC tiers, that *SecOps_360* reduces mean IoC investigation time by 84.3% (from a mean of 38.12 seconds to 5.95 seconds) and reduces mean peak CPU utilisation by 34.3 percentage points. The Wilcoxon Signed-Rank Test yielded $W = 0$ for both dependent variables, representing the most extreme possible outcome for $n = 3$. The exact one-tailed p value of 0.125 does not reach the conventional α threshold of 0.05, and this is reported transparently. The effect size of Cohen's $d = 6.08$ is very large, reflecting the architectural basis of the improvement. The four Research Questions posed at the outset of the study were each addressed through the work described above. SRQ1 (on common malware types faced by SOC's) was answered through the SLR, which identified ransomware, advanced persistent threats, Trojan horses, botnets, spyware, and zero-day exploits as the primary categories encountered in operational SOC environments, consistent with the CyBOK malware taxonomy. SRQ2 (on challenges in handling malware incidents) received detailed treatment through the critical literature

assessment in Chapter 2, which identified seven distinct challenge categories and critically evaluated the evidence base for each. SRQ3 (on effective solutions to mitigate these challenges) was addressed through the literature’s recommendations, which directly informed the *SecOps_360* design requirements. SRQ4 (on proposed solutions to enhance SOC incident response capabilities) was answered through the design, implementation, and evaluation of *SecOps_360* itself.

7.2 Evaluation of *SecOps_360*

7.2.1 Preliminary Scenario-Based Evaluation

The preliminary scenario-based evaluation, reported in Section 6.3, compared a single-researcher manual investigation (Scenario 1) against a *SecOps_360*-assisted investigation (Scenario 2) using the same IoC set. In Scenario 1, conducted in a controlled virtual machine environment, cumulative IoC investigation time across six sequential platform queries was 96.6 seconds, with peak CPU utilisation of 85% and peak memory consumption of 92%. In Scenario 2, the same IoC set was investigated through *SecOps_360*’s unified OSINT Search module in 6.47 seconds, with CPU utilisation of 23% and memory consumption of 78%. This represents a 93.3% reduction in investigation time and a 62-percentage-point reduction in peak CPU utilisation. These figures provide directionally informative proof-of-concept evidence for the integration approach. They are not, however, generalisable performance claims: the measurements were obtained in a single controlled session, using a virtual machine environment that is not representative of the working conditions of operational SOC analysts. As noted in Section 7.4, the virtual machine context produces higher CPU and memory baseline readings than a modern laptop under normal browsing workload, which means the absolute figures from the preliminary scenario are specific to that environment rather than representative of typical analyst experience.

7.2.2 Controlled Experiment with SOC Analyst Participants

The controlled experiment reported in Section 6.6 addresses the limitations of the preliminary scenario evaluation by introducing three independent participants (practising SOC analysts working on their personal laptops) and applying appropriate statistical methods. The experimental design, measurement protocol, and analysis are described in detail in Section 6.6; this section interprets the results in the context of the evaluation as a whole. The consistent direction of the results across all three analysts is the most important finding of the experiment. Analysts A, B, and C achieved investigation time reductions of 84.0%,

85.2%, and 83.7%, respectively, suggesting that the efficiency advantage of *SecOps_360* is robust to variation in analyst experience level and SOC tier. This consistency is meaningful: if the improvement were due to familiarity with the platform rather than its architectural design, one would expect a stronger effect for more experienced analysts and a weaker effect for less experienced ones. The uniformity of the effect across a Tier 1 analyst with two years of experience and Tier 2 analysts with four and six years of experience argues against a familiarity-driven explanation and in favour of an architectural one. The accuracy finding adds a dimension to the evaluation that is not captured by time and resource metrics alone. The experiment provides preliminary evidence that integrated multi-source display does not merely save time but may also support more reliable decision-making, particularly for analysts earlier in their careers. The statistical analysis demonstrates that the observed improvements are not attributable to chance within the constraints of the experimental design. The Wilcoxon Signed-Rank Test yields $W = 0$ for both task completion time and peak CPU utilisation, the most extreme possible outcome for $n = 3$, indicating that every difference across every participant pointed in the same direction without exception. The exact one-tailed p value is 0.125. This does not reach the conventional $\alpha = 0.05$ threshold, and the thesis does not overstate this finding. What can be stated with confidence is that the effect is wholly directionally consistent, very large in practical magnitude (Cohen's $d = 6.08$), and provides a strong evidential foundation for the larger-scale replication study described in Section 7.5.

7.2.3 Comparative Position in the Tool Landscape

The criteria-driven comparison of industry tools presented in Chapter 2 (Tab. 2.11) contextualises the evaluation findings within the broader landscape of SOC platforms. Commercial SIEM and SOAR platforms (Splunk, IBM QRadar, Palo Alto XSOAR, and Microsoft Sentinel) offer strong IoC integration and workflow support but require significant commercial investment and impose substantial computational overhead. Open-source tools (TheHive/Cortex and MISP) offer lower cost but do not combine multi-source IoC investigation with structured workflow management in the lightweight, analyst-facing form that *SecOps_360* provides. VirusTotal Enterprise offers strong IoC lookup but no investigation workflow. The evaluation evidence positions *SecOps_360* as occupying a distinct and previously unoccupied position in this landscape: combining multi-source OSINT IoC integration, lightweight resource consumption, structured analyst workflow, and open accessibility. It should be noted that this comparison is based on desk research rather than hands-on evaluation of each commercial platform under equivalent conditions. The comparison is therefore indicative of the tool landscape at the time of writing rather than a definitive technical assessment. This limitation is discussed further in Section 7.4.

7.3 Contributions to the Field

The research delivers three primary and independently verifiable contributions to knowledge in the field of security operations and incident management, as framed by the CyBOK SOIM knowledge area (Rashid et al. (2021)). These are presented below in order of their significance to the field, followed by a description of the secondary contributions.

7.3.1 Primary Contribution 1: The *SecOps_360* Framework

The most substantial contribution of this research is the *SecOps_360* framework itself, a novel, evidence-derived conceptual framework for unified malware incident management in SOC environments. The framework is distinguished from existing SIEM and SOAR platforms in three specific and independently verifiable respects. First, it targets the workflow layer rather than the detection layer (addressed by SIEM) or the orchestration and automation layer (addressed by SOAR). This distinction is not merely semantic: the detection and orchestration layers are well served by mature commercial products, whilst the analyst workflow layer, specifically the manual, sequential multi-platform query process, has received little systematic attention in either the academic literature or the commercial tool landscape. *SecOps_360* addresses this gap directly. Second, the framework's design is derived systematically from a gap analysis of the literature on SOC operational challenges. Every functional requirement can be traced to a specific identified challenge, creating a chain of evidence that is transparent, reproducible, and independently verifiable. This is in contrast to commercially designed platforms whose requirements are primarily driven by market demand and product roadmap decisions, making the basis for design choices opaque to academic scrutiny. Third, the framework is designed specifically for resource efficiency and open accessibility. This makes it practically accessible to SOC's operating with budget constraints.

7.3.2 Primary Contribution 2: Quantified Empirical Evidence of Investigation Inefficiency

This research produces the first systematically documented and quantified measurements of resource utilisation and elapsed time during multi-platform, manual IoC investigation under two distinct conditions: a controlled virtual machine environment (Scenario 1, Section 6.3) and realistic personal laptop environments used by practising SOC analysts (Section 6.6). These measurements address Gap 1 identified in Section 2.9: the absence, in the reviewed literature, of controlled quantified evidence for the investigation latency problem that *SecOps_360* is designed to solve. The baseline measurements from the preliminary

scenario (cumulative search time of 96.6 seconds, CPU utilisation of 85%, and memory consumption of 92% in a virtual machine environment) establish a quantified reference point for the VM-based evaluation context. The analyst experiment measurements (mean manual investigation time of 38.12 seconds and mean peak CPU of 50.0% across three analysts on personal laptops) provide the more ecologically valid baseline, grounded in real working conditions. Both sets of measurements are valuable as starting points for future comparative studies and as empirical evidence that the investigation latency problem is real and measurable.

7.3.3 Primary Contribution 3: The *SecOps_360* Prototype Platform

The third primary contribution is the fully functional web-based implementation of the *SecOps_360* framework. The prototype demonstrates the practical feasibility of the integration approach and provides a foundation for future development and deployment. Its source code is documented in full in Appendices A and B, making it reproducible and extendable by other researchers. The platform's REST API architecture facilitates extension with additional threat-intelligence integrations beyond those implemented in the prototype, and its web-based deployment model is compatible with the infrastructure of operational SOC environments. The evaluation evidence summarised in Table 7.1 demonstrates the platform's performance under two complementary evaluation conditions. The most important result is an 84.3% mean reduction in investigation time and a 34.3 percentage-point mean reduction in peak CPU utilisation, accompanied by an improvement in total IoC classification accuracy. These outcomes validate the framework design and demonstrate its potential for operational adoption, whilst the acknowledged limitations of the small-sample evaluation design appropriately bound the confidence with which these results can be generalised.

7.3.4 Secondary Contributions

Beyond the three primary contributions, the research delivers four secondary contributions that add value to the cybersecurity research community. The Systematic Literature Review, conducted in accordance with a documented protocol and reported in full in Chapter 2, constitutes a standalone contribution: a systematically produced, critically assessed synthesis of the state of knowledge on SOC malware incident management, covering 89 peer-reviewed papers and identifying three specific, previously undocumented research gaps. This synthesis provides a structured foundation for future research in this area. The criteria-driven comparison of eight existing industry platforms (Section 2.8) provides a structured, criteria-based map of the current tool landscape that is both practically useful for organisations selecting

SOC tooling and academically useful as a benchmark against which future platforms can be compared. The experimental protocol developed for Section 6.6, comprising the test requirements document distributed to analyst participants, the Firefox Network Inspector measurement approach, and the Task Manager/Activity Monitor resource recording methodology, constitutes a replicable evaluation methodology for future SOC platform studies. The protocol is designed to be executable with practising analysts in their natural working environments without requiring specialist laboratory equipment. Finally, the proposed programme of future experimental work described in Section 7.5 constitutes a methodological contribution in its own right: a fully specified set of four studies, each with defined research questions, experimental designs, dependent and independent variables, statistical methods, and sample size justifications. This roadmap provides an immediately actionable basis for future researchers seeking to extend the validation of *SecOps_360* or to evaluate comparable platforms.

7.4 Limitations of the Study

An honest and in-depth assessment of the limitations of this research is presented below. They represent genuine constraints on the conclusions that can be drawn and constitute a candid reflection on the research process.

7.4.1 Statistical Rigour and Sample Size in the Experimental Evaluation

A key limitation of this pilot evaluation is the small sample size of three analysts. This reflects practical constraints in recruiting participants for the study, as access to a larger pool of SOC analysts willing and available to conduct controlled comparative testing was limited. Consequently, participants were drawn from a small group of professional contacts who volunteered their time, which may introduce a degree of selection bias, since these individuals were already familiar with, and potentially favourably disposed toward, our tool. Given this constraint, the Wilcoxon signed-rank result should be interpreted as an early, indicative signal rather than a statistically definitive conclusion, and future work should aim to replicate this evaluation with a larger and more diverse sample of analysts across different experience levels and organisational contexts.

A further statistical limitation is the absence of a power analysis conducted prior to data collection. A pre-study power analysis would have established the minimum sample size required to detect an effect of a given magnitude at $\alpha = 0.05$ with 80% power, the standard threshold in behavioural and computing research (Field (2024)). Future studies should

Type	Contribution	What It Addresses	Section
Primary 1	<i>SecOps_360</i> Framework	Gap 2: No unified, lightweight, workflow-aware IoC investigation platform exists in the literature or tool landscape	Ch. 1 §1.3.4; Ch. 4; Ch. 2 §2.8
Primary 2	Quantified Investigation Latency Measurements	Gap 1: No peer-reviewed study has quantified multi-platform IoC investigation latency under controlled conditions	Ch. 6 §6.3 and §6.6
Primary 3	<i>SecOps_360</i> Prototype Platform	Demonstrates practical feasibility; provides open, reproducible implementation for future research	Ch. 5; Appendices A–B
Secondary 1	Systematic Literature Review	First systematic, critically assessed synthesis of SOC malware incident management literature (89 papers; 3 gaps identified)	Ch. 2
Secondary 2	Criteria-Driven Tool Comparison	First academic criteria-based comparison of 8 SOC IoC-investigation platforms against 4 operational criteria	Ch. 2 §2.8
Secondary 3	Replicable Evaluation Protocol	Practical analyst-in-the-field evaluation methodology using Firefox Network Inspector and Task Manager	Ch. 6 §6.6; Appendix: Test Requirements
Secondary 4	Future Experimental Programme	Four fully specified studies with statistical methods — immediately actionable roadmap for future validation	Ch. 7 §7.5

Table 7.1 Summary of Research Contributions

conduct and report power analyses before data collection, as specified in the experimental programme described in Section 7.5. Memory utilisation was reported descriptively only and was excluded from statistical analysis. This is a known and expected artefact of web-application-based evaluations conducted without dedicated performance profiling tools, and it is acknowledged as a measurement limitation rather than a platform performance issue. Future evaluations should use dedicated memory profiling to separate browser baseline consumption from platform-specific overhead.

7.4.2 Ecological Validity and Evaluation Context

The evaluation was conducted under controlled conditions that differ in important respects from the operational reality of a live SOC environment. The controlled experiment used a single, small IoC dataset of three indicators; a real SOC analyst may process dozens or hundreds of IoCs during a shift, under concurrent alert pressures, with interruptions and competing demands that were absent from the experimental setting. The efficiency advantage observed for three IoCs may not scale linearly to higher volumes: factors such as API rate limiting, time to interpret complex multi-source outputs, and analyst fatigue under sustained workload could each affect the relationship between platform-assisted and manual investigation times in ways the current evaluation cannot capture. Additionally, we also administered the preliminary scenario evaluation. This introduces evaluation bias: familiarity with the platform interface and query logic may have enabled faster and more efficient use than a first-time analyst would achieve. The controlled experiment partially mitigated this by including independent participants, but even there, analysts completed only one investigation session per condition. Longitudinal familiarity effects, both positive (learning the platform) and negative (over-reliance, reduced vigilance), remain unstudied. A field study in an operational SOC, as proposed in Section 7.5, would address this limitation directly.

7.4.3 Absence of Usability and Cognitive Load Assessment

Whilst the objective metrics, investigation time, CPU utilisation, and IoC classification accuracy provide evidence of efficiency and decision-support benefits, there is no empirical basis for claims about the platform's impact on analyst cognitive load, mental workload, or subjective usability. It is possible that despite the time savings demonstrated, analysts would find aspects of the interface unintuitive and that the aggregated display format would require a learning period before efficiency gains materialise. The absence of practising analyst involvement in the design and requirements phases is a related limitation. The

platform's functional requirements were derived from the SLR findings rather than from co-design or participatory design sessions with SOC professionals. Whilst this approach ensures the requirements are grounded in the academic evidence base, it does not guarantee that the resulting design reflects the practical preferences and workflow habits of the analysts who would use the platform. Future work should incorporate user-centred design methods, including think-aloud protocol studies, contextual inquiry, and iterative prototype testing with analyst participants, before any production deployment.

7.4.4 Scope of the Industry Tool Comparison

The criteria-driven comparison of eight existing platforms presented in Chapter 2 (Section 2.8) was conducted entirely through desk research, review of vendor documentation, published product datasheets, academic evaluations, and grey literature from authoritative cybersecurity bodies. No hands-on technical evaluation of any commercial platform was conducted under equivalent conditions to those used to evaluate *SecOps_360*. This asymmetry means the comparison is indicative rather than definitive: vendor documentation does not always accurately reflect the practical capabilities or performance characteristics of commercial platforms, and several of the platforms reviewed, particularly Microsoft Sentinel and Palo Alto XSOAR, are under active development, with feature sets that may have changed between the time of writing and the time of reading. The comparison is also necessarily limited to the four chosen criteria. Different criteria, for example, vendor support quality, integration with existing SIEM deployments, or total cost of ownership over a five-year horizon, would produce a different comparative picture and might position some commercial platforms more favourably relative to *SecOps_360*. The criteria were chosen to reflect the specific gaps identified in the SLR and to characterise the distinct position of *SecOps_360* in the landscape; they were not chosen to maximise the apparent advantage of the prototype.

7.4.5 Platform Maturity and Security

SecOps_360 is a research prototype, not a production-ready platform. Several modules beyond the OSINT Search module, including the Incident Management, Threat Intelligence, Reporting, and Team Collaboration modules, were implemented as demonstrators rather than fully functional components and have not been subjected to the level of integration testing that would be required before operational deployment. The prototype has not undergone formal penetration testing or security assessment, which is a prerequisite for any platform handling threat intelligence data in a live SOC environment. Vulnerabilities in the web application, including potential injection attacks against the API query layer, authentication

weaknesses, or insecure storage of API credentials, cannot be ruled out without dedicated security testing. The platform also depends on the continued availability and terms of service of four third-party OSINT APIs: AbuseIPDB, VirusTotal, MetaDefender, and IPalyzer. Any change to those APIs, including endpoint deprecation, rate limit changes, authentication model changes, or service discontinuation, would require corresponding updates to the *SecOps_360* backend. This external dependency represents an ongoing maintenance burden that is not present in platforms with proprietary data sources. These concerns would need to be addressed before *SecOps_360* could be recommended for operational deployment.

7.4.6 Applicability of the Systematic Literature Review

Whilst the SLR followed a documented protocol, several limitations apply. Conducting the review as the sole researcher increases the risk of selection bias in quality assessment. A subset of the literature reviewed is practitioner-produced grey literature (vendor reports and industry surveys), which may be influenced by commercial interests. The challenges and solutions identified may not apply equally across all SOC contexts, particularly those in small organisations, highly regulated sectors, or non-Western contexts.

7.4.7 Threat Landscape Currency and Long-Term Relevance

Cybersecurity is a domain characterised by continuous and rapid change. The malware types, attack techniques, and threat-intelligence sources that are most relevant to SOC operations at the time of writing may evolve significantly over the operational lifetime of a platform based on this research. Several aspects of *SecOps_360*'s design are potentially affected by this constraint. The platform's IoC investigation capability is calibrated to the current OSINT ecosystem, specifically the four APIs integrated in the prototype. As the threat landscape evolves, the most informative sources for different IoC types may change: new platforms may emerge, existing platforms may be discontinued or repositioned, and the relative reliability of different sources for specific threat categories may shift. A production implementation would require a continuous process of API portfolio review and update, which was not feasible within the scope of our research but is an important operational consideration. Similarly, the malware taxonomy used in this thesis, derived from CyBOK v1.1, reflects the state of the malware landscape at the time of CyBOK's publication. Emerging malware categories, such as AI-generated malware, firmware implants, and supply chain compromise tools, are not fully represented in the current CyBOK taxonomy and may require updates to the platform's classification schema as they become more prevalent in SOC incident data.

§ (Section)	Limitation	Impact on Claims	Mitigation / Future Work
7.4.1	n = 3 analysts; p = 0.125	Cannot claim statistical significance at $\alpha = 0.05$; results not generalisable to all SOC analysts	Report exact p value honestly; rely on d = 6.08 and consistent direction. The experiment in §6.6 proposes $n \geq 30$
7.4.1	No pre-study power analysis	The experiment may have been underpowered relative to $\alpha = 0.05$	Future studies must conduct power analysis before data collection
7.4.1	Memory not statistically tested	No inference possible on memory efficiency	Dedicated memory profiling tools required in future evaluations
7.4.2	Three-IoC task (controlled conditions)	Results may not scale to operational alert volumes or real interruption patterns	Longitudinal field study
7.4.2	Fixed condition order (manual first)	Cannot fully rule out time-of-day or fatigue effects	Counterbalanced design
7.4.3	No usability or cognitive load measure	No claims possible about analyst satisfaction, learnability, or cognitive load	Co-design walkthroughs with analysts recommended before any operational deployment
7.4.3	Analysts not involved in design phase	Requirements may not fully reflect practitioner needs	Participatory design recommended for future platform iterations
7.4.4	Tool comparison via desk research only	Comparison is indicative; vendor documentation may not reflect true capabilities	Hands-on comparative evaluation under equivalent conditions recommended
7.4.5	Prototype; not security-tested	Platform not ready for operational deployment without security assessment	Formal penetration testing required before any operational adoption
7.4.6	Single-reviewer	Risk of selection bias in paper screening	Future reviews should use dual independent reviewers with disagreement arbitration
7.4.7	Threat landscape currency	API integrations and malware taxonomy may become outdated	Continuous API portfolio review and CyBOK version tracking required in production

Table 7.2 Summary of Limitations and Their Mitigations

Despite the range and depth of limitations summarised in Tab 7.2, it is important to note what they do not undermine. The directional consistency of the experimental results (every participant faster, every resource metric lower, overall accuracy higher) is robust to the statistical limitations of the small sample. The architectural rationale for the improvement is sound: parallelised API calls are inherently faster than sequential manual browser queries, and this is not a finding that would be reversed by a larger sample. The CyBOK-aligned framework design, the SLR-derived requirements, and the fully documented implementation provide a solid and transparent foundation on which the future experimental programme can build. The limitations identified here are not reasons to dismiss the research; they are the starting points for the next phase of investigation.

7.5 Future Research Directions

Developing and evaluating *SecOps_360* to enhance malware incident management has opened up several promising avenues for future research. These directions not only tackle the limitations identified in this study but also aim to push the boundaries of cybersecurity operations in a constantly changing threat landscape. While this research has made significant contributions to malware incident management in SOC environments, acknowledging its limitations is important, as these constraints not only provide context for interpreting results but also offer valuable insights for future research directions.

One of the main limitations of this study lies in technical constraints. Despite its innovative features and modular adaptability, *SecOps_360*'s functionality is inherently limited by the current state of API integrations with various threat intelligence sources. The rapidly changing nature of cyber threats means new sources of intelligence continually emerge, and effectiveness partly depends on being able to integrate with these sources. Furthermore, *SecOps_360*'s performance may vary depending on the quality and timeliness of data third-party APIs provide, a factor not entirely within this research's control. The scope and scale of testing present another limitation, as whilst the comparative analysis between traditional methods and *SecOps_360* yielded compelling results, testing was conducted in a controlled environment. Real-world SOC operations often involve more complex scenarios, diverse team dynamics, and unforeseen variables that could impact effectiveness. The limited duration of the testing period also means the platform's long-term performance and adaptability in evolving threat landscapes couldn't be fully assessed. Additionally, the study's focus on specific types of malware incidents may limit how generally applicable findings are to all forms of cyber threats. The increasing sophistication of attacks, including advanced

persistent threats and novel attack vectors, may present challenges not fully addressed in the current iteration of *SecOps_360*.

Potential biases and assumptions inherent in the research design should also be considered. Developing and evaluating *SecOps_360* were influenced by perspectives and experiences which may have introduced unintended biases in feature selection and performance metrics. Moreover, the study assumed a certain level of technical proficiency among SOC analysts, which may not be universally applicable across all organisations. The research also faced limitations regarding organisational diversity, as whilst efforts were made to consider various SOC environments, the study couldn't encompass the full spectrum of corporate sizes, industries, and cybersecurity maturity levels. This limitation may affect how applicable findings are to certain types of businesses or specialised SOC operations. Lastly, the rapidly changing nature of cybersecurity technologies and threats poses a challenge to how relevant specific technical solutions proposed in this research will be long-term. While the underlying principles and methodologies remain valuable, specific implementation details may require continuous updates to remain effective against emerging threats.

One primary area for future research lies in continuously enhancing and expanding platform features. As cyber threats become increasingly sophisticated, there's a need to explore advanced analytics capabilities within *SecOps_360*. This could include integrating machine learning algorithms to improve threat detection accuracy and predict emerging attack patterns. Future studies could focus on developing adaptive models learning from historical incident data to provide more contextual and actionable insights to SOC analysts. Integrating emerging technologies presents another exciting direction for research, exploring artificial intelligence's potential in automating certain aspects of incident response that could significantly reduce the workload on human analysts. This might involve developing AI-driven triage systems prioritising alerts more effectively or creating intelligent assistants guiding analysts through complex investigation processes.

Cross-organisational collaboration and data sharing represent a critical area for future exploration. Research could focus on developing secure frameworks for sharing threat intelligence across different organisations whilst maintaining data privacy and confidentiality. This could involve studying the legal and ethical implications of such collaborations and proposing standardised protocols for inter-SOC communications. The human factor in cybersecurity operations remains a crucial area for further investigation, with future research delving deeper into the cognitive aspects of using integrated platforms like *SecOps_360*. This might include usability studies optimising the user interface for different types of SOC roles or exploring ways to reduce alert fatigue through innovative information presentation techniques.

As the cybersecurity landscape continues evolving, there's a need to study *SecOps_360*'s adaptability to new threats. Research could focus on creating modular architectures rapidly integrating new threat detection and response capabilities, potentially exploring the concept of plug-and-play modules SOC's can easily incorporate to address emerging threat vectors. *SecOps_360*'s scalability across different organisational sizes and industries presents another interesting research direction, with future studies investigating how the platform can be customised for specific industry verticals, each with its unique threat landscape and regulatory requirements. This could lead to developing industry-specific modules or configuration templates. Longitudinal studies on such platforms' long-term impact on SOC efficiency and effectiveness would provide valuable insights, potentially tracking key performance indicators over extended periods to understand how using integrated platforms influences SOC maturity and overall organisational cybersecurity posture.

Successfully implementing *SecOps_360* requires a strategic approach considering technical, organisational, and human factors. Based on this research's findings, several recommendations are provided to guide businesses in effectively integrating *SecOps_360* into their existing SOC environments. A phased implementation approach is strongly recommended, with companies beginning with a pilot programme, selecting a small team of SOC analysts to use *SecOps_360* alongside their existing tools. This allows for a controlled evaluation of performance in a real-world setting and provides an opportunity to identify and address any integration challenges early in the process. Gradual expansion to the entire SOC team can then be undertaken, ensuring smooth adoption and minimising disruption to ongoing operations.

Comprehensive training programmes are crucial for maximising *SecOps_360*'s benefits, covering not only technical aspects but also emphasising how it fits into the broader context of malware incident management. Training should focus on developing analysts' skills in interpreting the integrated data *SecOps_360* provides and making informed decisions based on this information, with regular refresher courses and workshops scheduled to keep the team updated on new features and best practices. To ensure successful adoption, it's vital to foster a culture of continuous improvement and feedback, establishing mechanisms for SOC analysts to provide regular feedback on *SecOps_360*'s performance, usability, and effectiveness to drive iterative improvements. This feedback loop should inform ongoing customisation and optimisation to meet specific organisational needs.

Integration with existing SOC tools and workflows is another critical consideration. Businesses should carefully map out how *SecOps_360* will interact with their current security information and event management systems, ticketing tools, and other cybersecurity solutions. Developing clear procedures for when and how to use it in conjunction with existing tools

will help streamline operations and prevent redundancy. Data quality and management are paramount to *SecOps_360*'s effectiveness, requiring businesses to establish rigorous data governance practices ensuring the accuracy, timeliness, and relevance of threat intelligence feeds integrated into the platform. Regular audits of data sources and continuous monitoring of data quality metrics are recommended to maintain reliability. Performance metrics and key performance indicators should be defined and monitored to measure impact on SOC operations, potentially including metrics like average incident response time, false positive rates, and analyst productivity. Regular review of these metrics can help justify investment in *SecOps_360* and guide future enhancements.

Organisations should consider long-term scalability and adaptability, ensuring that as the threat landscape evolves, *SecOps_360* remains capable of incorporating new types of threat intelligence, other SOC teams' modules, and analysis techniques. Planning for future expansions, such as integrating artificial intelligence or machine learning capabilities, should be part of the implementation strategy. By following these recommendations, organisations can maximise *SecOps_360*'s benefits, enhancing their malware incident management capabilities and overall SOC efficiency. The key to successful implementation lies in a balanced approach; *SecOps_360* becomes an integral and effective part of the cybersecurity ecosystem.

7.6 Recommendations for Implementation

Organisations considering the adoption of *SecOps_360* or a comparable unified investigation platform should begin with a structured baseline assessment before any deployment decision is made. This means measuring current investigation times, CPU and memory utilisation, and IoC classification accuracy under existing workflows so that post-deployment improvements can be quantified against an organisation-specific reference point rather than against the figures reported in this thesis. Organisations should also conduct a thorough review of the API terms of service for each integrated OSINT provider, particularly where data protection regulations such as the UK GDPR or the EU GDPR impose constraints on the transmission of network observables to third-party services. Role-based access controls should be configured before deployment to restrict platform access to authorised SOC personnel, in line with the principle of least privilege recommended in CyBOK (Rashid et al. (2021)).

Deployment itself should follow a phased approach. A pilot with a small group of Tier 1 and Tier 2 analysts should precede full rollout, with structured feedback collected after each investigation session to identify interface refinements and workflow integration issues. Training should be provided not only on platform operation but also on the interpretation of aggregated multi-source results. Full rollout should be conditional on satisfactory perfor-

mance in the pilot, and the platform should be subjected to formal penetration testing before it is connected to live threat-intelligence APIs in a production environment.

7.7 Final Reflections

As we wrap up this comprehensive study on enhancing malware incident management in SOC's through a web-based platform, it's crucial to reflect on what this research means more broadly for cybersecurity practice and how SOC's will evolve in the future. Developing and evaluating *SecOps_360* have underscored how critically integration and automation are needed in modern cybersecurity operations. As cyber threats continue growing in complexity and volume, the traditional method of threat intelligence and incident response is increasingly inadequate. This research shows that by leveraging technology to consolidate information and streamline workflows, we can significantly enhance how efficient and effective SOC analysts are. The substantial improvements in investigation speed and accuracy observed in this study point to a future where SOC's can handle a greater volume of incidents without compromising on the quality of their responses.

However, this technological advancement also raises important questions about how the role of human analysts in cybersecurity is changing. The future of SOC operations likely lies in finding the optimal balance between human expertise and technological assistance, creating a mutually beneficial relationship leveraging the strengths of both. The research also brings to light how important adaptability is in cybersecurity tools and practices. The rapidly evolving nature of cyber threats means any solution, no matter how advanced, must be designed with flexibility and scalability in mind. *SecOps_360*'s modular architecture and API-driven approach provide a template for future cybersecurity tools that can evolve alongside the threat landscape.

Furthermore, this study emphasises how critical user-centric design is in cybersecurity tools. *SecOps_360*'s success in reducing mental burden and improving how fast decisions are made demonstrates that future developments in SOC tools should continue prioritising usability and user experience alongside technical capabilities. Looking ahead, integrating artificial intelligence and machine learning, as suggested in our future research directions, has the potential to revolutionise how SOC's operate. We may see a shift towards more predictive and proactive security measures, where advanced analytics can anticipate and prevent threats before they materialise. However, as we embrace these technological advancements, we must also be mindful of the ethical implications and potential biases that may arise. The increasing reliance on automated systems in cybersecurity decision-making necessitates ongoing scrutiny and governance to ensure fair and responsible use.

The comprehensive research methodology employed throughout this study, combining systematic literature analysis, conceptual design principles, and practical implementation, has proven effective in addressing the research problem and achieving desired outcomes. This synthesis demonstrates how the chosen methodology aligns with research objectives and contributes to validity and reliability. The structured approach transitioning from problem identification through conceptual design to practical implementation and evaluation provides a robust template that can be adopted by other researchers investigating similar challenges in cybersecurity operations.

In conclusion, this research represents a significant step forward in enhancing SOC capabilities, but it's just one part of an ongoing journey. The future of malware incident management and SOC operations will be shaped by continued innovation, cross-disciplinary collaboration, and a commitment to balancing technological advancement with human expertise. As we move forward, the cybersecurity community must remain vigilant, adaptive, and committed to continuous improvement in the face of constantly evolving threats. Developing *SecOps_360* as both a theoretical framework and practical platform demonstrates that unified, integrated approaches to incident management can deliver measurable improvements in operational efficiency whilst addressing multiple challenges simultaneously. By bringing together threat intelligence sources, streamlining workflows, and reducing system resource consumption, this research provides both theoretical foundations and practical solutions, advancing the field of security operations and incident management. The substantial performance improvements demonstrated through rigorous evaluation validate the approach taken and establish a foundation for future innovations in this critical domain of cybersecurity practice.

References

- Ab Rahman, N. H. (2016), An evidence-based cloud incident handling framework, PhD thesis, University of South Australia.
- Abd Majid, M. and Zainol Ariffin, K. A. (2021), 'Model for successful development and implementation of cyber security operations centre (soc)', *Plos one* **16**(11), e0260157.
- Aboaoja, F. A., Zainal, A., Ghaleb, F. A., Al-rimy, B. A. S., Eisa, T. A. E. and Elnour, A. A. H. (2022), 'Malware detection issues, challenges, and future directions: A survey', *Applied Sciences* **12**(17), 8482.
- Afreen, A., Aslam, M. and Ahmed, S. (2020), Analysis of fileless malware and its evasive behavior, in '2020 International Conference on Cyber Warfare and Security (ICCWS)', IEEE, pp. 1–8.
- Agyepong, E., Cherdantseva, Y., Reinecke, P. and Burnap, P. (2020), Towards a framework for measuring the performance of a security operations center analyst, in '2020 international conference on cyber security and protection of digital services (cyber security)', IEEE, pp. 1–8.
- Agyepong, E., Cherdantseva, Y., Reinecke, P. and Burnap, P. (2023), 'A systematic method for measuring the performance of a cyber security operations centre analyst', *Computers & Security* **124**, 102959.
- Ahmad, A., Maynard, S. B., Desouza, K. C., Kotsias, J., Whitty, M. T. and Baskerville, R. L. (2021), 'How can organizations develop situation awareness for incident response: A case study of management practice', *Computers & Security* **101**, 102122.
- Akinrolabu, O., Agrafiotis, I. and Erola, A. (2018), The challenge of detecting sophisticated attacks: Insights from soc analysts, in 'Proceedings of the 13th International Conference on Availability, Reliability and Security', pp. 1–9.
- Alharbi, N. (2020), A Security Operation Center Maturity Model (SOC-MM) in the Context of Newly Emerging Cyber Threats, PhD thesis, The Claremont Graduate University.
- Ali, H., Ahmad, J., Jaroucheh, Z., Papadopoulos, P., Pitropakis, N., Lo, O., Abramson, W. and Buchanan, W. J. (2022), 'Trusted threat intelligence sharing in practice and performance benchmarking through the hyperledger fabric platform', *Entropy* **24**, 1379.
- Alrubayyi, H., Goteng, G., Jaber, M. and Kelly, J. (2021), 'Challenges of malware detection in the iot and a review of artificial immune system approaches', *Journal of Sensor and Actuator Networks* **10**(4), 61.

- Alsmadi, T. and Alqudah, N. (2021), A survey on malware detection techniques, in '2021 International Conference on Information Technology (ICIT)', IEEE, pp. 371–376.
- Alves, F., Bettini, A., Ferreira, P. M. and Bessani, A. (2021), 'Processing tweets for cybersecurity threat awareness', *Information Systems* **95**, 101586.
- Amer, E. and Zelinka, I. (2020), 'A dynamic windows malware detection and prediction method based on contextual understanding of api call sequence', *Computers & Security* **92**, 101760.
- Arfeen, A., Asim Khan, M., Zafar, O. and Ahsan, U. (2022), 'Process based volatile memory forensics for ransomware detection', *Concurrency and Computation: Practice and Experience* **34**(4), e6672.
- Aslam, M. M., Tufail, A., Apong, R. A. A. H. M., De Silva, L. C. and Raza, M. T. (2024), 'Scrutinizing security in industrial control systems: An architectural vulnerabilities and communication network perspective', *IEEE Access* .
- Aslan, Ö., Ozkan-Okay, M. and Gupta, D. (2021), 'A review of cloud-based malware detection system: Opportunities, advances and challenges', *European Journal of Engineering and Technology Research* **6**(3), 1–8.
- Aslan, Ö. and Samet, R. (2017), Investigation of possibilities to detect malware using existing tools, in '2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)', IEEE, pp. 1277–1284.
- Aslan, Ö. and Yilmaz, A. A. (2021), 'A new malware classification framework based on deep learning algorithms', *Ieee Access* **9**, 87936–87951.
- Aurangzeb, S. and Aleem, M. (2023), 'Evaluation and classification of obfuscated android malware through deep learning using ensemble voting mechanism', *Scientific Reports* **13**(1), 3093.
- Barrett, M. P. (2018), 'Framework for improving critical infrastructure cybersecurity', *National Institute of Standards and Technology, USA, Tech. Rep* .
- Basyurt, A. S., Fromm, J., Kuehn, P., Kaufhold, M.-A. and Mirbabaie, M. (2022), 'Help wanted-challenges in data collection, analysis and communication of cyber threats in security operation centers'.
- Bennett, K. W. and Robertson, J. (2023), Cyberspace exercises: defending against malicious cyber actors, in 'Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications V', Vol. 12538, SPIE, pp. 57–72.
- Beqqali, Y., Sadik, H. E., Herrmann, F. and Sabatier, F. (2019), Malware's double anti analysis (daa) analysis methodology, in '2019 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)', IEEE, pp. 1–8.
- Bhatt, S., Manadhata, P. K. and Zomlot, L. (2014), 'The operational role of security information and event management systems', *IEEE security & Privacy* **12**(5), 35–41.

- Bronk, H., Thorbruegge, M. and Hakkaja, M. (2006), 'A step-by-step approach on how to set up a csirt', *European Network and Information Security Agency (ENISA)* .
- Brower, H. H., Nicklas, B. J., Nader, M. A., Trost, L. M. and Miller, D. P. (2021), 'Creating effective academic research teams: Two tools borrowed from business practice', *Journal of Clinical and Translational Science* **5**(1), e74.
- Carrera-Rivera, A., Ochoa, W., Larrinaga, F. and Las, G. (2022), 'How-to conduct a systematic literature review: A quick guide for computer science research', *MethodsX* **9**, 101895.
- Carrier, T., Victor, P., Tekeoglu, A. and Lashkari, A. H. (2022), Detecting obfuscated malware using memory feature engineering., in 'ICISSP', pp. 177–188.
- Chae, D.-K., Park, S.-J., Kim, E., Hong, J. and Kim, S.-W. (2021), 'Identifying the author group of malwares through graph embedding and human-in-the-loop classification', *Applied Sciences* **11**(14), 6640.
- Chai, Y., Chen, X., Qiu, J., Du, L., Xiao, Y., Feng, Q., Ji, S. and Tian, Z. (2024), 'Malfscil: A few-shot class-incremental learning approach for malware detection', *IEEE Transactions on Information Forensics and Security* .
- Chanajitt, R., Pfahringer, B. and Gomes, H. M. (2021), Combining static and dynamic analysis to improve machine learning-based malware classification, in '2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)', IEEE, pp. 1–10.
- Chaudhary, S. (2020), Advance malware analysis using static and dynamic methodology, Technical report, EasyChair.
- Chen, C.-K., Lin, S.-C., Huang, S.-C., Chu, Y.-T., Lei, C.-L. and Huang, C.-Y. (2022), 'Building machine learning-based threat hunting system from scratch', *Digital Threats: Research and Practice (DTRAP)* **3**(3), 1–21.
- Chen, T. R., Shore, D. B., Zaccaro, S. J., Dalal, R. S., Tetrick, L. E. and Gorab, A. K. (2014), 'An organizational psychology perspective to examining computer security incident response teams', *IEEE Security & Privacy* **12**(5), 61–67.
- Cho, S. Y., Happa, J. and Creese, S. (2020), 'Capturing tacit knowledge in security operation centers', *IEEE Access* **8**, 42021–42041.
- Choudhary, S. and Vidyarthi, M. D. (2015), 'A simple method for detection of metamorphic malware using dynamic analysis and text mining', *Procedia Computer Science* **54**, 265–270.
- Cichonski, P., Millar, T., Grance, T., Scarfone, K. et al. (2012), 'Computer security incident handling guide', *NIST Special Publication* **800**(61), 1–147.
- Cook, M., Marnerides, A., Johnson, C. and Pezaros, D. (2023), 'A survey on industrial control system digital forensics: Challenges, advances and future directions', *IEEE Communications Surveys & Tutorials* **25**(3), 1705–1747.

- Creswell, J. W. and Creswell, J. D. (2017), *Research design: Qualitative, quantitative, and mixed methods approaches*, Sage publications.
- Danneman, N. and Gove, R. (2022), Tuning automatic summarization for incident report visualization, in '2022 IEEE 15th Pacific Visualization Symposium (PacificVis)', IEEE, pp. 191–195.
- Danquah, P. (2020), 'Security operations center: a framework for automated triage, containment and escalation', *Journal of Information Security* **11**(4), 225–240.
- Darem, A. A. (2022), 'A novel framework for windows malware detection using a deep learning approach.', *Computers, Materials & Continua* **72**(1).
- Darem, A. A., Ghaleb, F. A., Al-Hashmi, A. A., Abawajy, J. H., Alanazi, S. M. and Al-Rezami, A. Y. (2021), 'An adaptive behavioral-based incremental batch learning malware variants detection model using concept drift detection and sequential deep learning', *IEEE Access* **9**, 97180–97196.
- Datta, U., Kalam, A. and Shi, J. (2019), 'Battery energy storage system control for mitigating pv penetration impact on primary frequency control and state-of-charge recovery', *IEEE Transactions on Sustainable Energy* **11**(2), 746–757.
- Dawodu, S. O., Omotosho, A., Akindote, O. J., Adegbite, A. O. and Ewuga, S. K. (2023), 'Cybersecurity risk assessment in banking: methodologies and best practices', *Computer Science & IT Research Journal* **4**(3), 220–243.
- De Andrade, C. A. B., de Mello, C. G. and Duarte, J. C. (2013), Malware automatic analysis, in '2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence', IEEE, pp. 681–686.
- de Céspedes III, R. and Dimitoglou, G. (2021), 'Development of a virtualized security operations center', *Journal of Computing Sciences in Colleges* **37**(3), 108–119.
- Dener, M., Ok, G. and Orman, A. (2022), 'Malware detection using memory analysis data in big data environment', *Applied Sciences* **12**(17), 8604.
- Dhaka, P. and Johari, R. (2016), Crib: Cyber crime investigation, data archival and analysis using big data tool, in '2016 International Conference on Computing, Communication and Automation (ICCCA)', IEEE, pp. 117–121.
- Dietz, M., Vielberth, M. and Pernul, G. (2020), Integrating digital twin security simulations in the security operations center, in 'Proceedings of the 15th International Conference on Availability, Reliability and Security', pp. 1–9.
- Ehsan, A., Catal, C. and Mishra, A. (2022), 'Detecting malware by analyzing app permissions on android platform: A systematic literature review', *Sensors* **22**(20), 7928.
- Elradi, M. D., Abdelmaged, K. A. and Mohammad, M. O. (2021), 'Cyber security professionals' challenges: A proposed integrated platform solution', *Electrical Science & Engineering* **3**(2), 1–6.

- Field, A. (2024), *Discovering statistics using IBM SPSS statistics*, 6th edn, SAGE Publications, London. Wilcoxon Signed-Rank Test: Chapter 7. Cohen's d effect size: Chapter 3. Within-subjects design: Chapter 8.
- Flechais, I. and Chalhoub, G. (2023), Practical cybersecurity ethics: mapping CyBOK to ethical concerns, in 'Proceedings of the 2023 New Security Paradigms Workshop', ACM, pp. 62–75.
- Foreman, P. (2019), *Vulnerability management*, Auerbach Publications.
- Forsberg, J. and Frantti, T. (2023), 'Technical performance metrics of a security operations center', *Computers & Security* **135**, 103529.
- Gad, R., Kappes, M. and Medina-Bulo, I. (2015), Monitoring traffic in computer networks with dynamic distributed remote packet capturing, in '2015 IEEE International Conference on Communications (ICC)', IEEE, pp. 5759–5764.
- Gandotra, E., Bansal, D. and Sofat, S. (2014), 'Malware analysis and classification: A survey', *Journal of Information Security* **2014**.
- Gao, P. et al. (2021), A system for efficiently hunting for cyber threats in computer systems using threat intelligence, in 'IEEE 37th Int. Conf. on Data Engineering', IEEE, pp. 2705–2708.
- Gnanasekaran, V., Neudert, R., Heegaard, P. E. and Pernul, G. (2025), A role taxonomy in security-safety incident response, in 'International Conference on Availability, Reliability and Security', Springer, pp. 286–304.
- González-Granadillo, G., González-Zarzosa, S. and Diaz, R. (2021), 'Security information and event management (siem): analysis, trends, and usage in critical infrastructures', *Sensors* **21**(14), 4759.
- Gormont, N. Z., Selamat, A., Cheng, L. K. and Krejcar, O. (2023), 'Machine learning algorithm for malware detection: taxonomy, current challenges and future directions', *IEEE Access*.
- Goupil, F., Laskov, P., Pekaric, I., Felderer, M., Dürr, A. and Thiesse, F. (2022), Towards understanding the skill gap in cybersecurity, in 'Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1', pp. 477–483.
- Grier, C., Tang, S. and King, S. T. (2008), Secure web browsing with the op web browser, in '2008 IEEE Symposium on Security and Privacy (sp 2008)', IEEE, pp. 402–416.
- Guo, H., Huang, S., Huang, C., Pan, Z., Zhang, M. and Shi, F. (2020), 'File entropy signal analysis combined with wavelet decomposition for malware classification', *IEEE Access* **8**, 158961–158971.
- Gupta, B., Mufti, T., Sohail, S. S. and Madsen, D. Ø. (2023), 'Chatgpt: A brief narrative review', *Cogent Business & Management* **10**(3), 2275851.
- Gupta, M., Mittal, S. and Abdelsalam, M. (2020), 'Ai assisted malware analysis: a course for next generation cybersecurity workforce', *arXiv preprint arXiv:2009.11101*.

- Hadiprakoso, R. B., Kabetta, H. and Buana, I. K. S. (2020), Hybrid-based malware analysis for effective and efficiency android malware detection, in '2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)', IEEE, pp. 8–12.
- Hall, K. L., Vogel, A. L., Huang, G. C., Serrano, K. J., Rice, E. L., Tsakraklides, S. P. and Fiore, S. M. (2018), 'The science of team science: A review of the empirical evidence and research gaps on collaboration in science.', *American psychologist* **73**(4), 532.
- Hao, J., Luo, S. and Pan, L. (2022), 'Eii-mbs: Malware family classification via enhanced adversarial instruction behavior semantic learning', *Computers & Security* **122**, 102905.
- Hassan, I. U. and Asghar, S. (2021), 'A framework of software project scope definition elements: An ism-dematel approach', *IEEE Access* **9**, 26839–26870.
- Hofhuis, J., Mensen, M., ten Den, L. M., van den Berg, A. M., Koopman-Draijer, M., van Tilburg, M. C., Smits, C. H. and de Vries, S. (2018), 'Does functional diversity increase effectiveness of community care teams? the moderating role of shared vision, interaction frequency, and team reflexivity', *Journal of applied social psychology* **48**(10), 535–548.
- Hossain, S. M., Couturier, R., Rusk, J. and Kent, K. B. (2021), Automatic event categorizer for siem, in 'Proceedings of the 31st Annual International Conference on Computer Science and Software Engineering', pp. 104–112.
- Ijaz, M., Durad, M. H. and Ismail, M. (2019), Static and dynamic malware analysis using machine learning, in '2019 16th International bhurban conference on applied sciences and technology (IBCAST)', IEEE, pp. 687–691.
- Ioannou, M. e. a. (2019), Cybersecurity culture in computer security incident response teams: Investigating difficulties in communication and coordination, in 'Int. Conf. on Cyber Security and Protection of Digital Services', IEEE, pp. 1–4.
- ISC2 (2023), 'Cybersecurity workforce study 2023', ISC2.
URL: <https://cybergovernancealliance.org/wp-content/uploads/2024/01/ISC2cybersecurityworkforcestu1.pdf>
- Islam, C., Babar, M. A., Croft, R. and Janicke, H. (2022), 'Smartvalidator: a framework for automatic identification and classification of cyber threat data', *Journal of Network and Computer Applications* **202**, 103370.
- Islam, C., Babar, M. A. and Nepal, S. (2019), 'A multi-vocal review of security orchestration', *ACM Computing Surveys (CSUR)* **52**(2), 1–45.
- Jacobs, P., Arnab, A. and Irwin, B. (2013), Classification of security operation centers, in '2013 Information Security for South Africa', IEEE, pp. 1–7.
- János, F. D. and Dai, N. H. P. (2018), Security concerns towards security operations centers, in '2018 IEEE 12th International Symposium on Applied Computational Intelligence and Informatics (SACI)', IEEE, pp. 000273–000278.
- Jaramillo, L. E. (2019), 'Malware threats analysis and mitigation techniques for compromised systems', *Jour. of Information Systems Engineering & Mangt.* **4**(1).

- Jha, R. K. (2023), 'Cybersecurity and confidentiality in smart grid for enhancing sustainability and reliability', *Recent Research Reviews Journal* **2**(2), 215–241.
- Jiang, Y., Jeusfeld, M. A., Ding, J. and Sandahl, E. (2023), 'Model-based cybersecurity analysis: Y. jiang et al.', *Business & Information Systems Engineering* **65**(6), 643–676.
- Jin, S., Guo, Z., Liu, D., Yang, Y. et al. (2022), 'A study on the application of distributed system technology-guided machine learning in malware detection', *Computational Intelligence and Neuroscience* **2022**.
- Jindal, C., Salls, C., Aghakhani, H., Long, K., Kruegel, C. and Vigna, G. (2019), Neurlux: dynamic malware analysis without feature engineering, in 'Proceedings of the 35th Annual Computer Security Applications Conference', pp. 444–455.
- Kara, I. (2022), 'Fileless malware threats: Recent advances, analysis approach through memory forensics and research challenges', *Expert Systems with Applications* p. 119133.
- Kayani, A. K. and Saeed, M. Q. (2021), Comparative analysis of anti-virus evasion malware creator tools of kali linux, with proposed model for obfuscation, in '2021 International Conference on Cyber Warfare and Security (ICWS)', IEEE, pp. 24–29.
- Kitchenham, B., Madeyski, L. and Budgen, D. (2022), 'Segress: Software engineering guidelines for reporting secondary studies', *IEEE Transactions on Software Engineering* **49**(3), 1273–1298.
- Kokulu, F. et al. (2019), Matched and mismatched socs: A qualitative study on security operations center issues, in 'Proc. of the 2019 ACM SIGSAC Conf. on Computer and Communications Security', ACM, pp. 1955–1970.
- Kolokotronis, N. and Shiaeles, S. (2021), 'Cyber-security threats, actors, and dynamic mitigation'.
- Kovács, A. (2022), 'Ransomware: a comprehensive study of the exponentially increasing cybersecurity threat', *Insights into Regional Development* **4**(2), 96–104.
- Kowtha, S. et al (2012), Cyber security operations center characterization model and analysis, in 'IEEE Conf. on Technologies for Homeland Security', IEEE, pp. 470–475.
- Li, L., Cui, J., Zhang, R., Xia, H. and Cheng, X. (2020), 'Dynamics of complex networks: Malware propagation modeling and analysis in industrial internet of things', *IEEE Access* **8**, 64184–64192.
- Mahmood, S., Chadhar, M. and Firmin, S. (2024), 'Digital resilience framework for managing crisis: A qualitative study in the higher education and research sector', *Journal of Contingencies and Crisis Management* **32**(1), e12549.
- Makandar, A. and Patrot, A. (2015), Malware analysis and classification using artificial neural network, in 'International conference on trends in automation, communications and computing technology', IEEE, pp. 1–6.
- Markevych, M. and Dawson, M. (2023), A review of enhancing intrusion detection systems for cybersecurity using artificial intelligence (ai), in 'International conference Knowledge-based Organization', Vol. 29, pp. 30–37.

- Marquardson, J. (2022), 'Soc it to 'em: Bringing a security operations center onto a university campus', *Journal of Information Systems Education* **33**(3), 300–305.
- McLaughlin, M.-D. and Gogan, J. (2018), 'Challenges and best practices in information security management', *MIS Quarterly Executive* **17**(3), 12.
- Meyer, P. and Métille, S. (2023), 'Computer security incident response teams: are they legally regulated? the swiss example', *International Cybersecurity Law Review* **4**(1), 39–60.
- Miloslavskaya, N. (2016), Security operations centers for information security incident management, in '2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)', IEEE, pp. 131–136.
- Mohanta, A. and Saldanha, A. (2020), *Malware Analysis and Detection Engineering: A Comprehensive Approach to Detect and Analyze Modern Malware*, Springer.
- Moser, A., Kruegel, C. and Kirda, E. (2007a), Exploring multiple execution paths for malware analysis, in '2007 IEEE Symposium on Security and Privacy (SP'07)', IEEE, pp. 231–245.
- Moser, A., Kruegel, C. and Kirda, E. (2007b), Limits of static analysis for malware detection, in 'Twenty-third annual computer security applications conference (ACSAC 2007)', IEEE, pp. 421–430.
- Mughal, A. A. (2022), 'Building and securing the modern security operations center (soc)', *International Journal of Business Intelligence and Big Data Analytics* **5**(1), 1–15.
- Muniz, J., McIntyre, G. and AlFardan, N. (2015), *Security Operations Center: Building, Operating, and Maintaining Your SOC*, Cisco Press.
- Murali, R., Ravi, A. and Agarwal, H. (2020), A malware variant resistant to traditional analysis techniques, in '2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)', IEEE, pp. 1–7.
- Murthy, G., Chowdary, M., Sangameswar, M. and Vital, T. (2019), 'Exploring the api calls for malware behavior detection using concordance and document frequency', *International Journal of Advanced Technology and Engineering Exploration* pp. 8(6):2249–8958.
- Mutemwa, M. et al. (2018), Integrating a security operations centre with an organization's existing procedures, policies and information technology systems, in 'Int. Conf. on Intelligent and Innovative Computing Applications', IEEE, pp. 1–6.
- Mwangi, J., Wall, J., Ismail, U. and Al-Nemrat, A. (2023), Building a human-centric soc: A new framework for success, in 'International Conference on Global Security, Safety, and Sustainability', Springer, pp. 537–558.
- Nadeem, A., Rimmer, V., Joosen, W. and Verwer, S. (2022), Intelligent malware defenses, in 'Security and artificial intelligence: A crossdisciplinary approach', Springer, pp. 217–253.
- Naik, N., Jenkins, P., Savage, N., Yang, L., Boongoen, T. and Iam-On, N. (2020), Fuzzy-import hashing: A malware analysis approach, in '2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)', IEEE, pp. 1–8.

- Naseer, A., Naseer, H., Ahmad, A., Maynard, S. B. and Siddiqui, A. M. (2023), 'Moving towards agile cybersecurity incident response: A case study exploring the enabling role of big data analytics-embedded dynamic capabilities', *Computers & Security* **135**, 103525.
- Naseer, H. (2018), A framework of dynamic cybersecurity incident response to improve incident response agility, PhD thesis.
- Neri, M., Niccolini, F., Francesco, V. et al. (2023), 'Organizational resilience: state of the art and new future cyber inquiries', *Impresa Progetto* **1**(1), 1–33.
- Nifakos, S., Chandramouli, K., Nikolaou, C. K., Papachristou, P., Koch, S., Panaousis, E. and Bonacina, S. (2021), 'Influence of human factors on cyber security within healthcare organisations: a systematic review', *Sensors* **21**, 5119.
- Nyre-Yu, M., Gutzwiller, R. S. and Caldwell, B. S. (2019), Observing cyber security incident response: qualitative themes from field research, in 'Proceedings of the Human Factors and Ergonomics Society Annual Meeting', Vol. 63, SAGE Publications Sage CA: Los Angeles, CA, pp. 437–441.
- Nyre-Yu, M. M. (2019), Determining System Requirements for Human-Machine Integration in Cyber Security Incident Response, PhD thesis, Purdue University Graduate School.
- Oesch, S, e. a. (2020), An assessment of the usability of machine learning based tools for the security operations center, in '2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)', IEEE, pp. 634–641.
- Oh, C., Ha, J. and Roh, H. (2021), 'A survey on tls-encrypted malware network traffic analysis applicable to security operations centers', *Applied Sciences* **12**(1), 155.
- O'Kane, P., Sezer, S. and McLaughlin, K. (2011), 'Obfuscation: The hidden malware', *IEEE Security & Privacy* **9**(5), 41–47.
- Okolica, J. and Peterson, G. L. (2010), 'Windows operating systems agnostic memory analysis', *Digital investigation* **7**, S48–S56.
- Onwubiko, C. (2015), Cyber security operations centre: Security monitoring for protecting business and supporting cyber defense strategy, in '2015 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)', IEEE, pp. 1–10.
- Onwubiko, C. (2021), Rethinking security operations centre onboarding, in '2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)', IEEE, pp. 1–9.
- Onwubiko, C. (2022), 'Cyberops: Situational awareness in cybersecurity operations', *arXiv preprint arXiv:2202.03687*.
- Onwubiko, C. and Ouazzane, K. (2019), Cyber onboarding is 'broken', in '2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)', IEEE, pp. 1–13.

- Onwubiko, C. and Ouazzane, K. (2020), ‘Soter: A playbook for cybersecurity incident management’, *IEEE Transactions on Engineering Management* **69**(6), 3771–3791.
- Onwubiko, C. and Ouazzane, K. (2022a), ‘Challenges towards building an effective cyber security operations centre’, *arXiv preprint arXiv:2202.03691*.
- Onwubiko, C. and Ouazzane, K. (2022b), ‘Soter: a playbook for cybersecurity incident management’, *IEEE Transactions on Engineering Management* **69**, 3771–3791.
- Or-Meir, O., Nissim, N., Elovici, Y. and Rokach, L. (2019), ‘Dynamic malware analysis in the modern era—a state of the art survey’, *ACM Computing Surveys (CSUR)* **52**(5), 1–48.
- Oriola, O., Adeyemo, A. B., Papadaki, M. and Kotzé, E. (2021), ‘A collaborative approach for national cybersecurity incident management’, *Information & Computer Security* **29**(3), 457–484.
- Ortloff, A.-M., Tiefenau, C. and Smith, M. (2023), {SoK}: I have the (developer) power! sample size estimation for fisher’s exact, {Chi-Squared}, {McNemar’s}, wilcoxon {Rank-Sum}, wilcoxon {Signed-Rank} and t-tests in {Developer-Centered} usable security, in ‘Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)’, pp. 341–359.
- Pachhala, N., Jothilakshmi, S. and Battula, B. P. (2021), A comprehensive survey on identification of malware types and malware classification using machine learning techniques, in ‘2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)’, IEEE, pp. 1207–1214.
- Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E. et al. (2021), ‘The prisma 2020 statement: an updated guideline for reporting systematic reviews’, *bmj* **372**.
- Park, S., Jeon, S. and Kim, H. K. (2022), ‘Hmlet: Hunt malware using wavelet transform on cross-platform’, *IEEE Access* **10**, 124821–124834.
- Pawlicka, A., Pawlicki, M., Kozik, R. and Choraś, R. S. (2021), ‘A systematic review of recommender systems and their applications in cybersecurity’, *Sensors* **21**(15), 5248.
- Pedreira, V., Barros, D. and Pinto, P. (2021), ‘A review of attacks, vulnerabilities, and defenses in industry 4.0 with new challenges on data sovereignty ahead’, *Sensors* **21**(15), 5189.
- Peffer, K., Tuunanen, T., Rothenberger, M. A. and Chatterjee, S. (2007), ‘A design science research methodology for information systems research’, *Journal of management information systems* **24**(3), 45–77.
- Plohmann, D. et al. (2013), Patterns of a cooperative malware analysis workflow, in ‘5th Int. Conf. on Cyber Conflict’, IEEE, pp. 1–18.
- Prähofer, H., Angerer, F., Ramler, R., Lacheiner, H. and Grillenberger, F. (2012), Opportunities and challenges of static code analysis of IEC 61131-3 programs, in ‘Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)’, IEEE, pp. 1–8.

- Preuveneers, D., Joosen, W., Bernal Bernabe, J., Skarmeta, A. et al. (2020), 'Distributed security framework for reliable threat intelligence sharing', *Security and Communication Networks* **2020**.
- Preuveneers, D., Llamas, J. M., Bulut, I., Rúa, E. A., Verfaillie, P., Demortier, V., Surinx, D. and Joosen, W. (2023), On the use of automl for combating alert fatigue in security operations centers, in 'European Symposium on Research in Computer Security', Springer, pp. 609–627.
- Rajapakse, R. N., Zahedi, M., Babar, M. A. and Shen, H. (2022), 'Challenges and solutions when adopting devsecops: A systematic review', *Information and Software Technology* **141**, 106700.
- Rashid, A., Chivers, H., Lupu, E., Martin, A. and Schneider, S., eds (2021), *The Cyber Security Body of Knowledge (CyBOK), Version 1.1.0*, University of Bristol.
URL: https://www.cybok.org/wp-content/uploads/CyBOK_v1.1.0.pdf
- Reeves, A. and Ashenden, D. (2023), 'Understanding decision making in security operations centres: building the case for cyber deception technology', *Frontiers in Psychology* **14**, 1165705.
- Roseline, S. A., Geetha, S., Kadry, S. and Nam, Y. (2020), 'Intelligent vision-based malware detection and classification using deep random forest paradigm', *IEEE Access* **8**, 206303–206324.
- Salman, M., Husna, D. and Viani, N. (2019), Static analysis method on portable executable files for remnux based malware identification, in '2019 IEEE 10th International Conference on Awareness Science and Technology (iCAST)', IEEE, pp. 1–6.
- Saqib, M., MahdaviFar, S., Fung, B. C. and Charland, P. (2024), 'A comprehensive analysis of explainable ai for malware hunting', *ACM Computing Surveys* **56**(12), 1–40.
- Saunders, M. (2009), 'Research methods for business students', *Person Education Limited*.
- Sawalha, I. H. (2021), 'Views on business continuity and disaster recovery', *International Journal of Emergency Services* **10**(3), 351–365.
- Schofield, M., Alicioglu, G., Binaco, R., Turner, P., Thatcher, C., Lam, A. and Sun, B. (2021), Convolutional neural network for malware classification based on api call sequence, in 'Proceedings of the 8th International Conference on Artificial Intelligence and Applications (AIAP 2021), Zurich, Switzerland', pp. 23–24.
- Shahjee, D. and Ware, N. (2022), 'Integrated network and security operation center: A systematic analysis', *IEEE Access* **10**, 27881–27898.
- Sharma, S. and Bharti, S. (2021), Malware analysis using ensemble techniques: A machine learning approach, in '2021 International Conference on Artificial Intelligence and Machine Vision (AIMV)', IEEE, pp. 1–5.
- Shree, R., Shukla, A. K., Pandey, R. P., Shukla, V. and Bajpai, D. (2022), 'Memory forensic: Acquisition and analysis mechanism for operating systems', *Materials Today: Proceedings* **51**, 254–260.

- Sihwail, R., Omar, K. and Ariffin, K. Z. (2018), 'A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis', *Int. J. Adv. Sci. Eng. Inf. Technol* **8**(4-2), 1662–1671.
- Singh, J. and Singh, J. (2018), 'Challenge of malware analysis: malware obfuscation techniques', *International Journal of Information Security Science* **7**(3), 100–110.
- Skendžić, A., Kovačić, B. and Balon, B. (2022), Management and monitoring security events in a business organization-siem system, in '2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)', IEEE, pp. 1203–1208.
- Soltani, N., Rahmani, A. M., Bohlouli, M. and Hosseinzadeh, M. (2022), 'Artificial intelligence empowered threat detection in the internet of things: A systematic review', *Concurrency and Computation: Practice and Experience* **34**(22), e6894.
- Soni, H., Kishore, P. and Mohapatra, D. P. (2022), Opcode and api based machine learning framework for malware classification, in '2022 2nd International Conference on Intelligent Technologies (CONIT)', IEEE, pp. 1–7.
- Souppaya, M., Scarfone, K. et al. (2013), 'Guide to malware incident prevention and handling for desktops and laptops', *NIST Special Publication* **800**, 83.
- Subedi, K. P., Budhathoki, D. R. and Dasgupta, D. (2018), Forensic analysis of ransomware families using static and dynamic analysis, in '2018 IEEE Security and Privacy Workshops (SPW)', IEEE, pp. 180–185.
- Tafazzoli, T. e. a. (2016), Security operation center implementation on openstack, in '8th International Symposium on Telecommunications', IEEE, pp. 766–770.
- Tcharnetsky, M. and Vogt, F. (2023), 'The osqe model: The ai cycle against the shortage of skilled professionals: A holistic solution approach based on artificial intelligence in times of demographic change'.
- Thompson, E. C. (2018), *Cybersecurity incident response: How to contain, eradicate, and recover from incidents*, Apress.
- Torres, A. (2015), 'Building a world-class security operations center: A roadmap', *SANS Institute*, May .
- Turner, S. L., Forbes, A. B., Karahalios, A., Taljaard, M. and McKenzie, J. E. (2021), 'Evaluation of statistical methods used in the analysis of interrupted time series studies: a simulation study', *BMC medical research methodology* **21**(1), 181.
- US-CERT (2013), 'Incident management'. <https://www.us-cert.gov/bsi/articles/best-practices/incident-management/incident-management>.
- Vaarandi, R. and Mäses, S. (2022), How to build a soc on a budget, in '2022 IEEE International Conference on Cyber Security and Resilience (CSR)', IEEE, pp. 171–177.
- Van Haastrecht, M., Sarhan, I., Shojafar, A., Baumgartner, L., Mallouli, W. and Spruit, M. (2021), A threat-based cybersecurity risk assessment approach addressing sme needs, in 'Proceedings of the 16th International Conference on Availability, Reliability and Security', pp. 1–12.

- Vielberth, M., Böhm, F., Fichtinger, I. and Pernul, G. (2020), 'Security operations center: A systematic study and open challenges', *Ieee Access* **8**, 227756–227779.
- Vurdelja, I., Blažić, I., Bojić, D. and Drašković, D. (2021), 'An automated framework for runtime analysis of malicious executables on linux', *Telfor Journal* **13**(2), 87–91.
- Wael, D. and Azer, M. A. (2018), Malware incident handling and analysis workflow, in '14th International Computer Engineering Conference', IEEE, pp. 242–248.
- Walkowski, M., Oko, J. and Sujecki, S. (2021), 'Vulnerability management models using a common vulnerability scoring system', *Applied Sciences* **11**(18), 8735.
- Wang, J., Tian, Y., An, D., Liang, Z., Guo, C., Hu, H., Luo, Q., Li, H., Wang, H., Zeng, S., Zhou, C., Ma, L. and Qi, F. Z. (2021), 'A comprehensive security operation center based on big data analytics and threat intelligence', *Proceedings of International Symposium on Grids Amp; Clouds 2021 — PoS(ISGC2021)*.
- Wang, N. and Fu, H. (2024), Research on deep-learning-based techniques for advanced persistent threat malware detection and attribution, in 'Fourth International Conference on Sensors and Information Technology (ICSI 2024)', Vol. 13107, SPIE, pp. 596–603.
- Wang, X. (2018), *A secure computing platform for building automation using microkernel-based operating systems*, University of South Florida.
- Wang, Z. and Zhu, Y. (2017), A centralized hids framework for private cloud, in '2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)', IEEE, pp. 115–120.
- Wazid, M., Das, A. K., Rodrigues, J. J., Shetty, S. and Park, Y. (2019), 'Iomt malware detection approaches: analysis and research challenges', *IEEE Access* **7**, 182459–182476.
- Wei, Q., Xiao, S. and Dongbao, L. (2019), Malware classification system based on machine learning, in '2019 Chinese Control And Decision Conference (CCDC)', IEEE, pp. 647–652.
- Xia, H., Li, J., Weng, J., Zhang, Z. and Gao, Y. (2021), 'Collaborative knowledge sharing in global distributed teams: antecedents of innovation performance', *Journal of Knowledge Management* **25**(10), 2523–2539.
- Xiao, K., Zhu, C., Xie, J., Zhou, Y., Zhu, X. and Zhang, W. (2020), 'Dynamic defense against stealth malware propagation in cyber-physical systems: a game-theoretical framework', *Entropy* **22**(8), 894.
- Yaacoub, J.-P. A., Noura, H. N., Salman, O. and Chehab, A. (2022), 'Advanced digital forensics and anti-digital forensics for iot systems: Techniques, limitations and recommendations', *Internet of Things* p. 100544.
- Yucel, C., Lockett, A., Chalkias, I., Mallis, D. and Katos, V. (2021), 'Mait: Malware analysis and intelligence tool', *Information & Security* **50**(1), 49–65.
- Zeltser, L. (2019), 'How malware generates mutex names to evade detection', *online documentation at <https://isc.sans.edu/forums/diary/How+Malware+Generates+Mutex+Names+to+Evade+Detection/19429/>*, retrieved 7(01).

- Zheng, L., Zhang, J. et al. (2022), 'A new malware detection method based on vmcadr in cloud environments', *Security and Communication Networks* **2022**.
- Zhu, X. and Huang, J. (2021), 'Malware propagation model for cluster-based wireless sensor networks using epidemiological theory', *PeerJ Computer Science* **7**, e728.
- Zoppelt, M. J. (2021), Software Security Design for Cyber-Physical Systems, PhD thesis, Friedrich-Alexander-Universitaet Erlangen-Nuernberg (Germany).

Appendix A

Appendix

NameSpace	Description
GuzzleHttp	Imported from the GuzzleHttp library, which is used for making HTTP requests
Symfony\Component\HttpFoundation\Request	Used to handle incoming HTTP requests
Symfony\Component\Routing\Annotation\Route	Used for defining the routes (URLs) for the controller actions.
Symfony\Bundle\FrameworkBundle\Controller\AbstractController	base class for Symfony controllers

Table A.1 Imported Classes from Namespaces and Descriptions

A.1 Appendix 1 - BackEnd Source Code

Fig. A.1 represents the first block of backend code. In this block, the namespace *App* is defined, which specifies the location of the controller class within the application's directory structure. Additionally, several classes are imported from different namespaces described in Tab A.1

```
namespace App\Controller;
use GuzzleHttp\Client as GuzzleHttp;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
```

Fig. A.1 Backend Source Code Block1

Fig A.2 represents the second block of the backend code. This block defines a new class named *SearchController* that extends the *AbstractController* class. The *@Route("/search")* annotation defines the base URL path for all actions within this controller.

Fig A.3 shows the third block of the backend code. This block declares and initialises private class properties for various API keys and URLs used throughout the controller. These keys and URLs are related to different services or APIs that the controller interacts with, such as OPSWAT, VirusTotal,

```

/**
 * @Route("/search")
 */
class SearchController extends AbstractController
{

```

Fig. A.2 Backend Source Code Block2

Hybrid Analysis, and AbuseIPDB.

Fig A.4 shows the Fourth block of the backend code. This block defines the main action method `checkup` within the `SearchController`. The `@Route("/check", name="check", methods="POST")` annotation defines the URL path `/check` for this action and specifies that it accepts only POST requests. Inside the method, the following actions are performed:

1. The method verifies if the incoming request is a valid POST request using `\$request->isMethod('POST')`.
2. The request data is retrieved and decoded based on the request type (form data or JSON) using `\$request->request->all()` or `json_decode(\$request->getContent(),true)`

Fig A.5 shows the Fourth block of the backend code. This block interacts with the AbuseIPDB API. It performs the following steps:

1. An instance of the GuzzleHttp client is created with the base URI set to the `\$abuseipURL`.
2. A GET request is made to the `/check` endpoint with the request data and appropriate headers (including the API key).
3. The response from the API is stored in the `\$result['check']` array after decoding the JSON response body.
4. Another GET request is made to the `/check-block` endpoint with the request data and appropriate headers.

```

/**
 * Global declaration of KEYS and Initialization.
 */
private $keyOpswat = 'ee0a10b16392ff3fe0ee872a3676cab4';
private $keyVirus = '58a6fe675b9b849e7086fe4d662b5a6c5df5ae667c322b839c3c7a1f1dd0ea91';
private $keyHybrid = 'fmgvzcjuc8c4674cr8jm6b2x31aea1949dvxp3wkad66e8e01kt4pbnf629c13d8';
private $keyAbuse = 'd3304c0b2f0febcbade4750868a99c9567988ba940031856257929e64588908065698

/**
 * Global Declaration of URLs and Initialization.
 */
private $opswatURL = 'https://api.metadefender.com/v4/';
private $abuseipURL = 'https://api.abuseipdb.com/api/v2/';
private $virusTotalURL = 'https://www.virustotal.com/api/v3/';
private $hybridURL = 'https://www.hybrid-analysis.com/api/v2/quick-scan/file';

```

Fig. A.3 Backend Source Code Block3

5. The response from the API is stored in the *\$result['block']* array after decoding the JSON response body.
6. Any exceptions that occur during the requests are caught and can be handled accordingly.

Fig A.6 shows the Fourth block of the backend code. This block interacts with the AbuseIPDB API. It performs the following steps:

1. An instance of the *GuzzleHttp* client is created with the base URI set to the *\$abuseipURL*.
2. GET request is made to the */check* endpoint with the request data and appropriate headers (including the API key).
3. The response from the API is stored in the *\$result['check']* array after decoding the JSON response body.
4. Another GET request is made to the */check-block* endpoint with the request data and appropriate headers.

```

public function __construct(
    $keyAbuse,
    $keyVirus,
    $keyOpswat,
    $opswatURL,
    $virusTotalURL,
    $abuseipURL,
    $hybridURL
) {
    $this->keyAbuse = $keyAbuse;
    $this->keyVirus = $keyVirus;
    $this->keyOpswat = $keyOpswat;
    $this->opswatURL = $opswatURL;
    $this->virusTotalURL = $virusTotalURL;
    $this->abuseipURL = $abuseipURL;
    $this->hybridURL = $hybridURL;
}

```

Fig. A.4 Backend Source Code Block4

5. The response from the API is stored in the *\$result['block']* array after decoding the JSON response body.
6. Any exceptions that occur during the requests are caught and can be handled accordingly.

Fig A.7 shows the Fourth block of the backend code. This block interacts with the VirusTotal API to check the IP address. It performs the following steps:

1. A new instance of *GuzzleHttp* is created with the base URI set to the *\$virusTotalURL* concatenated with "ip_addresses/" and the *\$ip* value. *AGET* request is made to the VirusTotal API for IP addresses, with the appropriate headers (including the API key).
2. The response from the API is stored in the *\$result['virusIp']* array after decoding the JSON response body.

```

/**
 * Main route accessible in METHOD POST. This route is the only route called in this project
 * all requests and calls to the different APIs used in the application
 * @Route("/check", name="check", methods={"POST"})
 */
public function checkIp(Request $request) {
    // Verification if the requests sent are valid and good in POST METHOD.
    // Once this is verified, we continue the action
    if ($request->isMethod('POST')) {
        // Retrieval of request data according to the results that come to us from the client
        // Then these are encoded again according to the type of request (File upload or s
        $data = $request->request->all() ? : json_decode($request->getContent(), true);

        // Basic verification or data validation system for subsequent queries
        $ip = (!empty($data['ipAddress'])) ? $data['ipAddress'] : null;
        $url = (!empty($data['url'])) ? $data['url'] : null;
        $hash = (!empty($data['hash'])) ? $data['hash'] : null;
        $file = $request->files->get('file');
        $name = ($file) ? time() . '_' . $file->getClientOriginalName() : null;

        // Initialization of the results array
        $result = [];

        // It is good to note that all requests made from this moment are made
        // via an HTTP request utility named GuzzleHttp see documentation https://docs.gu

```

Fig. A.5 Backend Source Code Block5

3. Any exceptions that occur during the request are caught and can be handled accordingly.

Fig A.8 shows the Fourth block of the backend code. This block interacts with the VirusTotal API to check the URL. It performs the following steps:

1. A new instance of GuzzleHttp is created with the base URI set to the *\$virusTotalURL* concatenated with "domains/" and the *\$url* value.
2. A GET request is made to the VirusTotal API for domains, with the appropriate headers (including the API key).
3. The response from the API is stored in the *\$result['virusURL']* array after decoding the *JSON* response body.
4. Any exceptions that occur during the request are caught and can be handled accordingly.

Fig A.9 shows the Fourth block of the backend code. This block interacts with the VirusTotal API to check the file hash. It performs the following steps:

```

/**
 * ABUSEIPDB
 * Formalization of the API POST request
 */
$abuseipdb = new GuzzleHttp(['base_uri' => $this->abuseipURL]);
try {
    $check = $abuseipdb->request('GET', 'check', [
        'query' => $data,
        'headers' => ['Accept' => 'application/json', 'Key' => $this->keyAbuse],
    ]);
    $result['check'] = json_decode($check->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

try {
    $block = $abuseipdb->request('GET', 'check-block', [
        'query' => $data,
        'headers' => ['Accept' => 'application/json', 'Key' => $this->keyAbuse],
    ]);
    $result['block'] = json_decode($block->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.6 Backend Source Code Block6

1. A new instance of *GuzzleHttp* is created with the base URI set to the *\$virusTotalURL* concatenated with "search?query=" and the *\$hash* value.
2. A GET request is made to the VirusTotal API for searching by hash, with the appropriate headers (including the API key).
3. The response from the API is stored in the *\$result['virusHASH']* array after decoding the *JSON* response body.
4. Any exceptions that occur during the request are caught and can be handled accordingly.

Fig A.10 shows the Fourth block of the backend code. This block interacts with the OPSWAT API to check the URL. It performs the following steps:

1. A new instance of *GuzzleHttp* is created with the base URI set to the *\$opswatURL* concatenated with "domain/" and the *\$url* value.
2. A GET request is made to the OPSWAT API for domains, with the appropriate headers (including the API key).

```

/** VIRUSTOTAL **/
//Check IP
$virustotalIp = new GuzzleHttp(['base_uri' => $this->virusTotalURL."ip_addresses/".$ip]);
try {
    $virusIp = $virustotalIp->request('GET', '', [
        'headers' => ['Accept' => 'application/json', 'X-Apkey' => $this->keyVirus],
    ]);
    $result['virusIp'] = json_decode($virusIp->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.7 Backend Source Code Block7

3. The response from the API is stored in the *\$result['opswatURL']* array after decoding the *JSON* response body.
4. Any exceptions that occur during the request are caught and can be handled accordingly.

Fig A.11 shows the Fourth block of the backend code. This block interacts with the OPSWAT API to check a file. It performs the following steps:

1. A new instance of *GuzzleHttp* is created with the base URI set to the *\$opswatURL* concatenated with "file".
2. A *POST* request is made to the OPSWAT API for file uploads, with the appropriate headers (including the API key) and the file data sent as a multipart request.
3. The response from the API is stored in the *\$result['opswatFile']* array after decoding the *JSON* response body.

```

//Check URL
$virustotalURL = new GuzzleHttp(['base_uri' => $this->virusTotalURL."domains/".$url]);
try {
    $virusUrl = $virustotalURL->request('GET', '', [
        'headers' => ['Accept' => 'application/json', 'X-Apkey' => $this->keyVirus],
    ]);
    $result['virusURL'] = json_decode($virusUrl->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.8 Backend Source Code Block8

4. The sha256 hash value from the response is extracted and stored in the *\$data_idvariable*.

The \$hashvariable is updated with the

Fig A.12 shows the Fourth block of the backend code. This block interacts with the OPSWAT API to check the file hash. It performs the following steps:

1. A new instance of GuzzleHttp is created with the base URI set to the *\$opswatURL* concatenated with "hash/" and the *\$hash* value.
2. A GET request is made to the OPSWAT API for checking the file hash, with the appropriate headers (including the API key).
3. The response from the API is stored in the *\$result['opswatHash']* array after decoding the *JSON* response body.
4. Any exceptions that occur during the request are caught and can be handled accordingly.

Fig A.13 shows the Fourth block of the backend code. This code appears to be handling file uploads to a specific API endpoint using the Guzzle

```

//Check HASH
$virustotalHASH = new GuzzleHttp(['base_uri' => $this->virusTotalURL."search?query=".$hash
try {
    $virusHash = $virustotalHASH->request('GET', '', [
        'headers' => ['Accept' => 'application/json', 'X-Apkey' => $this->keyVirus],
    ]);
    $result['virusHASH'] = json_decode($virusHash->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.9 Backend Source Code Block9

HTTP client. It uploads a file along with a *scan_type* parameter set to 'all'. The response from the API is decoded from JSON and returned as a JSON response.

1. The code starts with a single-line comment */Check FILE*.
2. It creates a new instance of *GuzzleHttp* client with a base URI set to *\$this->hybridURL*.
3. A try block is used to handle exceptions.
4. Inside the try block, a POST request is made using the *\$hybridURL client*.
5. The response from the POST request is decoded from JSON and assigned to *\$result['hybridFILE']*.
6. If an exception occurs within the *try* block, it is caught in the *catch* block, but no specific action is taken.
7. The *\$result* array is checked, and if it's empty, it's assigned *null*.

```

/** OPSWAT */
//Check URL
$opswatURL = new GuzzleHttp(['base_uri' => $this->opswatURL."domain/".$url]);
try {
    $opswatUrl = $opswatURL->request('GET', '', [
        'headers' => ['Accept' => 'application/json', 'apikey' => $this->keyOpswat],
    ]);
    $result['opswatURL'] = json_decode($opswatUrl->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.10 Backend Source Code Block10

```

//Check FILE request 1 (result is data_id)
$opswatFILE = new GuzzleHttp(['base_uri' => $this->opswatURL."file"]);
try {
    $opswatFile = $opswatFILE->request('POST', '', [
        'headers' => [
            'Accept' => 'application/json',
            'apikey' => $this->keyOpswat
        ],
        'multipart' => [
            [
                'name' => 'file',
                'contents' => fopen($file, "r"),
                'filename' => $name
            ]
        ]
    ]);
    $result['opswatFile'] = json_decode($opswatFile->getBody(), true);
    $data_id = $result['opswatFile']['sha256'];
    $hash = (!empty($data_id)) ? $data_id : null;
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.11 Backend Source Code Block11

```

//Check HASH
$opswatHASH = new GuzzleHttp(['base_uri' => $this->opswatURL."hash/".$hash]);
try {
    $opswatHash = $opswatHASH->request('GET', '', [
        'headers' => ['Accept' => 'application/json', 'apikey' => $this->keyOpswat],
    ]);
    $result['opswatHash'] = json_decode($opswatHash->getBody(), true);
} catch (\Throwable $e) {
    // Handle exceptions
}

```

Fig. A.12 Backend Source Code Block12

```

$hybridFILE = $hybridURL->request('POST', '', [
    'headers' => [
        'Accept' => 'application/json',
        'api-key' => $this->keyHybrid,
        'user-agent' => 'app-search'
    ],
    'multipart' => [
        [
            'name' => 'scan_type',
            'contents' => 'all'
        ],
        [
            'name' => 'file',
            'contents' => fopen($file, "r"),
            'filename' => $name
        ]
    ]
]);

```

Fig. A.13 Backend Source Code Block13

Appendix B

Appendix

B.1 Frontend Source Code

```
import DataService from '@services/data.service';
import Component, OnInit, Injectable from '@angular/core';
import FormBuilder, FormGroup, Validators from '@angular/forms';
import Swal from 'sweetalert2/dist/sweetalert2.js';
@Injectable()
@Component({
  selector: 'app-search',
  templateUrl: './search.component.html',
  styleUrls: ['./search.component.scss']
}) export class SearchComponent implements OnInit {
  // Declaration of necessary variables and use in the component
  public loading = false;
  form: FormGroup
  ipAddress = null
  url = null
  hash = null
  result: any
  check: any
  block: any
  virusIp: any
  virusUrl: any
  antiVirus: any
  search: any
```

```

opswat: any
opswatURL: any
virusHash: any
antiVirusHash: any
hybridFILE: any
file: File = null
fullRepport: boolean = false
//Initialization of variables in the constructor, these variables are called first as
soon as
the component is called
constructor(
private formBuild: FormBuilder,
private data: DataService
) {
/** * The function newSearch is the one that builds the form from the form-
Build (native component
of angular * "@angular/forms")
**/
this.newSearch()
}

ngOnInit(): void {
} newSearch() {
/** * Construction of the fumulaire with the application of the different data
validation systems
* avec Validators.pattern de "@angular/forms".
*/
this.form = this.formBuild.group({
ipAddress: [null, [Validators.compose([
Validators.pattern('(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?).)3(?:25[0-5]|2[0-
4][0-9]|[01]?[0-9][0-9]?')
])]],
ip: [null],
network: [null],
hash: [null],
url: [null, [Validators.compose([ Validators.pattern('(https?:)?([

```

```

da-z.-]+)
.([a-z.]2,6)[/
w .-]*/?') ]]], file: [null] }) } /** * Function for sending the form, precisely
the sending of a file
*/ onSubmitFile() {
Function allowing to initialize all the returns of data which were carried out
before
this.onNull()
Verification if the FILE object does indeed exist before any sending
if (this.file) {
Activate the loader so as not to allow the user to click several times and also
make him
wait while waiting
The return of the request
this.loading = true
Explicit sending of data to the Back End from the postFile function allowing
to perform an Http client request
From the client(user) interface to the back-end
this.data.postFile(this.file).subscribe(data => {
Return of the request
We deactivate the loader
this.loading = false
We retrieve the results of the query that brought us
this.result = data
We check if this result has actually returned usable results. if(this.result !==
null) {
We display the results confirmation POP to the client (user)
this.notif('Congratulations', 'Results corresponding to '+this.search, 'suc-
cess') this.hybridFILE = data?.hybridFILE ? data?.hybridFILE : null this.opswat
= data?.opswatHash ? data?.opswatHash : null
this.antiVirus = data.opswatHash

```

?

```

data.opswatHash?.scan_results?.scan_details
: null this.loading = false
}

```

```

else{
  Otherwise we display the confirmation POP of NO results to the customer (user)
  this.notify
  ('Results', 'No results match your search' + this.search, 'warning')
  this.loading = false
}
}, error => {
})
}
else{
  If the FILE object does not exist then we send a POP message to the client (user)
  And we deactivate loader
  this.notify('Warning', 'Please fill in at least one field', 'warning')
  this.loading = false
}
}
onSubmit(){
  Function allowing to initialise all the
  returns of data which were carried out before
  this.onNull()
  Verification if the IP, URL or HASH object exists before any sending
  if(this.ipAddress||this.url||this.hash){
    Activate the loaders so as not to allow the user to click several times and also make him
    wait while waiting
    The return of the request
    this.loading = true
    We format data before sending
    this.form.value.ip = this.ipAddress?this.ipAddress : null
    this.form.value.network = this.ipAddress?this.ipAddress + '/' + 24 : null
    Explicit sending of data to the BackEnd from the getCheck function allowing to
    perform an HttpClient request
    From client (user) interface to back – end
    this.data.getCheck(this.form.value).subscribe(res => {
      Return of the request
      we deactivate the loader
      this.loading = false
    })
  }
}

```

```

We retrieve the results of the query that brought us
this.result = res
if(this.result! == null){
We display the results confirmation POP to the client (user)
this.notify('Congratulations', 'Results corresponding to' + this.search, 'success')
this.opswat = res?.opswatHash?res?.opswatHash : null
this.check = res?.check?res?.check.data : null
this.block = res?.block?res?.block.data : null
this.virusIp = res?.virusIp?res?.virusIp.data : null
this.virusUrl = res?.virusURL?res?.virusURL.data : null
this.virusHash = res?.virusHASH?res?.virusHASH.data[0] : null
this.opswatURL = res?.opswatURL?res?.opswatURL : null
this.antiVirusHash = this.hash?
this.virusHash?.attributes?.last_analysis_results : null if (res.virusIp||res.virusURL){
this.antiVirus = this.ipAddress?this.virusIp?.attributes?.last_analysis_results
: this.virusUrl?.attributes?.last_analysis_results
}elseif(res.opswatHash){
this.antiVirus = this.opswat?.scan_results?.scan_details
}
this.loading = false}else{
Otherwise, the NO results confirmation POP is displayed to the client (user)
this.notify('Results', 'No results match your search' + this.search, 'warning')
this.loading = false
}
}, error => {
this.loading = false
this.notify('Error', 'Error caught', 'error')
})
}else{
If the IP, URL or HASH objects do not exist then a POP message is sent to the client
(user)
And we deactivate loader
this.notify('Warning', 'Please fill in at least one field', 'warning')
this.loading = false
}
}
}

```

Recovery of the IP address and validation before sending

```
onSearchIp(){  
  if(this.ipAddress! == null){  
    this.form.patchValue("url" : null)  
    this.form.patchValue("hash" : null)  
    this.form.patchValue("file" : null)  
    this.search = this.ipAddress  
  }  
}
```

URL recovery and validation before sending

```
onSearchUrl(){if(this.url! == null){  
  this.form.patchValue("ipAddress" : null)  
  this.form.patchValue("hash" : null)  
  this.form.patchValue("file" : null)  
  this.search = this.url  
}  
}
```

Recovery of the hash and validation before sending

```
onSearchHash(){  
  if(this.hash! == null){  
    this.form.patchValue("url" : null)  
    this.form.patchValue("ipAddress" : null)  
    this.form.patchValue("file" : null)  
    this.search = this.hash  
  }  
}
```

Recuperation du fichier et validation avant envoi

```
onSearchFile(files : FileList){  
  if(files.length > 0){  
    const file = files.item(0)  
    this.form.patchValue({"ipAddress" : null})  
    this.form.patchValue({"hash" : null})  
    this.form.patchValue({"url" : null})  
    this.file = file  
    this.search = file.name  
  }  
}
```

```

}
Eventallowingto displaytheinformationcorrespondingtotheclient(user)
onFullRepport(){this.fullRepport =!this.fullRepport}
onNull(){
this.result = null
this.opswat = null
this.check = null
this.block = null
this.virusIp = null
this.virusUrl = null
this.virusHash = null
this.opswatURL = null
this.antiVirusHash = null
this.antiVirus = null
this.hybridFILE = null
}
ImplementationofthePOPnotificationsystem
notif(title,text,type){
Swal.fire({
title : title,
text : text,
icon : type,
timer : 3000,
showConfirmButton : false
})
}
get f(){returnthis.form.controls}
}

```

Appendix C

Appendix

C.1 SOC Incident Creation Playbook

This process ensures that security incidents are identified, documented, prioritised, and addressed systematically, allowing SOC stakeholders to mitigate risks promptly and efficiently. This playbook outlines the steps an analyst should follow when identifying and creating a security incident within the SOC, including escalation procedures based on incident priorities.

C.1.1 Step 1: Identify Potential Incident

Review alerts and notifications from various sources:

- SIEM (Security Information and Event Management) alerts
- IDS/IPS (Intrusion Detection/Prevention System) alerts
- EDR (Endpoint Detection and Response) alerts
- User reports (phishing, suspicious activity, etc.)
- Vulnerability scan findings
- Threat intelligence feeds
- External notifications (ISACs, CERTs, vendors)

C.1.2 Step 2: Triage the Event

- Collect and review available data (logs, alerts, reports)
- Perform initial analysis to identify potential indicators of compromise
- Assess the severity and potential impact of the event
- Gather contextual information (affected systems, users, data, finding answers to what, when, where, who, how)
- Determine if the event warrants escalation to a formal incident

C.1.3 Step 3: Determine Incident Priority

Assign a priority level to the incident based on the following criteria:

- Critical (P1): Incidents involving critical systems, data breaches, widespread impact, or potential regulatory violations
- High (P2): Incidents with significant impact or potential for further compromise
- Medium (P3): Incidents with moderate impact or risks
- Low (P4): Incidents with minimal impact or risks

C.1.4 Step 4: Determine Incident Creation Criteria

Review the incident creation criteria based on the assigned priority:

- P1 (Critical) and P2 (High) incidents should be immediately escalated and formally documented within the required SLA (15 min, 30 min, 1 hour)
- P3 (Medium) incidents may require further analysis before formal documentation (24h SLA)
- P4 (Low) incidents may be monitored and documented at the analyst's discretion (up to 72h SLA)

C.1.5 Step 5: Create Incident Record (for P1, P2, and escalated P3 incidents)

- Document incident summary (description, timeline, affected assets)
- Record initial analysis findings
- Collect and document evidence (logs, network captures, forensic images)
- Generate an Incident Tracking ID from the Incident Management System (e.g., ServiceNow, Jira)

C.1.6 Step 6: Notify and Escalate

For P1 (Critical) incidents:

- Immediately notify (email and Telephone call) the Incident Response Team (IRT), management, and relevant stakeholders
- Escalate to management
- Engage subject matter experts or external resources as required

For P2 (High) incidents:

- immediately notify the IRT and relevant stakeholders
- Escalate to management
- Engage additional resources as required

For P3 (Medium) and P4 (Low) incidents:

- Notify (Email) the IRT and relevant stakeholders as appropriate
- Escalate to management or engage additional resources if needed

C.1.7 Step 7: Transfer to Incident Response Team

For P1 (Critical) and P2 (High) incidents:

- Conduct an urgent incident handover meeting with the IRT

- Share incident documentation and analysis findings
- Provide a briefing on the current situation and potential next steps
- Transfer incident ownership to the IRT for further investigation and response

For escalated P3 (Medium) incidents:

- Schedule an incident handover meeting with the IRT or keep in touch via email
- Share incident documentation and analysis findings
- Provide a briefing on the current situation and potential next steps
- Transfer incident ownership to the IRT for further investigation and response

By following this playbook, analysts can ensure a consistent and structured approach to identifying, triaging, and creating security incidents within the SOC, with appropriate escalation procedures based on the incident's priority level. Proper documentation, notification, and handover processes will be followed, facilitating effective incident management and response.

Appendix D

Appendix

D.1 Scenario 1 Test Results

D.1.1 IP Search Results

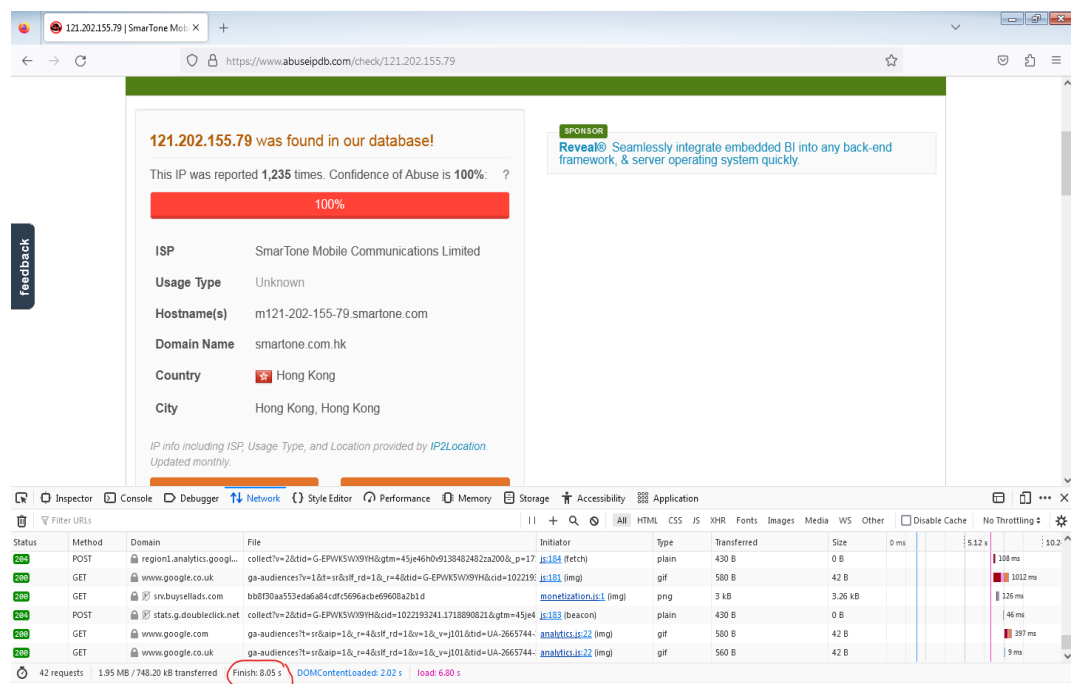


Fig. D.1 Scenario 1 IP Search Results on AbuseIPDB

D.1.2 URL/DOMAIN Search Results

D.1.3 Hash Search Results

D.2 Scenario 2 Test Results

D.2.1 IP Search Results

D.2.2 URL/DOMAIN Search Results

D.2.3 Hash Search Results

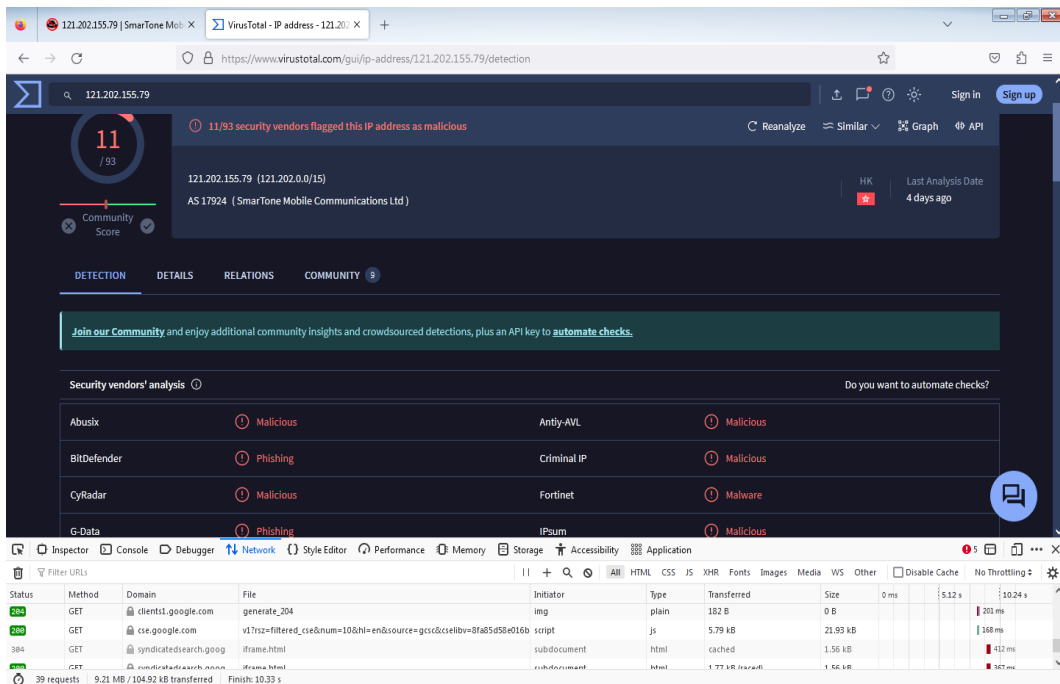


Fig. D.2 Scenario 1 IP Search Results on VirusTotal

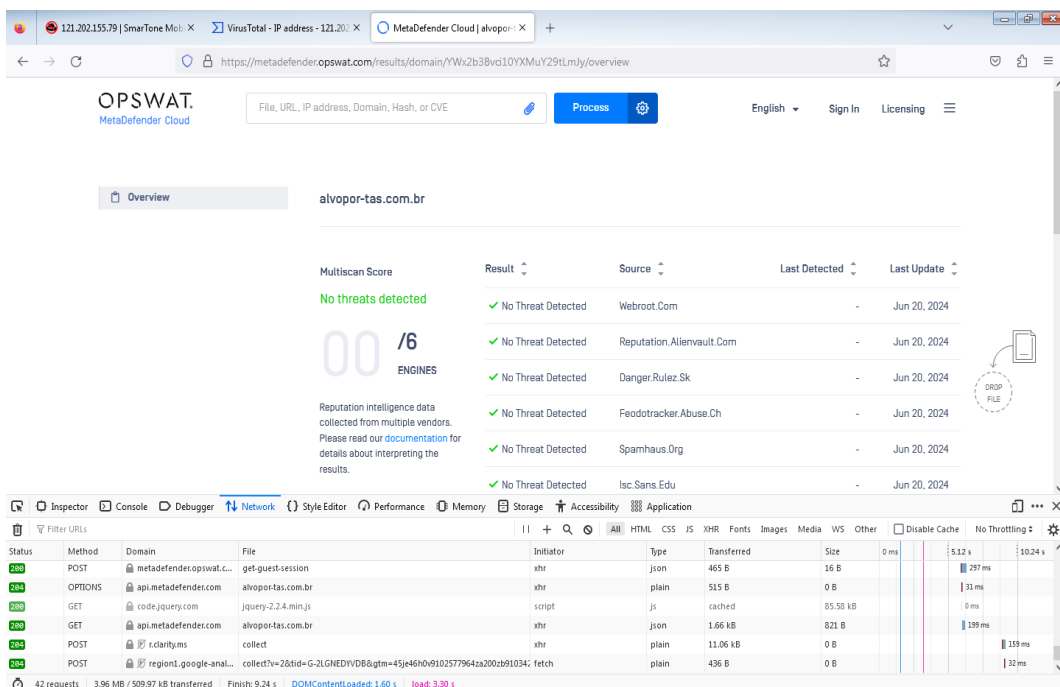


Fig. D.3 Scenario 1 URL/DOMAIN Search Results on MetaDefender

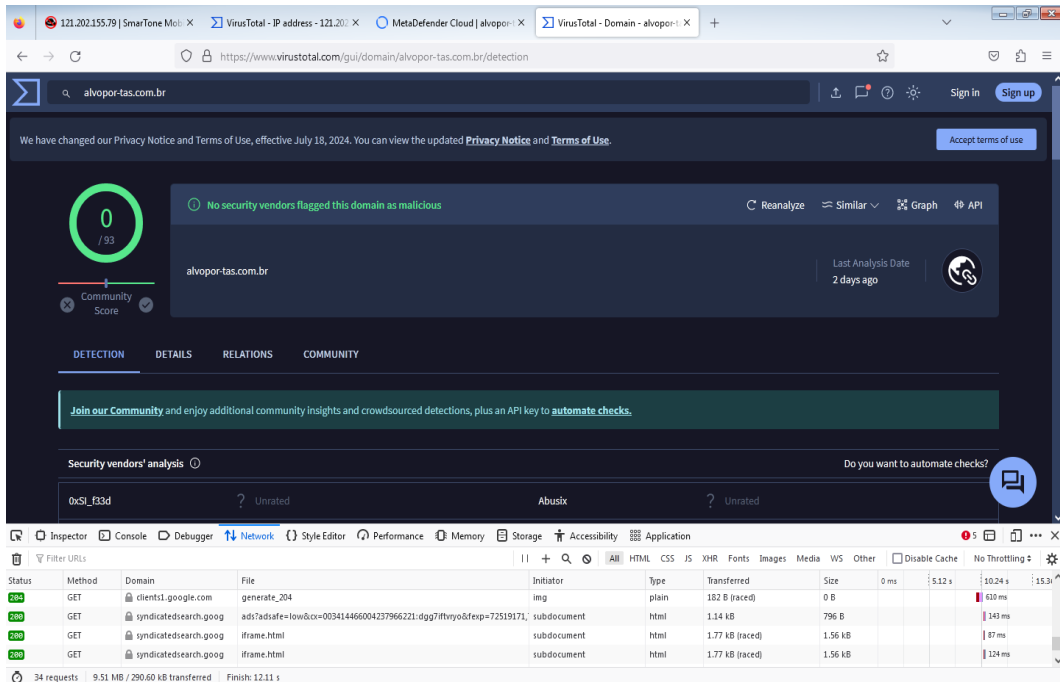


Fig. D.4 Scenario 1 URL/DOMAIN Search Results on VirusTotal

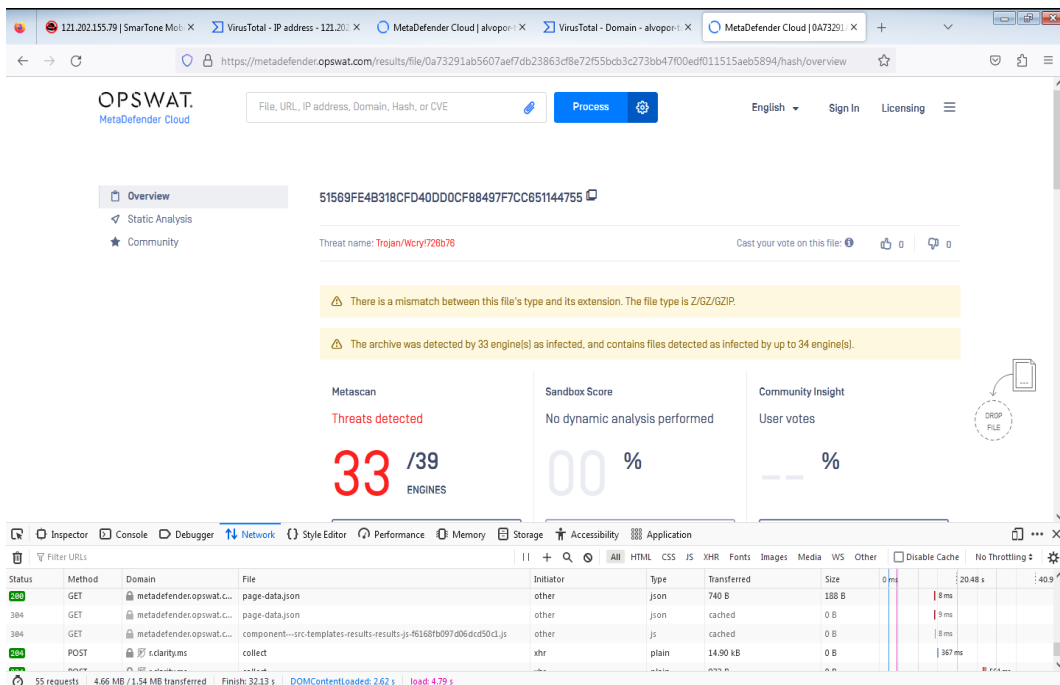


Fig. D.5 Scenario 1 Hash Search Results on MetaDefender

0a73291ab5607aef7db23863cf8e72f55bcb3c273bb47f00edf011515aeb5894

48 / 61

48/61 security vendors and no sandboxes flagged this file as malicious

Reanalyze Similar More

0a73291ab5607aef7db23863cf8e72f55bcb3c273bb47f00edf011515aeb5894

Btr6Le24.exe

Size: 82.35 KB

Last Modification Date: 2 months ago

GZIP

Community Score

DETECTION DETAILS RELATIONS COMMUNITY 18+

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Security vendors' analysis

Do you want to automate checks?

Vendor	Detection	Antiy-AVL	Trojan.Win32.Deshacop
AhnLab-V3	Trojan.Win32.WannaCryptor.R200589		
Arcabit	Trojan.Ransom.WannaCryptor.L	Avast	Win32:WanaCry-A [Trj]
Avert Labs	Ransom-O.g	AWG	Win32:WanaCry-A [Trj]
Avira (no cloud)	TR/FileCoder.724645	BitDefender	Trojan.Ransom.WannaCryptor.L

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Filter URLs

Status	Method	Domain	File	Initiator	Type	Transferred	Size	ms	10.24 s	20.48 s	30.72 s
200	GET	www.virustotal.com	behaviour_mitre_trees	main.0d95fa7b7b104f918a...	json	service worker	181 B	0 ms			
200	GET	www.virustotal.com	votes?relationships=Item,voter	main.0d95fa7b7b104f918a...	json	service worker	11.38 kB	0 ms			

95 requests 17.74 MB / 886.16 kB transferred Finish: 24.74 s DOMContentLoaded: 890 ms load: 12.74 s

Fig. D.6 Scenario 1 Hash Search Results on VirusTotal

MALWARE ANALYSIS

IP ADDRESS URL / DOMAIN FILE HASH UPLOAD FILE

121.202.155.79 ✓ IP ADDRESS Q

121.202.155.79 was found in our database!

AbuseIPdb Check an IP Address

This IP was reported 356 times. Confidence of Abuse is 100%:

ISP : SmartTone Mobile Communications Limited

Usage Type :

Domain Name : smarttone.com.hk

Country : HK

City :

AbuseIPdb blocked check

Max Address : 121.202.155.255

Min Address : 121.202.155.0

Network Mask : 255.255.255.0

Hôtes possibles : 256

City : 121.202.155.255

VirusTotal

DETECTION

Engine	Verdict	Category
ADMINUSLabs	clean ✓	harmless
AlienVault	clean ✓	harmless
Antiy-AVL	malicious ⚠	malicious
AutoShun	unrated ⚡	undetected
BitDefender	phishing	malicious
Blueliv	clean ✓	harmless
MalwarePatrol	clean ✓	harmless
MalwareD	clean ✓	harmless
Netcraft	unrated ⚡	undetected
OpenPhish	clean ✓	harmless
PREBYTES	clean ✓	harmless
PhishLabs	unrated ⚡	undetected

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Filter URLs

Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms	1.28 s	2.56 s	3.8
200	POST	api.researchpurpose.net	check	polyfills.1b4e77b16a9247f...	json	3.47 kB	15.89 kB				2993 ms

One request 15.89 kB / 3.47 kB transferred Finish: 2.99 s

Fig. D.7 Scenario 2 IP Search Results

MALWARE ANALYSIS

IP ADDRESS URL / DOMAIN FILE HASH UPLOAD FILE

alvopor-tas.com.br ✓ URL

Search alvopor-tas.com.br

MetaDefender

Result	Source	Last Detected	Last Update
✔ No Threat Detected	webroot.com	-	Jun 20, 2024
✔ No Threat Detected	reputation.alienvault.com	-	Jun 20, 2024
✔ No Threat Detected	spamhaus.org	-	Jun 20, 2024
✔ No Threat Detected	isc.sans.edu	-	Jun 20, 2024

VirusTotal

DETECTION

ADMINUSLabs	unrated	undetected	MalwarePatrol	unrated	undetected
AlienVault	unrated	undetected	Malward	unrated	undetected
Antiy-AVL	unrated	undetected	Netcraft	unrated	undetected
AutoShun	unrated	undetected	OpenPhish	unrated	undetected
BitDefender	unrated	undetected	PREBYTES	unrated	undetected
Blueliv	unrated	undetected	PhishLabs	unrated	undetected

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Filter URLs

Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms	640 ms	1.28 s	1.92 s
200	POST	api.researchpurpose.net	check	polyfills.1b4ef7b16ad9471...	json	2.18 kB	11.41 kB				

One request 11.41 kB / 2.18 kB transferred Finish: 1.65 s

Fig. D.8 Scenario 2 URL/DOMAIN Search Results

MALWARE ANALYSIS

IP ADDRESS URL / DOMAIN FILE HASH UPLOAD FILE

73291ab5607aef7db23863cf8e72f55bcb3c273bb47f00edf011515aeb5894 HASH Q

48/62

48 engines detected this file

0a73291ab5607aef7db23863cf8e72f55bcb3c273bb47f00edf011515aeb5894

.js

grip contains-pe

VirusTotal

K7AntiVirus Trojan (0001140e1) Zoner Trojan.Win32.55633

K7GW Trojan (0001140e1)

MetaDefender : 51569FE4B318CFD40DD0CF88497F7CC651144755

Threat name: Trojan/Wcry!726b76

Metascan

Threats detected

33 / 39

GET FULL REPORT

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms	1.28 s
200	OPTIONS	api.researchpurpose.net	check	xhr	html	590 B	0 B	172 ms	
200	POST	api.researchpurpose.net	check	pohfills:1b4ef7fb16ad947f...	json	5.20 kB	22.72 kB	1649 ms	

2 requests 22.72 kB / 5.79 kB transferred Finish: 1.83 s

Fig. D.9 Scenario 2 Hash Search Results