UNIVERISTY OF PIRAEUS - DEPARTMENT OF INFORMATICS

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

MSc «Cybersecurity and Data Science"

ΠΜΣ «Κυβερνοασφάλεια και Επιστήμη Δεδομένων»

MSc Thesis

Μεταπτυχιακή Διατριβή

Thesis Title: Τίτλος Διατριβής:	**"Designing a Cybersecurity Training Environment (Cyber Range): A Scalable Approach for Real-World Defense Simulations and Practices"** «Σχεδιασμός περιβάλλοντος εκπαίδευσης Κυβερνοασφάλειας (Cyber Range): Μία κλιμακούμενη προσέγγιση για την ανάπτυξη ρεαλιστικών προσομοιώσεων και πρακτικών άμυνας»
Student's name-surname: Ονοματεπώνυμο φοιτητή:	**Tassis Panagiotis** Τάσσης Παναγιώτης
Father's name: Πατρώνυμο:	**Georgios** Γεώργιος
Student's ID No: Αριθμός Μητρώου:	ΜΠΚΕΔ2341
Supervisor: Επιβλέπων:	**Panayiotis Kotzanikolaou, Professor** Παναγιώτης Κοτζανικολάου, Καθηγητής

Ιούλιος 2026 / July 2026

3-Member Examination Committee

Τριμελής Εξεταστική Επιτροπή

Panayiotis Kotzanikolaou, Professor	**Dimitrios Apostolou Professor**	**Konstantinos Patsakis Grade**
Παναγιώτης Κοτζανικολάου, Καθηγητής	Δημήτριος Αποστόλου Καθηγητής	Κωνσταντίνος Πατσάκης Καθηγητής

Acknowledgements

I would like to express my deepest gratitude to Professor Mr. Panagiotis Kotzanikolaou for his precious help and guidance in the accomplishment of this thesis. I am also very thankful to Mr. Christos Grigoriadis, Ph.D., for his valuable assistance in this effort. The enlightening and brain-storming discussions that we had in the area of cyber security guided me well throughout my work, overcame obstacles and polished my work.

All my gratitude goes to every single of the professors I studied with since secondary school and then until today. It is their teachings that imparted in me the wisdom, belief and capability to reach this significant achievement. Scholarship of science is lifelong, and I hope to further it.

Above all, I am profoundly grateful to my family for their endless patience, encouragement, and understanding throughout this journey. Finally, my deepest appreciation goes to my girlfriend, whose unwavering support, love, and belief in me have been a constant source of motivation and strength. Her understanding and encouragement helped me persevere through the most challenging moments of this journey.

Table of Contents

Abstract

As digitalization extends to every industry, the frequency and volume of cyber-attacks are increasing at a rate never experienced before. Ongoing shortage of cybersecurity professionals in combination with constantly evolving threat vectors illustrates the urgent need for state-of-the-art training facilities that emulate true-to-life cyber environments. It is a conventional technique such as hands-on lab practice, prepared hacking exercises, Capture the Flag (CTF) contests, and practice on virtual machines that add up to skill acquisition. These resources rapidly become obsolete as there are fresh threats emerging every day.

Cyber ranges provide a more dynamic and integrated answer by simulating networks, systems, and apps in realistic but controlled settings. They provide scalable cybersecurity training, education, and testing through the power of testing the effects of emerging threats on their updated clone of their operating environment without risking downtime or leaking sensitive data. Such facilities allow the world of cybersecurity to keep pace with the accelerated development of disruptive technologies and the increased interconnection of digital systems.

This thesis proposes both a **methodology** and an **implementation** for constructing a scalable cyber range through a virtual infrastructure. The proposed approach includes:

1. The design approach for constructing a scalable, automated and replicable cyber range environment, outlining the underlying infrastructure, technologies, procedures used to model realistic attack scenarios essential services.
2. Details of the practical realization of this environment, focusing on the deployment of the Ludus Cyber Range, system architecture, automation tools, and the configuration of a segmented enterprise-like topology.
3. Execution and evaluation of a **Full-Chain Attack Lifecycle** through **adversary emulation and detection validation**, assessing the cyber range's functionality, reliability, and scalability in simulating realistic cyber incidents.

Through such demonstration of a series of procedures, a methodology is presented in this thesis that makes it easy to develop realistic cyber ranges. This methodology can be applied in organizations of either the private or public sectors to build cyber ranges based on their real infrastructure and keep testing them against threats as well as ensuring the reliability of detection capabilities.

SUBJECT AREA: Cyber Range

KEYWORDS: Cyber-threats, cyber-security, simulation platforms, risk analysis, threat forecasting, cyber range, information security, testbed, cyber exercises, training

Περίληψη

Καθώς η ψηφιοποίηση επεκτείνεται σε κάθε κλάδο, η συχνότητα και ο όγκος των κυβερνοεπιθέσεων αυξάνονται με ρυθμούς που δεν έχουν παρατηρηθεί ποτέ στο παρελθόν. Η συνεχιζόμενη έλλειψη εξειδικευμένων επαγγελματιών στον τομέα της κυβερνοασφάλειας, σε συνδυασμό με τη διαρκή εξέλιξη των μεθόδων επίθεσης, καταδεικνύουν επιτακτική την ανάγκη για εκπαιδευτικές υποδομές που να προσομοιώνουν ρεαλιστικά κυβερνοπεριβάλλοντα. Παραδοσιακές τεχνικές, όπως η πρακτική εξάσκηση σε εργαστηριακό περιβάλλον, οι διαγωνισμοί Capture the Flag (CTF) και η εκπαίδευση μέσω εικονικών μηχανών, συμβάλλουν σημαντικά στην απόκτηση δεξιοτήτων. Ωστόσο, αυτοί οι πόροι καθίστανται γρήγορα ξεπερασμένοι, καθώς νέες απειλές εμφανίζονται καθημερινά.
Τα Cyber Ranges προσφέρουν μια ολοκληρωμένη λύση, προσομοιώνοντας δίκτυα, συστήματα και εφαρμογές σε ρεαλιστικά αλλά ελεγχόμενα περιβάλλοντα. Παρέχουν δυνατότητες κατάρτισης και δοκιμών στον τομέα της κυβερνοασφάλειας, επιτρέποντας την αξιολόγηση νέων απειλών σε λειτουργικά περιβάλλοντα, χωρίς τον κίνδυνο διακοπής λειτουργίας ή διαρροής ευαίσθητων δεδομένων.

Η παρούσα διπλωματική εργασία προτείνει τόσο μια μεθοδολογία όσο και μια υλοποίηση για την κατασκευή ενός Cyber Range μέσω μιας εικονικής υποδομής. Η προτεινόμενη προσέγγιση περιλαμβάνει:

1. Τον σχεδιασμό και τη μεθοδολογία για την κατασκευή ενός επεκτάσιμου, αυτοματοποιημένου περιβάλλοντος Cyber Range, με ανάλυση της υποδομής, των τεχνολογιών και των διαδικασιών που χρησιμοποιούνται για τη μοντελοποίηση ρεαλιστικών σεναρίων επιθέσεων και κρίσιμων υπηρεσιών.
2. Την πρακτική υλοποίηση του περιβάλλοντος, με έμφαση στην ανάπτυξη του Ludus Cyber Range, την αρχιτεκτονική του συστήματος, τα εργαλεία αυτοματοποίησης και τη διαμόρφωση της τοπολογίας που προσομοιώνει ένα επιχειρησιακό δίκτυο.
3. Την εκτέλεση και αξιολόγηση ενός Full-Chain Attack Lifecycle μέσω εξομοίωσης αντιπάλου (adversary emulation) και επαλήθευσης ανίχνευσης (detection validation), με στόχο την αποτίμηση της λειτουργικότητας, της αξιοπιστίας και της επεκτασιμότητας του Cyber Range στην προσομοίωση ρεαλιστικών κυβερνοεπιθέσεων.

Με την επίδειξη αυτών των διαδικασιών, η εργασία προτείνει μια δομημένη μεθοδολογία που απλοποιεί την ανάπτυξη ρεαλιστικών Cyber Ranges. Η προσέγγιση αυτή μπορεί να υιοθετηθεί από οργανισμούς τόσο του δημόσιου όσο και του ιδιωτικού τομέα, προκειμένου να κατασκευάζουν Cyber Ranges με την πραγματική τους υποδομή, να τα δοκιμάζουν τακτικά απέναντι σε νέες απειλές και να επαληθεύουν τη λειτουργικότητα των μηχανισμών ανίχνευσης που διαθέτουν.

1. Introduction

1.1 Current State

The evolution of technology and cybersecurity has presently moved its attention toward protecting countries and their international security, as compared to information security. Due to the complexity of protecting digital assets along with sensitive information, effective cybersecurity measures are needed to protect national security systems as well as critical infrastructure systems.

Innovative tactics are adopted to disable critical social infrastructure. Most of these are disruptions to critical networks loss of sensitive data, corruption of operational controls, or total system failure. Previous incidents, such as the Mirai botnet attack, WannaCry ransomware and the Stuxnet worm, all serve the intent of showing active and inventive defensive measures against technological ecosystems, for example, IoT-Based systems in health and energy sector industrial control systems. Along with the wide variety of cyber threats, these cases depict that cybersecurity training and assessment should be routine, responsive, and authentic. [17]

Cyber Ranges (CR) are becoming more of a presence in the world of cybersecurity, as they offer secure and structured environments for training, testing or exploration [13]. Cyber ranges are the terminology that is applied to virtualized environments that mimic real IT infrastructure and network behaviors, from anything in between a large quantity of equipment and configuration to one-way simulation to potentially open public systems that can mimic the Internet. With simulated cyber-attacks, students and security practitioners have the opportunity to gain experience first-hand how to identify potential threats, measure the degree of vulnerability, respond in an effective manner to incidents (and planning for strategy). Apart from that, the training is a hands-on one. Cyber ranges compared to traditional learning platforms have the advantage of using engaging and dynamic training environments replicating actual cybersecurity breaches. They permit education and discovery of cyber-attacks through simulation-based modeling of attack tactics, defense technique and the impact of a breach. They also offer the amenities to execute and experiment with cybersecurity programs face-to-face, along with pedagogy that bridges the theory-practice gap. [3]

The cybersecurity solutions available are multi-directional, spanning across several cyber domains and industries. Elsewhere IoT devices will be responsible for security threats in other areas such as agriculture, marine and emergency services. Sensors, actuators and attached devices are highly vulnerable to cyber-attacks since they are wireless in nature, require extremely high processing powers for operation, no physical protection against attackers; and can be located in open or non-protected environments. The growth and development of computing infrastructures, such as cloud clouds built upon their architecture, have created a very sophisticated cybersecurity environment.

Edge computing has replaced or supplemented traditional data center environments with its centralized management systems, physical security and end-to-end monitoring. The environment within that paradigm is being redefined by this. These architectures are particularly vulnerable to security threats as end-users are exposed to more malicious and insecure environments in these settings where computer resources are scarce. These devices in such settings lack self-protecting abilities, and are vulnerable to cyber-attack intrusions, which necessitate the use of unique cybersecurity controls. The use of cyber ranges as virtual networking and computing infrastructure supports testing, verification and calibration of cybersecurity in a structured framework. Cyber ranges are used for broad scenario testing, such as devices like control systems, IoT devices and edge devices in segmented integrated setups.

It is possible to accurately emulate network segmentation and firewall rules, intrusion detection systems (IDS), and other security aspects using them. In addition, these facilities offer virtual spaces where attackers can watch and capture their attacks to facilitate complete vulnerability tests, attack chains or quick response. While cyber ranges have been enhanced and certified as indispensable resources for cybersecurity training and research, challenges are multifarious and

work is not yet finished. Despite their inherent complexity, cyber range deployments may be highly flexible and versatile in providing accurate simulation as well as useful learning values. Moreover, standardization of cyber range tools and environments, along with efficient automation of vulnerability assessment, forms the cornerstone for the improvement of the effectiveness of cyber spaces in applications as well as academic environments.

To answer these requirements, this thesis proposes an integrated, scalable solution to cyber range planning and deployment, with special emphasis on automation in vulnerability assessment and risk management. Last, the objective is to develop a cybersecurity training environment that can aptly simulate and address tomorrow and today's cybersecurity issues, and therefore aptly equip professionals to respond and counter future cyber threats.

1.2 Definition of Cyber Ranges

A Cyber Range (CR) is an environment dedicated to practicing the skills including personal and technical, of IT security professionals and apprentices by facing hands-on complex networks that are emulated in a manner to resemble existing network infrastructure attacked in a cyber warfare situation. Commonly implemented as a standalone virtual environment, all the required infrastructure is bundled together into a Cyber Range facility, allowing participants to immerse themselves in fully realistic cyber-attack and defense scenarios[13]. incidents with greater competence.

Beyond training purposes, Cyber Ranges are also vital in operational and project support contexts, serving as critical platforms for cyber-warfare training and technological research and development within cybersecurity. These platforms facilitate the testing and enhancement of endpoint prevention, detection and response mechanisms, operational technology (OT) security measures and the execution of attack simulations. By providing tools aimed at reinforcing the stability, security and overall performance of cyber-infrastructures, Cyber Ranges have become a big component in cybersecurity frameworks.

The construction of a cyber range can incorporate a wide variety of components, including:

- Cloud instances and services
- Applications
- SCADA systems and OT components
- Network infrastructure
- Servers
- User endpoints
- Security Information and Event Management (SIEM) Systems
- Intrusion Detection/Prevention Systems (IDS/IPS)
- Firewalls and VPNs
- Active Directory / Identity & Access Management (IAM)
- Threat Intelligence Feeds
- Vulnerability Scanners (e.g., Nessus, OpenVAS)
- Red Team / Blue Team Toolkits
- Simulated Users / Traffic Generators
- Forensics Tools

1.3 Technical Components of Cyber Ranges

A fully functional cyber range relies on several key technical components working in tandem. This section details the virtualized infrastructure, networks, scenario libraries, traffic simulation tools, and target systems that form the backbone of the environment.

1.3.1 Virtualized Infrastructure

Containers and hypervisors are employed within cyber ranges to virtualize enterprise-scale networks. They support realistic topologies with flexibility, scalability and safety. Virtualization also adds a layer of buffering between the target environment and the underlying infrastructure, lowering the risk but making it possible for rapid resets and reconfigurations. While hypervisor-based solutions remain pervasive, software-defined infrastructures are increasingly employed to reduce costs and make cloud extensibility possible.[3]

1.3.2 Underlying Infrastructure

All cyber ranges depend upon their underlying layer of computing, storage, and networking. Others employ physical, rack-based environments (routers, switches, firewalls, endpoints) for highest fidelity but at high cost and with low scalability. To solve these issues, most modern ranges rely upon virtualized or software-defined infrastructure with elastic growth ability and provisioning of private or public cloud resources. An important feature is support for legacy systems, when necessary, since many training scenarios require experience of older but operational technologies.[9]

1.3.3 Scenario Libraries

Pre-configured and customizable scenarios emulate a variety of threats, from phishing to advanced persistent threats (APTs). They provide tested environments for ongoing training as well as stress-testing of infrastructures. In advanced ranges, scenarios can be installed with traffic generation, attack simulation and even client-specific infrastructure profiles for more realism.[9]

1.3.4 Traffic and Data Simulation

Background traffic, user activity and malicious payloads are injected to provide realism. This kind of "digital noise" is needed to train analysts to detect benign and malicious activity[11]. This capability is typically included in the orchestration layer to address the specific learning or exercise objective.

1.3.5 Target Infrastructure

The target infrastructure is the practice environment in which the participants work. Depending on the scenario, it may emulate enterprise stacks of IT or simulate a customer's actual production infrastructure, such as applications, endpoints, servers, and firewalls. By dynamically provisioning accurate configurations (IP address space, routing, endpoint software), cyber ranges can provide highly customized, mission-oriented environments for participants.[9]

1.4 Cyber Range Types

The type of a Cyber Range (CR) is in big way a question of how focused its goals are, the roles participants assume, and the characteristics of participants involved. Normally, CRs belong to one or more of the following types:

- **Educational CRs** have the primary focus on providing structured cybersecurity education and training. These are well-structured environments to study various topics of cybersecurity in addition to giving them practical hands-on experiences.
- **Certification CRs** are utilized for evaluating and certifying officially participants' knowledge and skills in cybersecurity. They are in the form of structured competitions, such as Cyber Defense Exercises (CDX) and Capture the Flag (CTF) competitions.

- **Test CRs** are primarily for research and development, where testing and evaluation of cybersecurity products, technologies, tools and protective measures can be done.
- **Project support** includes evaluating products, testing specific features or capabilities and comparing performance (benchmarking).

Educational and certification CRs involve various stakeholders like educators, cybersecurity experts and trainees or students in cybersecurity training schemes. Trainees are told to observe and guide trainees and step into where they can help learn and solve problems. Technical issues are usually left to cybersecurity experts, who create and establish the CR infrastructure, debug, and intentionally introduce vulnerabilities for training. Users typically communicate individually or in teams, usually organized as some roles within teams, typically as blue or red teams. Blue team members carry out defensive roles, such as either cybersecurity personnel defending systems or forensic examiners analyzing the traces left by cyber-attacks. On the contrary, red team members assume offensive roles, staging mock attacks in order to discover and attack vulnerabilities. During training sessions for defensive purposes, cyber-attacks are conducted by either red teams composed of experts or created from automated sources.

Cyber Ranges are built to serve different types of users, including defense agencies, CERTs, CSIRTs and the NOC or SOC personnel of large service providers and operators of critical infrastructure. A good example is the United States National Cyber Range, built by DARPA[10], who use special techniques to test and improve immunity against advanced cyber-attacks.

Contests like Capture the Flag (CTF) and Cyber Defense Exercises (CDX) are most widely used applications of Cyber Ranges.

- **CTF competitions** focus on offensive skills, which teach people about how the attackers work, use hacking techniques and prepare for possible cyber-attacks.
- **CDX competitions** focus on defensive skills, allowing participants to be prepared to predict attacks and protect systems effectively.

While they may have different goals, both competitions share identical basic benefits: the spread of information, building abilities and testing participants. That is why Cyber Ranges are the best places for them to be held. Several leading Cyber Ranges, such as SPIDER CR, US CR, AIRBUS CR, AIT CR and the University of Delaware CR, hold CTF events regularly. These environments make the competitions more scalable and realistic, showing the use of Cyber Ranges as versatile training and education environments for advanced cybersecurity training.

Cyber Ranges (CRs) may be systematically categorized based on their respective technological implementations. Different CR platforms in academic, commercial, and military domains, categorizing them into three broad groups of technology [3] [13]:

- **Simulation CRs** depend primarily on software to establish networks in virtual form without using any physical network equipment. It has good scalability, flexibility, and cost savings and thus is suitable for low-cost businesses or businesses requiring rapid deployment of numerous scenarios. Since the environments are isolated from physical equipment, resetting, modifying, and aligning them with orchestration levels and cloud infrastructures is easier. But at the expense of lower fidelity and realism for hardware based. For, while simulated systems cannot in all instances accurately replicate the subtlety of actual latency, jitter, or device-specificity, despite that realism simulation CRs remain the preferred choice for learning environments, boot camp training, and high-scale exercises where breadth and scale are issues over best realism.
- **Overlay CRs** incorporate hardware networking gear and computer equipment, such as switches, routers, servers, and firewalls into the training environment. From physical hardware, overlay CRs have more fidelity and realism, enabling users to work with systems in a way that closely approximates real operating infrastructures. This makes them particularly appropriate for high-end training in which on-the-metal hands-on experience with hardware is the issue, e.g., configuring firewalls, adjusting intrusion detection systems, or coping with device-specific quirks. Overlay CRs are more costly, however, with additional costs in hardware buying, maintenance, and lifecycle

management. They do introduce threats associated with the integrity and security of the physical infrastructure because hardware can expose itself to actual malicious payloads when simulating. Owing to this cause, overlay ranges are typically deployed in safe, controlled environments and used in situations where realism comes at the expense of scalability.

- **Emulation CRs** aim to closely simulate the physical infrastructures of real enterprise networks, combining aspects of both simulation and overlay approaches. They afford high fidelity, realism, and repeatability, allowing participants to train in conditions that closely simulate real business operations right down to the very applications, protocols, and system configurations. This makes emulation CRs especially useful for mission-critical operations, APT simulation, and exercises that require advanced forensic or incident-response procedures. The main limitation, though, is cost and scalability because constructing and maintaining completely emulated infrastructures requires massive amounts of resources. To address this, some emulation CRs combine virtualization technologies and shared resources, sacrificing realism for efficiency. While less scalable than simulation CRs, emulation ranges provide the most realistic training experience and are therefore the gold standard for high-risk training and research when accuracy is paramount.

Of these varieties, emulation and simulation CRs are most frequently employed due to their applicability in the real world. Nevertheless, hybrid CRs, which combine two or more technological approaches, are increasingly popular.

Hybrid CRs as the name suggests, hybrid ranges are developed by a customized blend of any one of the aforementioned types. Hybrid ranges bring together many capabilities and features from simulations, overlay ranges and emulation, to create a hybrid environment that meets specific requirements. A good example of a hybrid range that employs several features outlined in this document to provide an entire cyber range experience is the Virginia Cyber Range[15]. And yet another instance of a hybrid range is the European Future Internet.

1.5 Cyber Range Scenarios

Cyber ranges are highly flexible platforms allowing organizations to replicate a wide variety of cyberattack situations, including sophisticated threats such as zero-day attacks and supply chain compromise. Exercises place participants in realistic cyber crisis situations, occasionally with collaboration, high-pressure decision-making under short timeframes to maximize learning. A few scenarios can last days, providing a thorough picture of managing complex incidents from detection to resolution.

Examples of scenarios that are replicated include:

- **Malware attacks:** training course attendees to spot, analyze and respond to common malicious software infections.
- **Insider threats:** training responders to recognize and counter insider risks from inside individuals in the business.
- **Ransomware incidents:** practicing containment, mitigation and recovery strategies following an encryption assault.
- **Critical infrastructure attacks:** modeling targeted attacks on core services and operational systems.
- **Advanced Persistent Threats (APTs):** simulating sustained, evasive attacks by expert attackers to develop countermeasures.
- **Red team vs. blue team exercises:** establishing competition and proficiency through attack (offensive, or attacker) and defense (defensive, or defender) simulations.

- **Purple team training:** combining offensive and defensive roles to better facilitate collaboration and solidify overall security posture.

1.6 Main Objectives of Cyber Ranges

- Security detection and response testing — confirming the effectiveness of existing monitoring and response techniques.
- Threat actor and tactic simulation — replicating adversary behavior to test and improve defensive strategies.
- Live attack scenario training — creating dynamic, interactive attack simulations for team and individual training exercises.
- Blue team vs. red team coordination — enabling collaborative efforts between offensive and defensive teams for the creation of better security measures.
- Synchronizing incident responses aligning people, processes and technology to more effectively coordinate response.
- Preparing for real-world attacks — providing defenders with realistic experience using real data, tools and environments.

1.7 Background and Motivation

The creation of Cyber Range (CR) systems is a new step forward in the practice of cybersecurity. From the beginnings of simple network test beds for finding elementary vulnerabilities, they have matured into complex, realistic systems capable of simulating entire cyber environments. This is in response to the rapidly evolving digital technologies and the increasing sophistication of threats in cyberspace, calling for more dynamic, hands-on and technically advanced training and research environments.

Early CRs looked a lot like traditional network test beds. They were designed to test system strength against known vulnerabilities and include some intrusion detection. These early versions were useful but not well positioned to respond to the quickly growing variety of cyber-attacks. As IT environments became more complicated with cloud computing, mobile devices and Internet of Things (IoT), the demand increased for larger and more realistic cybersecurity environments. This called for the development of modern CRs that could mirror whole network organizational networks. Such environments provide the ability to simulate and analyze safely cyberattacks ranging from basic malware to sophisticated, state-sponsored attacks.

In education, CRs have transformed cybersecurity education by closing the divide between theory and practical, hands-on training. Students and professionals can practice their response to attack in real time, building technical competence, critical thinking and adaptability. They solve complex problems under realistic conditions similar to what they will face in professional practice. CRs are also needed for cybersecurity research. They are utilized by researchers to perform controlled experiments, try new defenses, study malware behavior and test new tools and techniques. They also allow researchers to study technical, policy, ethical and legal considerations together. This makes CRs valuable in improving defensive capabilities as well as informing cybersecurity strategy.

Commercial (proprietary) and open-source CRs have been developed to mimic more realistic and relevant cyber security training. Commercial products like IBM's X-Force Command Cyber Range, Microsoft's Cybersecurity Center and Cisco's Cyber Range offer specialized training for specific organizational needs[15].They are capable of duplicating complex threat environments, enhancing incident response and offering industry-specific knowledge. They are scenario-based and technology-centered, although and therefore may lack flexibility. Some of the open-source solutions, such as the RangeForce CyberSkills Platform and the Cyber Range and Training

Environment (CRATE) [3], offer a more versatile and flexible solution. For these ranges, it is easier to adapt scenarios to organizational needs without adding extra cost. Open-source ranges also foster collaboration within the cybersecurity community, resulting in innovation and knowledge sharing. Given the open and flexible nature of open-source systems, the solutions enable companies to dynamically replicate cyber threats and comprehend the reach of vulnerabilities within their systems.

Current Cyber Range solutions, whether proprietary or open source, typically rely on static infrastructures that react in a pre-programmed way, lessening their capacity to mimic the unpredictability of real attacks. The application of machine learning, though promising, typically is with inaccurate input data or compromised system compatibility, particularly if closed systems or proprietary solutions are employed. Additionally, with increasing complexity, and particularly with the incorporation of Internet of Things (IoT) devices, most Cyber Range solutions do not scale well and become unrealistic. Contributing to this is the unmodular design, use-friendliness as well as the fantastically high costs of hosting and keeping these environments. Most Cyber Ranges are designed to mimic exact situations but are not nimble enough to respond to new emerging threats or infrastructures.

With more sophisticated cyber-attacks, traditional training environments fail to equip cybersecurity practitioners with what they need to counter new attack vectors and zero-day exploits. While some ranges have the potential to attract malicious activity by being used as honeypots, they mostly still lack realistic simulations that mirror the dynamic nature of real-world environments, diminishing their usefulness for research and training. In answer to these deficits, the creation of automation within Cyber Ranges can reduce re-deployment times and improve usability. Automation would also allow for reconfiguration of the environment more easily to enable it to be more responsive to emerging threats.

Besides, application of the Cyber Ranges as highly realistic, simulated laboratory environments holds the potential for providing meaningful information through the capture of malicious traffic and recreation of bespoke attacks on real systems.

These enhanced abilities will allow researchers to monitor attacker activity, detect new vulnerabilities and alert the worldwide cybersecurity community to results to assist in hardening defenses globally. Since the type of cyber threats keeps evolving, it is important to develop more adaptive and extensible Cyber Ranges that can keep pace with the dynamically evolving nature of the world of cybersecurity and provide worth to researchers and professionals on an ongoing basis. As a response to some of these issues, there are some interesting projects like Ludus and Immersive Labs. [18]

1.7.1 Application Domain: The Healthcare Sector

The reference environment implemented in this thesis is deliberately not sector-agnostic. Alongside the Windows Active Directory domain, the range hosts OpenEMR, an open-source Electronic Health Record (EHR) platform used by healthcare providers to manage clinical and administrative patient data [22]. This choice situates the implementation within the healthcare domain, and that decision requires explicit justification, since the sector is legally designated as critical infrastructure and is subject to a dedicated regulatory regime that is examined in Section 1.7.2. Four considerations motivated the selection.

First, criticality. Healthcare is classified under Annex I of Directive (EU) 2022/2555 (NIS2) as a sector of high criticality, placing healthcare providers among the entities subject to the strictest cybersecurity obligations in the European Union [36]. The distinguishing characteristic of the sector is that the consequences of a successful intrusion are not confined to financial loss or reputational damage: ENISA documents that incidents in the health sector result in delays to treatment, cancelled operations and the diversion of patients to other facilities, affecting patients and healthcare professionals alike [38]. The most extensively documented European case is the Conti ransomware attack against the Health Service Executive (HSE) of Ireland in May 2021, which led the organization to shut down all national IT systems and disconnect its healthcare network, resulted in the encryption of approximately 80% of its IT estate, and required recovery

efforts extending beyond four months, during which services across the country reverted to manual, paper-based processes [37]. In this sector, therefore, information security is inseparable from patient safety.

Second, threat exposure. The healthcare sector is among the most intensively targeted domains in Europe. The first sectoral threat landscape published by the European Union Agency for Cybersecurity (ENISA), which analyzed 215 publicly reported incidents in the European Union and neighboring countries, found that ransomware accounted for 54% of observed incidents, that healthcare providers represented 53% of affected entities and hospitals alone 42%, and that patient data, including electronic health records constituted the most frequently targeted asset, at 30% of incidents [38]. The same study reported that 46% of incidents were directed at the data held by health organizations, while only 27% of the surveyed organizations operated a dedicated ransomware defence programme. Particularly relevant to the present work, 80% of the healthcare organizations surveyed declared that more than 61% of their security incidents originated from exploitable vulnerabilities [38]. A cyber range that reproduces a vulnerable clinical application therefore addresses the dominant, empirically documented initial-access vector of the sector rather than a hypothetical one.

Third, technical representativeness. The information technology estate of a healthcare provider is characteristically hybrid: an identity and access management backbone built on Windows Active Directory coexists with web-facing clinical applications frequently deployed on Linux hosts. This composition is precisely what the topology described in Chapter 3 reproduces, and it is what allows a deliberately small environment to exercise two distinct and complementary classes of attack, identity-centric attacks against the domain, presented in Sections 4.4 and 4.5, and application-layer exploitation of the clinical system, presented in Section 4.13. The selection of the healthcare domain is thus not incidental to the design: it is the reason a minimal three-node topology can remain operationally realistic while covering the full attack surface of an enterprise environment.

Healthcare environments are widely acknowledged to operate legacy and infrequently patched software, a condition arising from continuous-availability requirements, dependencies on certified clinical software, and constrained maintenance windows. The deliberate deployment of OpenEMR version 5.0.1, affected by CVE-2018-17179, therefore reproduces a realistic rather than an artificial condition, and is consistent with the ENISA finding cited above regarding the predominance of vulnerability-driven incidents. In addition, the use of a documented and stably reproducible vulnerability is a methodological requirement for a training environment since exercises must yield identical outcomes across repeated executions.

A delimitation must be stated explicitly. The environment models the information technology layer of a healthcare provider, the electronic health record system and the surrounding enterprise domain infrastructure. It does not model connected medical devices, clinical engineering networks, or hospital operational technology, which are governed by a distinct regulatory and standardization framework, including Regulation (EU) 2017/745 on medical devices [43] and the IEC 80001 series on risk management for IT networks incorporating medical devices [39]. The extension of the range towards medical device and operational technology emulation is identified as future work in Chapter 5.

1.7.2 Regulatory Context: The NIS2 Directive

Because the implemented environment models a healthcare provider, the regulatory framework applicable to that sector defines the operational requirements that such an organization must satisfy, and consequently the requirements that a training and validation environment should help address. Directive (EU) 2022/2555, commonly referred to as the NIS2 Directive, was adopted on 14 December 2022, repealed the earlier Directive (EU) 2016/1148, and had to be transposed by the Member States by 17 October 2024 [36]. It establishes a harmonized framework for cybersecurity risk management, incident reporting, supervision and enforcement across eighteen sectors.

Healthcare is listed in Annex I of the Directive among the sectors of high criticality, together with energy, transport, banking, drinking water and digital infrastructure. Entities operating in Annex I sectors that meet the size thresholds of the Directive are classified as essential entities and are consequently subject to the most demanding supervisory regime, which includes ex-ante supervision through regular audits, on-site inspections and security scans, and to administrative fines of a maximum of at least EUR 10 000 000 or 2% of total worldwide annual turnover, whichever is higher, pursuant to Article 34 [36]. The Directive further establishes accountability at the level of management bodies, which under Article 20 are required to approve the cybersecurity risk-management measures, to supervise their implementation, and to undergo cybersecurity training themselves.

In Greece, the Directive was transposed by Law 5160/2024 (Government Gazette A' 195, 27 November 2024), which places Greece, according to the National Cybersecurity Authority, among the first Member States to complete transposition [40]. The Law designates the National Cybersecurity Authority as the competent authority, provides for the compilation of the national register of essential and important entities, and was subsequently supplemented by Joint Ministerial Decision 1689/2025 (Government Gazette B' 2186, 6 May 2025), which established the National Cybersecurity Requirements Framework [41]. Healthcare providers operating in Greece that fall within the scope of the Law are therefore subject to a legally binding and enforceable set of cybersecurity obligations.

1.8 Challenges

Despite the progress in the development and deployment of Cyber Range (CR) systems, various challenges still limit their full potential to aid cybersecurity education, training and research. All of these challenges are related to scalability, flexibility and integration with existing technological and educational infrastructures. The overcoming these challenges is key to making CR systems more efficient, accessible and widely used.

1.8.1 Scalability constraints

Increasing threats and complexity require CR tools to scale, which includes more users, larger simulations, and greater realism. The current CR systems are not capable of serving large quantities of users at once or accurately depicting complex, extensive cyber environments. However, this lack of scale prevents schools and research centers from providing broad, experiential training on cyber incidents or high-fidelity simulations of such massive scale cyber-attacks. It is also a challenge to test network resilience in adverse conditions and adequately train cybersecurity professionals for actual pressure.[3] [14]

1.8.2 Adaptability limitations

Cybersecurity rapidly changes, with new tools, threats, and standards developing all the time. Training platforms have to stay up to speed. Too many CRs are, however, founded on inflexible architectures that do not allow for quick upgrading or flexible adjustment of simulation scenarios. Therefore, practice exercises may become outdated or obsolete and no longer be effective. If CRs are not able to adapt new attack methodologies or defense mechanisms, they do not effectively prepare professionals against current and emerging cyber threats. [7]

1.8.3 Integration

The majority of CR systems are independent of school and organizational tools and systems. They do not become integrated with Learning Management Systems (LMS), organizational IT networks, or other instructional technology. Isolation of CRs reduces their ease of use, limits accessibility and makes it difficult for instructors to include practical cybersecurity practice exercises as part of standard coursework. Integration is lost without which wonderful learning opportunities in hands-on experimentation go wasted. To address these challenges, CR systems must evolve. They must be flexible, scalable and tightly integrated architecture capable of simulating a high diversity of cyber-attacks with high accuracy and support multiple users concurrently without a performance degradation. They also must be adaptive and able to rapidly refresh and customize scenarios as new cyber threats and defense methods emerge. Lastly, they should provide high interoperability so that they can be easily integrated with the current learning and technology resources to enhance usability and learning effect.

These issues will require resolution through collaborative work from developers, researchers, and educators to develop and simplify the design of CR platforms. Ensuring future CRs keep pace with the swiftly changing needs of cybersecurity education and operational readiness will allow the cybersecurity community to effectively train the next generation of professionals while continuing to support overall cyber defense efforts.

1.9 Solutions

For the purpose of resolving scalability, flexibility and integration issues in Cyber Range (CR) systems, some recent research and industry efforts have suggested several potential solutions:

1.9.1 Increasing scalability through virtualization and automation

Shifts to software-based virtual infrastructures allow CRs to scale effectively, managing mass groups of users and complex scenarios without the requirement of much physical equipment. It lowers costs and facilitates more dynamic training environments. Additionally, automated testing of training drills, especially for Blue Teams, keeps manual usage at a minimum. By way of reports and pre-configured databases, CRs make uniform, scalable feedback accessible, improving the entire training experience.

1.9.2 Improving flexibility with modular construction and AI integration

Building CRs with modular architecture facilitates rapid upgrading and customization of training environments and facilitates easy content currency as new threats emerge. Modularity makes it possible to add or modify components without the need to rebuild the system. This is complimented by the addition of Artificial Intelligence (AI) and Machine Learning (ML), which enable simulations of advanced cyber-attacks and testing of defense systems. The software facilitates training environments that can evolve in tandem with the threat horizon.

1.9.3 Enabling integration with existing frameworks

For the CRs to integrate successfully with existing technology and learning infrastructure, they must be designed with interoperability. Interoperability with Learning Management Systems (LMS) and other educational technology facilitates smoother training. The employment of standardized scenario description languages and operation protocols allows for easier integration, rendering the CRs more capable of being used in various organizational contexts.

1.10 Contribution

This dissertation offers an in-depth review of the existing Cyber Range (CR) systems within academic, government, military and private institutions. Drawing on the CR environments, simulated networks, training capabilities and operational features, the research seeks to improve our understanding of how CRs can be used to boost cybersecurity readiness. The study considers significant functionalities like scenario generation customized to the individual, identification of vulnerabilities, implementation of defense mechanisms and using various tools and platforms to

reduce cyber threats. All these broad analyses reveal gaps and opportunities present for the development of more advanced CR architecture.

One significant contribution of this research is the formulation of a novel, lightweight, container-based CR architecture. Innovatively crafted to be adaptable, scalable and easy to install, it surmounts much of the limitation inbuilt into current CR solutions. Its design allows real-time discovery of threats, response training and vulnerability assessment within a highly modular system that can quickly adapt to defeat emerging cybersecurity threats. With the help of modern virtualization techniques and open-source cloud platforms, the system offers scalable and affordable and highly customizable security research and training capabilities.

The dissertation also includes in-depth analyses of the hardware and software demands, virtualization platforms and network structures. These analyses are a step-by-step process towards the building of CR environments that precisely replicate actual operational networks. The process involves network configuration, installation of virtual machines, installation of software, and integration of security policies. It also involves the way to select dependable, compatible and inexpensive hardware and software essential considerations for implementing CRs in actual organizational networks.

Automation is the other key feature of this work. With automated redeployment and reconfiguration of environments, the proposed CR framework reduces the human intervention, improves responsiveness, and decreases downtime. Dynamic updates based on evolving threats are achieved by the automation, enabling training environments to remain up to date and effective. The modular design also facilitates smooth integration of extra components such as virtual machines, IoT segments, or advanced monitoring systems, and it becomes possible to simulate hard and varied infrastructures.

The platform also includes advanced monitoring and threat simulation capabilities. It supports realistic attack procedures, including penetration testing, vulnerability scanning and live threat simulation, for accurate measurement of organizational resilience. Though it is not its main focus, the platform can optimize penetration testing, producing detailed data worth using in intrusion detection as well as response analysis. This makes it possible for organizations to detect vulnerabilities in a secure manner and refine their defenses within a simulated environment.

Scalability problems, especially in cases of IoT devices and complex network topologies, are also addressed. The design supports large distributions without performance or simulation realism compromise, ensuring functionality with growing network sizes and complexities. This puts the CR to position both small and large organizations in accordance with the randomness and complexity of modern cyber threats.

This study provides theoretical and practical contributions to designing and applying Cyber Range. With its resolutions of current bottlenecks and introduction of innovations such as automation, modularity and better testing capability, the dissertation proposes a robust, adaptable and scalable Cyber Range architecture for modern-day cybersecurity training, testing and research.

1.11 Methodology and Approach

To achieve the objectives of this thesis, a structured methodology is followed that progresses from the definition of training goals, through the design and population of a representative environment, to the execution and validation of a Full-Chain attack simulation. The methodology is organized into a series of logical, interdependent phases that together enhance maintainability, replicability and scalability of the Cyber Range. It integrates both technical and operational considerations, encompassing infrastructure design, automated provisioning, logging and monitoring integration, attack emulation and detection validation.

1.11.1 Define Objectives

The first phase is to identify the key objectives of the Cyber Range, with a focus on simulating every stage of a realistic attack lifecycle and validating the corresponding defensive measures.

Targets include a controlled, repeatable environment for adversarial emulation, detection engineering and impact analysis. The core platform selected for this purpose is Ludus [18], which is modular, scalable and reproducible. Ludus enables the operation of multiple concurrent training environments and supports snapshot-based restoration for iterative testing, allowing simultaneous training sessions to run without impact on production environments.

1.11.2 Design the Core Network Topology

Our environment is built on Ludus' Basic Active Directory (AD) Network template [19], comprising a Windows Domain Controller, a Windows 11 enterprise workstation, a Kali Linux attacker node, and additional service nodes such as OpenEMR [22].

The detailed specifications, VLAN configuration, and firewall rules are presented in **Section 1.12**.

1.11.3 Prepare Active Directory environment

To move the lab from a clean, out-of-the-box installation to an environment that reflects a real-world enterprise network, the Active Directory domain is populated with a realistic assortment of users, groups, service accounts and devices using BadBlood [29]. This enriches the internal reconnaissance phase with real-world data without requiring additional VM instances, providing realistic enumeration opportunities while keeping the infrastructure footprint minimal.

1.11.4 Implement Security Controls and Misconfigurations

Construct artificial configurations to imitate deprecated or insecure real-world conditions, such as service accounts with higher privileges, administrative interfaces with default credentials and accessible but poorly secured applications (such as OpenEMR). Potential compromises are made by these vulnerabilities.

1.11.5 Execute Full-Chain Attack Lifecycle

Conduct an adversarial simulation from start to finish with a structured kill chain:

- **Initial Access**: Compromise an exposed service (e.g., OpenEMR) as an external foothold.
- **Privilege Escalation**: Gain elevated privileges on the compromised Linux host.
- **Internal Reconnaissance**: Enumerate the AD environment populated with BadBlood, identifying users, groups and trust relationships.
- **Lateral Movement**: Pivot to Windows machines using discovered credentials or misconfigurations.
- **Impact**: Simulate data exfiltration or sensitive information extraction to demonstrate potential business impact.

1.11.6 Validate Detection, Response and Scalability

The recorded attack steps are linked to a centralized SIEM platform to verify detection coverage and evaluate incident response methods. Telemetry, event correlation and visibility across the virtual infrastructure are achieved using centralized logging and monitoring technologies such as Wazuh [23], which enable effective identification of host- and network-based events and support the iterative refinement of detection rules.

1.12 Cyber Range components

To demonstrate the proposed methodology, we design a representative enterprise environment. Below we outline the role and configuration of each system component deployed in this setup.

1.12.1 Active Directory Domain Controller (JD-DC01-2022)

A Windows Server 2022 instance acts as the primary Domain Controller for the ludus.domain domain and forms the backbone of the simulated enterprise network. It provides user authentication, domain group policy management and centralized access control, enabling a

realistic imitation of an enterprise identity infrastructure. This configuration supports a wide range of identity-based attacks, such as Kerberoasting and Pass-the-Hash, as well as privilege escalation and lateral movement across networked systems. By reproducing domain controller behavior in a high-fidelity setting, it becomes a fundamental element of the Cyber Range for both red and blue team training scenarios.

1.12.2 Windows 11. Enterprise Workstation (JD-WIN11-22H2-1)

This workstation serves as an endpoint that is linked to the domain of another device and provides a typical user experience within an organization. Featuring key applications like Office 2019 (64-bit), Chrome, Firefox, Burp Suite, VS Code, 7-Zip, ILSpy and Process Hacker, it is well-suited for conducting client-side exploitations as well as lateral movement simulations and post-exploitation analysis. It is designed to be akin to a modern working environment with the potential for phishing, malicious macros and insider threats.

1.12.3 Kali Linux Attacker Machine (JD-Kali)

The Kali Linux VM operates in an isolated VLAN and acts as the enemy system during red team operations. It has full access to the Windows environment (VLAN 10) on all ports and protocols, allowing for a diverse range of attack simulations. The Windows systems communicate with Kali exclusively on ports 80, 443, and 8080, indicating a controlled exposure typical of segmented enterprise environments. The presence of controlled asymmetry is necessary to simulate common network constraints while still providing complete offensive tooling for the attacker node.

1.12.4 Network Segmentation and Firewall Rules

By using VLANs (10 for Windows systems and 99 for the Kali system), the virtual network is logically divided into segments, with a default policy of REJECT between each Vlan. Custom firewall rules ensure that Windows systems can only communicate with the Kali machine over HTTP/S and other web service ports (80, 443, 8080), while the virtual machine maintains full outbound reachability. The segmentation is based on real-world DMZ and intranet models, which allow for limited ingress traffic but provide attackers with more advanced outbound capabilities.

1.12.5 Infrastructure Automation and Template-Based Deployment

Ludus templates, which are deployed with each component of the infrastructure, ensure consistency and reproducibility across different training sessions and users. A modular YAML schema is used to define configuration parameters, including IP assignment, resource allocation (CPU/RAM), and domain roles, in line with best practices in infrastructure-as-code (IaC) methodologies.

1.12.6 SIEM: Security Information and Event Management

There is a centralized SIEM platform Wazuh [23] which is included in the configuration to process security events generated on each individual node. It allows gaining insight into attack methods and helps in fine-tuning the logic of detection and alerts in order to form a closed loop of attack-emulation and testing of defense mechanisms.

1.12.7 Nested Virtualization and Portability

To achieve better portability and manageability of the solution, the technique of nested virtualization was applied. With help of snapshotting of the virtualized host, the full system configuration with all processes and interaction can be backed up and stored as a unified whole. In this way, not only is the possibility of backing up the Cyber Range significantly improved, but it also becomes possible to transfer the environment to a new physical hardware altogether with no downtime. Furthermore, each component of the solution can be individually exported and reused independently of the rest, which means that it is possible to create a completely new setup using some components of this Cyber Range.

1.13 Structure of Thesis

This thesis consists of five chapters, each contributing to the systematic research and attainment of a Cyber Range for cybersecurity training and simulation. The structure is that of following the natural order from theoretical principles to practical application and evaluation.

Chapter 1: Introduction

This chapter introduces the topic of cybersecurity training, the rationale and background of the research, the research problem definition, and the key contributions. It also covers the methodology adopted, the Cyber Range components, the design principles followed, and the development strategy of the Cyber Range.

Chapter 2: Related Work

The related work chapter discusses existing cyber range projects and platforms, the building blocks such as tools and frameworks, and research from related academic papers.

Chapter 3: Implementation

This chapter details the technical implementation of the proposed Ludus Cyber Range used for this research, describing the system architecture, the virtualization and orchestration stack (e.g., Proxmox), automation mechanisms (Ansible), and threat-scenario implementations. It explains how the Basic Active Directory (AD) Network template and a segmented enterprise-like topology were deployed and instrumented and how automated deployment commands and Ansible roles were used to create reproducible labs.

Diagrams, bash/CLI snippets and small configuration examples are provided to support explanations of system components, configurations and outcomes. The core research objective executing a Full-Chain attack lifecycle (recon > initial access > escalation > lateral movement > persistence > exfiltration) is modelled as repeatable, contained scenarios run inside the controlled lab.

Chapter 4: Testing

This chapter details the practical testing phase, where we populate a simulated Active Directory environment using GitHub repos to create a realistic structure with vulnerabilities. We utilize enumeration tools to identify attack paths and subsequently execute specific attack vectors to the Active Directory environment, while simultaneously implementing detection mechanisms to identify and monitor these malicious activities.

Chapter 5: Conclusions and Future Work

This final chapter summarizes the key outcomes of the study, reviewing the decisions made concerning the design and development process of the thesis as well as the lessons learned from the implementation and testing of the Cyber Range. This chapter also revisits the initial problems of scalability, flexibility, and integration that were raised and discusses how these were handled. Finally, this chapter outlines further avenues of exploration for future studies, such as cluster and cloud configurations, expansion of scenarios outside Active Directory, integration with SIEM and detection engineering, automation of threat emulation, and machine learning applications.

2. Related Work

The growing frequency, complexity and automation of cyberattacks have generated a pressing need for more sophisticated and flexible cybersecurity training techniques. Modern, developing attacks require professionals to defend against threats for which conventional, classroom-based learning and theoretical models are insufficient. Cyber Ranges (CRs) have become essential

tools for cybersecurity training and workforce growth to tackle this challenge. In safe, regulated, repeatable architectures replicating Information Technology (IT) and Operational Technology, these settings provide hands-on experience with actual-world threat situations.

Moreover, some Cyber Range initiatives are offered as vital instruments for testing, training, and reinforcing security measures in many settings. The broad range of methods and applications inside the cyber range scene is shown by these projects, each covering particular elements of cybersecurity training, simulation and system deployment. Together they draw attention to the range and complexity of approaches employed in the creation of contemporary Cyber Ranges.

2.1 Review of Cyber Range Platforms

Increased incidences, severity and usage of automation by threats made it imperative to better and adaptive learning in cybersecurity for coping with threats. Theory and traditional classroom-only sessions are insufficient to train personnel for threat defense in the new dynamically evolving threats. To mitigate this, Cyber Ranges (CRs) have emerged as infrastructure pillars for education and talent development in cybersecurity. These settings provide immediate interaction with realistic threat scenarios in protected, controlled and reproducible environments that are capable of replicating both Information Technology (IT) and Operational Technology (OT) environments. The power is derived from their capacity to create dynamic, interactive training experiences closely akin to actual cybersecurity attacks. Below we will present an overview of architecture and most features of a next-generation Cyber Range. A review of existing CR platforms reveals a wide range of development approaches, each motivated by particular design considerations. The construction of a Cyber Range is influenced by considerations such as flexibility, scalability, isolation, effectiveness, accessibility and testing mechanisms, as well as capabilities for risk assessment. Furthermore, Cyber Ranges can be categorized based on their primary area of use academic, military, or commercial and based on the specific capabilities for which they are designed to be used.

Five Cyber Range environments are discussed in detail in this section, highlighting their mission goals, supported features and relative strengths. Besides that, we will introduce a few Cyber Range projects also as tools for training, testing and improvement of security measures in various surroundings. These projects show the variety of approaches and implementations in the field of cyber ranges, all tailored to fulfill various elements of cybersecurity training, simulation and system installment. Generally, they refer to the scalability and variety of methods used in the establishment of Cyber Ranges in the modern digital world.

2.1.1 Department of Defense Cyber Security Range

Under the 2009 Comprehensive National Cyber Security Initiative (CNCI), the Department of Defense (DoD) Cyber Security Range (CSR) is housed in Stafford, Virginia, close to the Quantico Marine Corps Base. Including the U.S. Marine Corps, Army, Navy, Air Force, among other military and government bodies, the DoD CSR has assisted a great number of missions. Among others, the National Security Agency (NSA) and the Office of the Secretary of Defense (OSD). By replicating real-world network settings inside a restricted and safe infrastructure, it provides a strong educational and training platform.

Operable entirely within a virtualized system created on VMware vSphere version 5.1 and managed via VMware vCenter, the DoD CSR's core architecture has over 3,000 virtual machine profiles to allow for dynamic deployment and fast provisioning. The system employs Tintri VMstore models T540 and T650, arranged into three distinct storage networks with linked backup arrays to guarantee effective power use and dependable backup. Organizing all hardware resources falls under the virtual environment's job.

The three fundamental ideas underlying the DoD CSR's architectural structure are:

- Manages objects for virtualization resources.
- Event Authoring: A visual interface allowing drag-and-drop setup of network topologies.
- Event Orchestration: streamlines the administration of all virtualized resources inside the cyber range.

Through the Joint Regional Security Stack (JRSS), a whole range of tools providing firewall capabilities, intrusion detection and prevention, enterprise-level management, virtual range of network security features as well as routing and forwarder (VRF). Designed to function as independent sandboxes, the environment is totally digital and lacks native cloud capability. Its design has a dual-layer structure: the orchestration engine and physical hardware are integrated at the top level; lower layers offer Virtual Desktop Interfaces (VDIs), coupled to several outside security programs. [10] [15]

2.1.2 National Cyber Range (NCR)

The National Cyber Range (NCR), operated by the Test Resource Management Center (TRMC), is a comprehensive platform supporting realistic cybersecurity testing, evaluation and training. Its architecture leveraging a shared pool of hardware resources to create isolated and secure virtual environments. These environments are orchestrated via a Cyber Range Management Portal, which automates testbed creation. The NCR also incorporates encapsulation and automation toolkits to streamline the deployment and monitoring of simulations.

The NCR is structured around four primary components:

Secure Facility: Includes operator control rooms, the range itself, operations centers, the range support center and the central infrastructure data center, forming the physical and logistical foundation of range operations.

Network Encapsulation Architecture and Operational Procedures

- Utilizes a shared pool of hardware and software resources.
- Employs test specification tools to define the full scope of testing scenarios.
- Automates hardware and software allocation and configuration.
- Execute test scenarios while collecting monitoring and evaluation data.
- Reclaims and reallocates resources back to the shared pool for reuse.

Integrated Software Testing Tool Suite: Supports the rapid and reliable creation of test environments through automation. Test scientists define test specifications, the operations team allocates and manages required resources and participants engage in structured test scenarios using traffic generators, sensors, data analysis and visualization tools. The suite also provides test management verification controls and an event execution language to automate the building, validation and sanitization of test environments.

Cybersecurity Training and Cyber Test Team: Provides dedicated support for cybersecurity training through comparative analysis and research into emerging cyber range usage trends. The Cyber Test Team plays a key role in the design and execution phases, including the development of threat vectors, generation of custom traffic, analysis of test data, provision of hardware and software support and execution of simulation exercises such as Red and Blue Team operations.[10]

2.1.3 Cisco Cyber Range

Designed to equip attendees with the skills necessary to combat current cyber dangers, the Cisco Cyber Range is a service-based training platform. Founded on actual, war-gaming scenarios, it lets students play attacker and defender roles. This dual-perspective approach lets participants learn how weaknesses are taken advantage of while simultaneously obtaining practical experience with defensive measures and tactics for tackling complex attacks, including Advanced Persistent Threats (APTs). Hosted in the cloud, the Cisco Cyber Range enables remote access for flexible training since it duplicates the network and application environment of a standard business. It includes over fifty attack scenarios and can be used for over one hundred actual applications. The platform incorporates several security solutions as well as instruction in operational protocols, security techniques, and use of industry-standard solutions. Built into the training activities are Cisco's own technologies such as Splunk, Stealthwatch, TrustSec, firewalls and intrusion detection systems (IDS), which provide participants with actual experience with the

security ecosystem of Cisco. Furthermore, the Cisco Cyber Range can be incorporated with other Cisco security classes, network firewall setup and intrusion detection included to provide a solid base, for training in cybersecurity and professional growth.[15]

2.1.4 IBM Cyber Range

Hosted within the IBM X-Force Command Center, the IBM Cyber Range offers end-to-end training experience beyond technical teams and offers realistic simulations of actual cyber-attacks. IBM's solution spans across various areas like Security Operations Centers (SOC), Human Resources (HR), and Legal, while the majority of cyber ranges only concentrate on defense techniques teams, gathering them together for coordinated incident response training. The IBM Cyber Range Blue vs. Red Team training approach involves students with a broad spectrum of cyberattack vectors. The training is created to facilitate organizational resilience, enable interdepartmental cooperation and develop a coordinated response plan. Its primary objective is to equip participants with the skills to face the real-world operational challenges by simulating the pressure, decision-making and accountability that are faced in real-world cyber threats. IBM Cyber Range distinguishes itself through its emphasis on the procedural and strategic elements of incident response. It promotes cyber wargaming exercises designed to advance team communication, coordination, and decision-making. IBM's offer is presented here in the context of a stand-alone training facility, particularly given its executive-level involvement and company-wide preparedness emphasis.[15]

2.1.5 Virginia Cyber Range

The Virginia Cyber Range is an internet-based cloud platform whose purpose is to improve cybersecurity education in Virginia high schools and colleges. The primary purpose is to prepare students more adequately for entering the cybersecurity workforce in operations, development and research. By allowing experiential learning in a secure, stand-alone virtual environment, it bridges the gap between theory and experience. One of its primary characteristics is a sandbox setup that allows for interactive security training for different levels of skills. Its cloud architecture offers flexibility, scalability, and economy in the form of dynamic creation and removal of virtual environments based on requirements. This optimizes utilization of infrastructure at strategic expense. The platform also features role-based access control and user-specific content to support structured, individualized learning. It can model large-scale target networks, which support multiple training sessions, Capture the Flag (CTF) challenges and complex exercises with numerous users or teams at once. Together, these capabilities make the Virginia Cyber Range an affordable, scalable teaching tool for producing future cybersecurity specialists.[15]

2.2 Summary of Cyber Range review

The following table demonstrates the summary of the CR review focusing on the CR goals and mission, their capabilities and advantages.

Platforms	Mission	Capabilties	Advantages
DoD Cyber Security Range	Environment to support exercises, training, testing, evaluation and education for the military	Traffic generator, configurable user emulation, malware, spyware and botnets emulation	Cyber-security and computer network defence
National Cyber Range (NCR)	Realistic environment for cyber-training and support for cyber-testing complex governmental systems	Secure facility, role-based access, focus on cyber-security and computer network defence	Malware analysis, forensic analysis, architecture analysis, training events, research, Capture-the-flag environment, testing and product evaluation
Virginia CR	Provide an environment to	Hosted in the cloud, web access, role-based	Cyber-security and computer network

	increase the number and the preparedness of students entering the cybersecurity	access, User-specific content, dynamically creating and destruction of the virtual environments, large target networks can be replicated for multiple and simultaneous usage	defence, training exercises for SCADA and ICS, cyber-law and policy topics, CTF competitions
Cisco CR	To provide an environment to support training and education for the participants of an organization to combat modern cyber-threats.	Implemented with variety of applications which provide visibility, intelligence, threat detection, firewalling and thread detection e.g., Cisco Stealthwatch, Cisco Splunk, Control. Includes traffic generator, configurable user emulation, malware, spyware and botnets emulation, individual and team training, end-to-end real-time solution for machine data delivery	Focus on cyber-security and computer network defense, threat detection, identification of various applications, network traffic pattern, secure network and applications, Cisco Stealthwatch, Cisco Splunk, Cisco TrustSec
IBM CR	Provide an environment to offer an experience in a cyber-incident	Exercise rapid response thinking in a pressured environment, understand how security solutions work together, experience how your teams work together	Discover gaps in enterprise's response plan, technical cyber response and leadership best practices, attacking tools, cybercrime topics, risk analysis

Table 1: Cyber Range Mission - Capabilities – Advantages

2.3 Related Projects

There are several projects related to cyber ranges that have become crucial platforms for security protocol trainings, tests, and optimizations in a range of scenarios. Various projects showcase the diverse approaches that exist within real-world cyber range systems, offering a demonstration of different methods and their usage in cybersecurity trainings, simulations, and system deployments in an attempt to mitigate specific challenges within this field.

2.3.1 Ludus

The Ludus project introduces a platform for creating user-friendly cyber ranges for development and testing[18]. Built on top of Proxmox virtualization platform, Ludus leverages automation while continuing to allow for manual virtual machine and network customization. It operates as a server environment with Ansible generating templates and provisioning advanced cyber infrastructures from one config file. Even with extreme flexibility, Ludus is heavily built to deploy upon Proxmox and similar infrastructures, which puts architectural limitations, particularly when operating systems based on ARM are being used. Additionally, its advanced networking architecture means that integrating it with external monitoring tools or even simulating attack systems is difficult.

2.3.2 Game of Active Directory

Game of Active Directory is an in-lab project that intends to provide penetration testers with an insecure Active Directory environment to test normal attack tactics.[20] The project is available in three configurations that differ in terms of the number of domains and virtual machines. These configurations are primarily designed according to the capacity of the host computer rather than modifying the cyber range environment. A basic takeaway from this project is putting focus on the deployment methods particularly tailored based on the specifications of the target host. While all of them reduce the same list of vulnerabilities of Active Directory, they are not modular and provide little flexibility in the overall cyber range configuration.

2.3.3 Attack Range

Attack Range by Splunk is another commendable effort, which supports both cloud-based deployment and local deployments of cyber range.[21] It aggregates a variety of tools in order to model cyberattacks and feed telemetry into a Splunk instance to analyze. This can be used to generate and test detection mechanisms, but its customization is primarily based on the number of nodes engaged with minimal interconnectivity complexity or many types of nodes supported. Though its focus is heavier on the offensive cyber methods, it depicts an emulation environment rather than representing existing systems, which, as efficient as they might be for some purposes, limits flexibility and realism.

2.3.4 CyTrONE

CyTrONE is an example of a project around teaching uses of cyber ranges. It focuses on the integration of training materials along with automation of environment control. Although extremely customizable, CyTrONE is designed for instruction rather than environment testing or experimentation. Rather than building a fresh cyber range, it builds upon existing infrastructure and making any modifications to the underlying cyber range must be achieved externally. This makes it a good solution for formal training but not necessarily ideal for dynamic testing and scenario construction.[16]

2.3.5 Related Projects Comparison

According to the established cyber range criteria, each of the referenced projects adopts a distinct approach, resulting in unique characteristics that make them individually appealing. A comparative chart outlining their core feature sets can be structured as follows:

Project	Modularity	Source Platform	Host Platform	Deployment	Monitoring	Attack Simulation
Game of Active Directory project	Limited to specific topologies	N/A, project is not cloning actual environment	VirtualBox VMWare Proxmox Azure	Automated	Not Included	Not Included
Attack Range	Limited to one topology	N/A, project is not cloning actual environment	All x86 architecture platforms Azure AWS	Automated	Included	Included
CyTrONE	Limited to one topology	N/A, project is not cloning actual environment	Linux	Semi-automated	Not Included	Not Included

Ludus	Limited to specific topologies but can expand.	N/A, project is not cloning actual environment	VirtualBox VMWare Proxmox Azure	Automated	Not Included	Not Included

Table 2: Related Projects Comparison Chart

2.4 Building Blocks

Building such projects requires application of appropriate tools that simplify deployment, management, development and maintenance procedures. Different strong technologies such as Vagrant, Packer, Terraform, Ansible and Bash offer solutions for automating, configuring and scaling virtual environments in development and production. Combined, these tools form an integrated system that highly enhances the effectiveness of development processes in deploying virtual environments.

2.5 Automation Tools

Modern cyber ranges leverage Infrastructure-as-Code (IaC) and configuration management tools to achieve scalability and repeatability. This section reviews the key automation platforms used in our deployment.

2.5.1 Vagrant

Designed for managing and provisioning virtual systems, vagrant is a command-line automating program. It lets consumers use particular configurations and network topologies to deploy virtual machines and containers. Additionally, enabling the automatic setup of sophisticated surroundings, vagrant lets installation and execution of more software inside the given computers. Its simplicity and compatibility with tools like VirtualBox and VMware make it a great tool for creating development-oriented infrastructure.[28]

2.5.2 Terraform

Designed to automate the provisioning of infrastructure across both local systems and cloud service providers, terraform is an infrastructure-as-code (IaC) instrument. Though it has parallels with Vagrant, Terraform is especially fit for creating production-grade infrastructure. Favored for its flexibility and declarative setup model, it assists a broad spectrum of providers. Nevertheless, Terraform misses some integrated capabilities that Vagrant provides including automatic networking, HTTP tunneling, and synchronized folders. Though networking and orchestration often need more setup, it may be well combined with other instruments to supply virtual resources for cyber range settings given these constraints.[27]

2.5.3 Ansible

Configuration management, application deployment, and task automation use Ansible as a strong automation framework. It runs agentless, needing only an SSH connection to the intended system. Highly readable, predictable, and reusable YAML-based configuration files known as "playbooks" are used by Ansible. Particularly useful for scaling and duplicating cyber range setups, the tool shines at controlling infrastructure across several hosts at once. By not depending on the target system for execution timing, Ansible also helps operational efficiency by allowing scheduled or recurring chores.[26]

2.5.4 Bash

Most Linux systems use the Bourne-Again Shell (Bash), a widely used UNIX shell, as the default terminal interface. Frequently pre-installed, it needs no further setup, making it easily available

for scripting and job automation. Bash scripts are rather adaptable and often used in cyber range installations since they can run low-level system commands. Other automation tools like Ansible, Terraform, and Vagrant use Bash as a fundamental building block to accomplish OS-level operations, therefore strengthening its Critical Enabling Role in Infrastructure Management

2.6 Platforms

The deployment and performance of cyber range settings call for a fit virtualization platform that allows scalability, flexibility and dependability. Depending on the intended use case, performance demands, and available resources, different platforms including cloud-based solutions and hypervisors offer unique benefits. The major platform kinds and their relationship with cyber range growth are covered in this section.

2.6.1 Cloud Providers

For cyber range environments, cloud service providers Microsoft Azure and Amazon Web Services (AWS) provide a simple and scalable infrastructure solution. These systems allow quick instantiation of virtualized environments without the requirement for physical hardware, fitting perfectly for distant and distributed training or testing applications. Managing sophisticated networking setups and worldwide resource access is one of the main benefits of cloud-based deployment. Still, cost control and environmental isolation are really important factors to be considered. Segregation from other resources under the same cloud subscription which could be unintentionally impacted extends beyond internet exposure to include separation.

2.6.2 VMware Workstation

Running on top of an already installed host operating system, VMware Workstation is a Type 2 hypervisor. It offers a solid foundation in a local setting for constructing, accessing and managing virtual machines. Built-in networking features of VMware Workstation, when correctly set up, enable remote access and connectivity. For individual development, testing, or smaller-scale cyber range situations, it is quite appropriate.

2.6.3 Hyper-V

Depending on the underlaying Windows environment, Hyper-V developed by Microsoft can be either a Type 1 or Type 2 hypervisor. Though it provides a full set of virtualization capabilities like those of other hypervisors, it is closely coupled with the Windows operating system and cannot be naturally implemented on non-Windows systems. Hyper-V is still an acceptable choice for companies mostly working within Microsoft ecosystems.

2.6.4 Proxmox VE

Proxmox Virtual Environment (VE) is an open-source Type 1 Hypervisor on a Debian Linux foundation. With this package, hybrid environments consisting of both KVM-based virtual machines and LXC containers with plenty of flexibility are hosted. Proxmox supports inbuilt clustering, HA, and live migration, making it most suitable for cyber ranges required to be resiliency-enabled and always available. The open-source platform of it and Linux system compatibility make it a cost-effective, highly configurable option for deployments in the cyber range. [25]

2.6.5 VMware ESXi

The virtualization capabilities at the enterprise level of VMware ESXi, a Type 1 hypervisor constructed out of scrap metal, are clustering, high availability, and hot virtual machine migration between multiple nodes. It can be directly administered using VMware Workstation or vCenter for orchestrating more intricate scenarios. In operational environments, the solidity and wide range of features of ESXi render it a popular option to deliver uptime and continuity of service. Mass-scale and robust cyber ranges suit well to their sophisticated features.

2.7 Adversary Emulation Frameworks

Several cyber range environments utilize antagonistic simulation frameworks, which enable the realistic simulation of attacker behavior to test and enhance defensive capabilities. These tools provide a complete platform for defense strategy validation and red team operations, mimicking threat actor tactics, techniques, and procedures (TTPs).

2.7.1 Caldera

A platform called CALDERA, which is open-source, was created by MITRE to emulate adversaries.[24] It provides support for a broad spectrum of red team capabilities, such as advanced Command and Control techniques, remote agent deployment, privilege escalation, system compromise and lateral movement within. CALDERA also includes planning campaigns against adversaries, giving users the ability to design sequenced attack scenarios for training and testing. Designed as a modular system that is compatible with the MITRE ATT&CK framework[30], it offers significant flexibility and potential for adversary emulation in cyber ranges.

2.7.2 Cobalt Strike

A framework known as Cobalt Strike, which is commercial-grade, provides a framework for adversary emulation after exploitation and can be used in large-scale operations like red team[31]. It enables collaborative offensive engagements, imitated attacker behavior in a controlled environment and provides robust agent-based control. It is widely regarded as one of the leaders in the field and is often used in offensive security training, or even in actual World War II simulations. Its proprietary nature and necessary license may restrict access to projects that are academic or budget-challenged.

2.8 Monitoring Tools

The use of monitoring tools is necessary to observe, analyze, and react to cyber activities. The systems can be observed, abnormalities detected, and malicious activities detected. All of these are provided by the tools. When properly integrated, they enhance the feedback loop between red and blue teams, enabling better training and operational decisions.

2.8.1 PfSense

FreeBSD-based PfSense is an open-source router/firewall operating system used in cyber ranges to simulate perimeter network security. It has stateful packet inspection, IDS/IPS with Snort or Suricata for intrusion detection and prevention, VPN support, and full logging capabilities. With modular architecture and UI usage, it is a simple and flexible solution for building intricate network topologies and viewing real-time attack and defensive streams.[34]

2.8.2 Splunk

By employing Splunk, a robust data aggregator and log analysis platform can gather both system and application logs for monitoring events and identifying potential threats through pre-defined rules and patterns. Splunk, when properly configured, functions as a SIEM system that delivers dashboards, alerts and advanced analytics for cybersecurity monitoring. The financial cost of licensing Splunk can hinder small-scale or academic cyber range deployments, despite its capabilities being extensive. [32]

2.8.3 ELK Stack

Unlike commercial SIEMs like Splunk, the ELK Stack is an open-source alternative that includes Elasticsearch, Logstash, and Kibana. Featuring powerful features for data indexing, searching and visualization.

- Logstash collects and processes logs from various sources.
- Elasticsearch indexes and stores the data to enable quick search and retrieval.
- Kibana offers interactive dashboards and visualization tools to explore data trends and anomalies.

The ELK Stack is a versatile, scalable tool that is widely used in cyber ranges for security analytics, providing flexibility without the need for licensing expenses of proprietary platforms. Captured telemetry from many different virtual infrastructures is made possible by its compatibility with numerous input sources. [33]

2.8.4 Wazuh

The Wazuh System acts as a unified open-source security tool, serving extend network monitoring capabilities and logging aggregation systems by incorporating features of both Extended Detection Response (XDR) and Security Information and Event Management (SIEM). In our cyber range environment, Wazuh gives insights into individual endpoint devices (virtual machines) rather than focusing only on monitoring network communications.

The architecture involves lightweight agents deployed on the monitored computers (Windows Server and Linux Server), communicating with a central Wazuh server. This setup facilitates comprehensive telemetry data collection and analysis and provides a number of primary capabilities:

1. Intrusion Detection: Wazuh acts as a Host-based Intrusion Detection System because it analyzes the file integrity level, rootkits, and hidden processes that reveal unauthorized access and the presence of malware.
2. Vulnerability Identification: The system searches installed apps and operating systems against known vulnerabilities (CVE) databases in order to determine where vulnerabilities exist in the range environment.
3. Active Response: Unlike passive monitoring solutions, Wazuh can be used to perform an active response to threats based on preset criteria, such as disconnecting a network connection or deleting a malicious file for which the preset conditions have been met. Within the scope of this thesis, Wazuh can be utilized for the consolidation of security information related to various endpoints. It creates an end-to-end feedback mechanism by mapping host-based events with overall network information, and very often, it is combined with the ELK Stack (or its variant, OpenSearch) for creating dashboards related to security alerts [23].

2.9 Related Papers

Many research and proposals exist to focus on improving cyber range technologies and methodologies, with the goal of making cybersecurity training and testing environments as advanced and resilient in response to ever-changing threats.

The current and future trends in cyber-ranges and test beds[3]. In this comprehensive analysis, a classification of cyber ranges (CRs) and test beds (TBs), along with their type, purpose and technological implementation, is presented. This points out the growing need for cyber situational awareness and training tools that can simulate changing threat scenarios. It highlights the increasing overlap between IT and OT in CR development and explores their application in education, military and industry. Additionally, they introduce two taxonomies that synthesize various CR dimensions, including use cases, user types and threat environments. The research highlights the necessity of standardized design methods and adaptable CR implementations.

In "Exploiting, Training and Research in the Flexible Cyber Security Environment of the AIT Cyber Range" [4] , the AIT Cyber Range, a versatile platform that relies on open-source technologies like Terraform and OpenStack, is presented in this paper. It is architecture and use cases are discussed. Featuring a modular architecture that incorporates computing, infrastructure programming, software provisioning, and scenario engine functionality, the system is designed to cater to various cyber exercises training and research applications. It emphasizes the importance of scalability, interoperability and scenario-driven simulations to achieve greater real-life realism. According to the authors, this platform is advantageous for cross-organizational training and collaborative research.

A Multi-Cyber Range for Cyber Training and Testing [5], this study proposes a multi-cyber range architecture that is tailored to the military and can reflect the battlefield systems' interdependence. Incorporating integrated scenario authoring and evaluation functions, it enhances mission-based training, particularly in cyber-physical and weapon system settings. The authors' DDoS experiments reveal the effectiveness of the platform as a means to test interoperability. Multi-domain cyber ranges are increasingly necessary in military and industrial settings, as pointed out by them. It argues that "the paper proposes testbeds that are federated and scalable, with real-time responsiveness and the ability to simulate adaptive threats".

A study on the effectiveness of designing cyber training and cyber range to effectively respond [6] was published. The proposal in this paper suggests a cyber training framework that merges physical systems with virtual ones to generate more realistic and adaptive training programs. The cyber training is divided into three parts, namely environment, scenario and operation. It also introduces the integration of training elements in spatial and temporal ways through a system. According to the study, integrated training infrastructures that incorporate modular design and scenario-based exercises are essential. A contribution is the proposal of a national-level architecture that involves linking cyber ranges from various institutions. This is particularly notable. It improves group training, enhances resource sharing and prepares participants for real-world incident response.

In the Cyber Range Design Framework for Cyber Security Education and Training [7], The CRDF is a systematic approach to designing cyber ranges that are effective, reusable and teachable. It has been presented in this research. This framework uses a CR architecture and the life-cycle of CC to guide development and deployment, building on the success of the COFELET (Coeducational Education for Work) e-learning model. CRDF emphasizes the use of dynamic assessment, real-time feedback and scenario learning, as well as its adaptability to learner needs. According to the study, existing CRs have significant shortcomings such as high cost, complexity and a lack of standardization, which are addressed through the use of structured design principles.

SOC Teams' Cyber Range: Delivering NICE by Design Cybersecurity Learning Environments

The research [2] examines the NICE framework, which is a structured approach to learning cybersecurity developed at NIST. The paper highlights the nonexistence of established design principles in cyber ranges and suggests a method that assists in the creation of scenario scenarios and environmental assessment, with the objective of aiding in developing practical cybersecurity scenarios.

A Model Driven Cyber Range Training Perspective on Cyber Security Assurance

A model-driven approach for Cyber Range Training, which enables the automated deployment of specific environments through simulation and emulation capabilities, is presented in this paper [8]. Various levels of difficulty and complexity are involved in the training scenarios, including phishing threats.

A cyber range instantiation system called CyRIS is designed to help with security training.

The study [1] suggests a Cyber Range creation system that can automate the preparation and management of cyber range environments, which are designed for educational use. This instrument adheres to the guidelines of the Technical Guide of Information Security Testing and Assessment of NIST [12] and is also comparable with similar cyber range creation tools.

Security testing centers and cyber perimeters: The environment, functions, tools, and structure

This article [9] focuses on the role of cybersecurity training as the primary means of safeguarding against cyber threats and criminal activities. It outlines two types of training, one which improves skills and knowledge for security professionals on current threats (the "best practices") and risk mitigation strategies and another that seeks to increase cybersecurity awareness among general practitioners. Training scenarios can be carried out in Cyber Range environments, which also offer a practical environment for trainees to utilize. It creates a classification of cyber range systems and scrutinizes existing literature by examining various aspects such as architecture, scenarios, capabilities, roles, tools and evaluation criteria. These discoveries are intended to serve as a basis for future endeavors related to the design and assessment of cyber ranges, in keeping with current best practices.

Recent and future trends in cyber-relations

The paper [3] discusses cyber situational awareness and how it can improve threat and vulnerability insight in organizations, with a focus on how exposure is determined by the number of procedures under implementation and existing cyber-hygiene. Since the accelerating cyber-attack automation has caused the gap between information and operation technologies to narrow, there is a need to redouble current security robustness against future cyber-attacks. This proposition encourages the thorough exploration of security contexts with a view to anticipating potential weaknesses and guiding the deployment of effective technologies and secondly stresses the need for ongoing training practices in an effort to assist users and operators during decision-making. Cyber-Ranges (CRs) and Testbed (TBS) play a central role in this practice since they facilitate thoughtful deepening of attack evolutions and effective deployment of countermeasures. It also analyzes the documented CRs and "Transformers By Certification" (TBs) by class, Threat category ("Cause", Applications etc.), Training scope. Further, a soundly rooted classification system enhances insight into future development of CRs and TBs, suggesting fewer discrepancies among their application areas; essentially expanding the scope of implications and application in cybersecurity.

National Cyber Range Overview

A specialized environment for cybersecurity testing is provided by the Test Resource Management Center, which was originally established by DARPA and is now known as the National Cyber Range (NCR) [10]. In order to improve cybersecurity resilience, program managers (PMs) must utilize a cybersecurity range, as explained in this paper. One of the key issues that the NCR needs to address is creating a test environment that accurately simulates the scale and diversity of DoD's complex communication networks, thus providing realism in potential cyber threats such as malware attacks, DDoS, and cross-site scripting. This paper discusses two challenges. By utilizing virtual machines, physical hardware, traffic emulation, vulnerability scanning, and data capture tools, the NCR offers a test environment that is as close to the surface of the Internet as possible. PMs can utilize a structured test methodology to assess and enhance the cybersecurity resilience of their systems. The knowledge obtained by employing the NCR can facilitate early integration of cybersecurity measures, which may lead to cost-savings and system-level security. The paper intends to educate DoD PMs about the NCR's resources and its potential in testing/fine-tuning cyberspace resilience.

3. Implementation

In this section, we shift from theory to practice, detailing the practical execution of our cybersecurity simulation Cyber Range Lab. Our aim is to offer a clear account of how we put our cybersecurity strategy into action. We will explain the technical steps taken, tools used, and outcomes achieved, providing a practical guide for cybersecurity professionals and researchers.

3.1 Cyber Range Development

3.1.1 System Requirements

In the Cyber Range Development subsection, we dive into the initial steps of setting up our cybersecurity lab environment. To kickstart this setup, our fundamental requirement is a Debian Linux machine running version 12. This Linux host serves as the foundation for our lab environment, which will encompass multiple interconnected components.

To accommodate the complexity of this lab environment and ensure optimal performance, we have allocated a substantial 26 GB of RAM to the host virtual machine (VM). Additionally, we have maximized CPU (6 cores) resources and reserved 200 GB of free storage space. These resource allocations are essential to support the diverse range of components and functionalities within our Cyber Range. These allocations represent the maximum resources available on the host workstation used for this deployment.

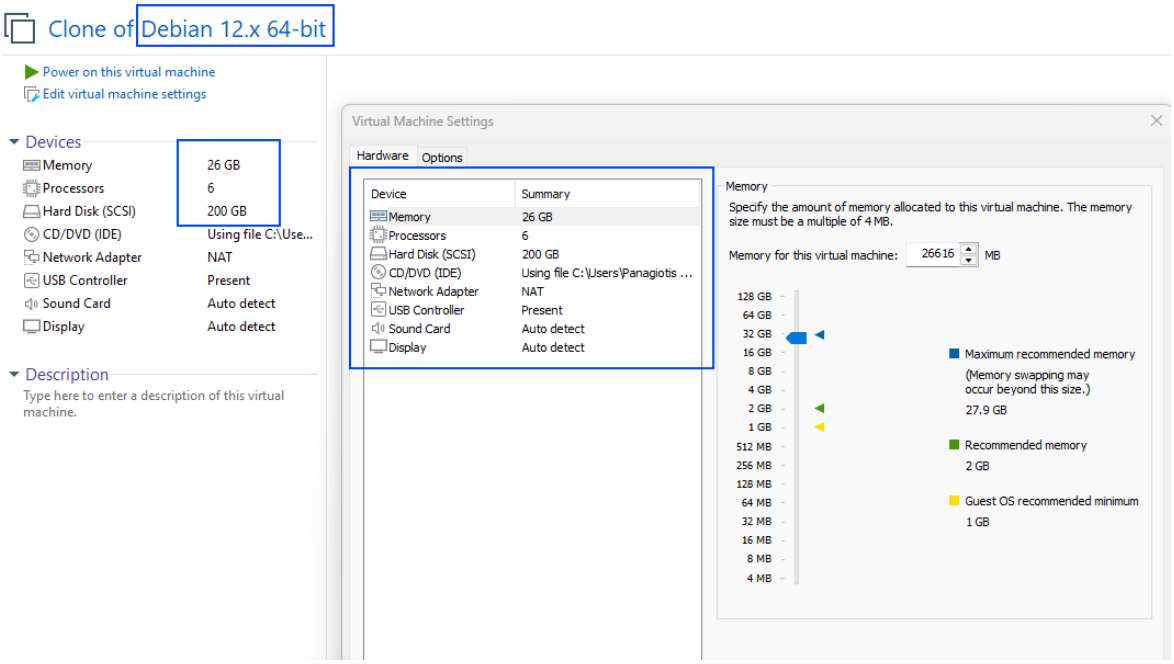


Figure 2: VMware virtual machine settings

Ludus is best practice to be installed on a host that meets the following requirements:

As for minimum requirements for the Ludus site, we will need the following:

- x86_64 (aka amd64 aka 64-bit "Intel") CPU with a Passmark score > 6,000
- Debian 12 or Proxmox 8
- Supports virtualization - vmx or svm in /proc/cpuinfo
- Has at least 32 GB of RAM
- Has at least 200 GB of disk space (fast NVMe recommended)
- Root access
- Internet access (not via WiFi).

Note: Bonded NICs or other advanced networking is not supported. If we use these, we will need to console in and fix the network after installation (edit /etc/network/interfaces), as Ludus assumes we have a single, standard interface.[18]

3.2 Ludus Implementation

The main objective of the current technical part of the thesis is to illustrate the way that a basic cyber range environment using Ludus can be deployed in a scriptable way[18]. The current topology can be easily changed and the whole environment can change according to the desired use case. The included systems and topology implemented here aim to provide a baseline for a representative environment and are also subject to change by changing the deployment script and/or the actual system images. Also, the deployment target/host can change but for illustrative purposes Proxmox Virtual Environment are chosen.

With automatic environment setup and content generation based on yaml files, it facilitates anyone to conduct security research or training with such a deployment with a very low fingerprint on any host with virtualization capabilities.

3.3 Reasons to choose Ludus

Setting up and maintaining these labs has often been a tedious and time-consuming process. Ludus turned out to be the simplest and most efficient solution encountered for building and managing a lab. One of the standout features of Ludus is its support for Templates (pre-configured VM templates) and Ranges (pre-built lab environments). Also, it is very useful that Ludus uses YAML files for configuration.

This allows us to:

- Customize our lab environment to suit our needs.
- Share our setup with colleagues or friends effortlessly.
- Use YAML files created by others to replicate their lab environments.

This flexibility and ease of use make Ludus an excellent choice for anyone looking to streamline the process of creating and maintaining a red team lab.[18] [19]

3.4 Underlying Infrastructure Analysis

Our Cyber Range network contains:

- Domain Controller (JD-DC01-2022) for the domain ludus.domain
- Windows 11 workstation (JD-WIN11-22H2-1 with Office 2019 64bit, Chrome, Firefox Burpsuite, VSCode, 7zip, process hacker, ilspy and other useful utilities)
- Kali VM (JD-Kali) in a separate network

The Windows hosts can only reach the Kali VM on tcp/80, tcp/443, and tcp/8080. The Kali VM can reach the Windows VM on any protocol and any port.

When testing mode is enabled, both Windows hosts will be snapshotted and blocked from reaching the internet, while the Kali VM will not be snapshotted or blocked from reaching the internet.

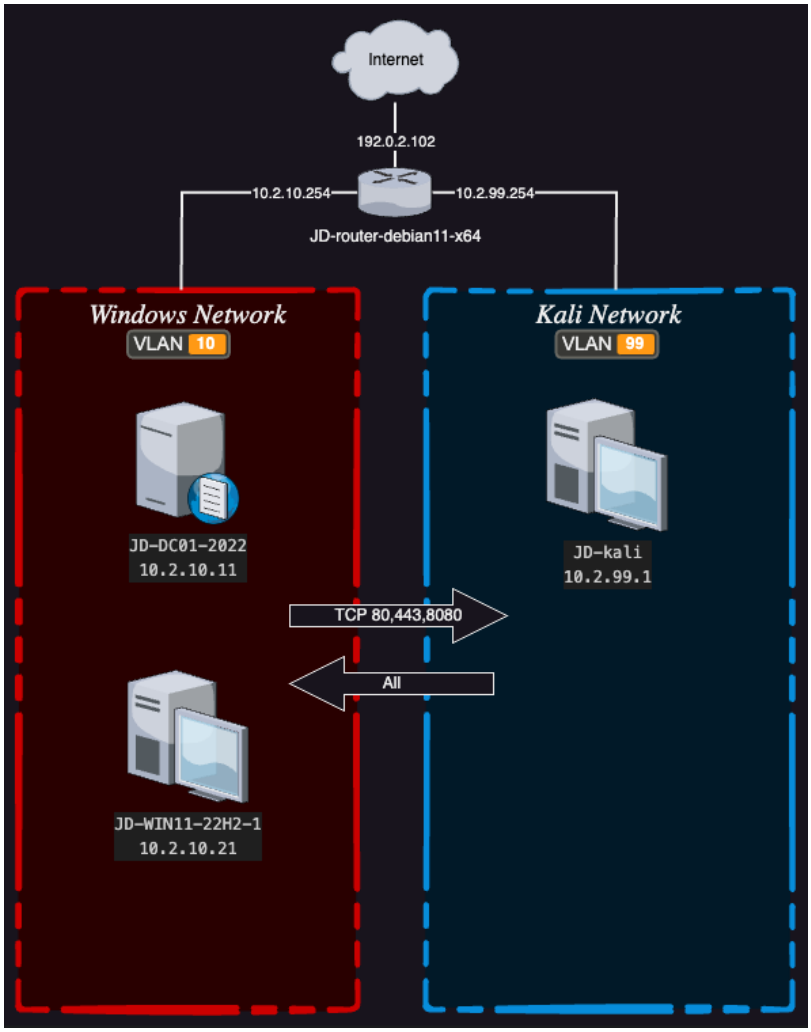


Figure 3: Cyber Range Topology

3.5 Preparing the Environment

Before installing and configuring the Ludus platform, the underlying operating system environment must be prepared. The following sections detail the Debian 12 base installation and post-installation configurations.

3.5.1 Getting Started with Debian 12

To begin, we have performed installation of Debian 12 on our server, as recommended by the Ludus documentation. The installation process is straightforward, but there is one key step to keep in mind: enabling the SSH server during setup. This allows us to connect to the server remotely at any time during the setup.

Figure 4: Debian 12 Installation Process

At this step, Debian 12 installation has been started and is in progress:

Figure 5: Debian 12 Installation in progress

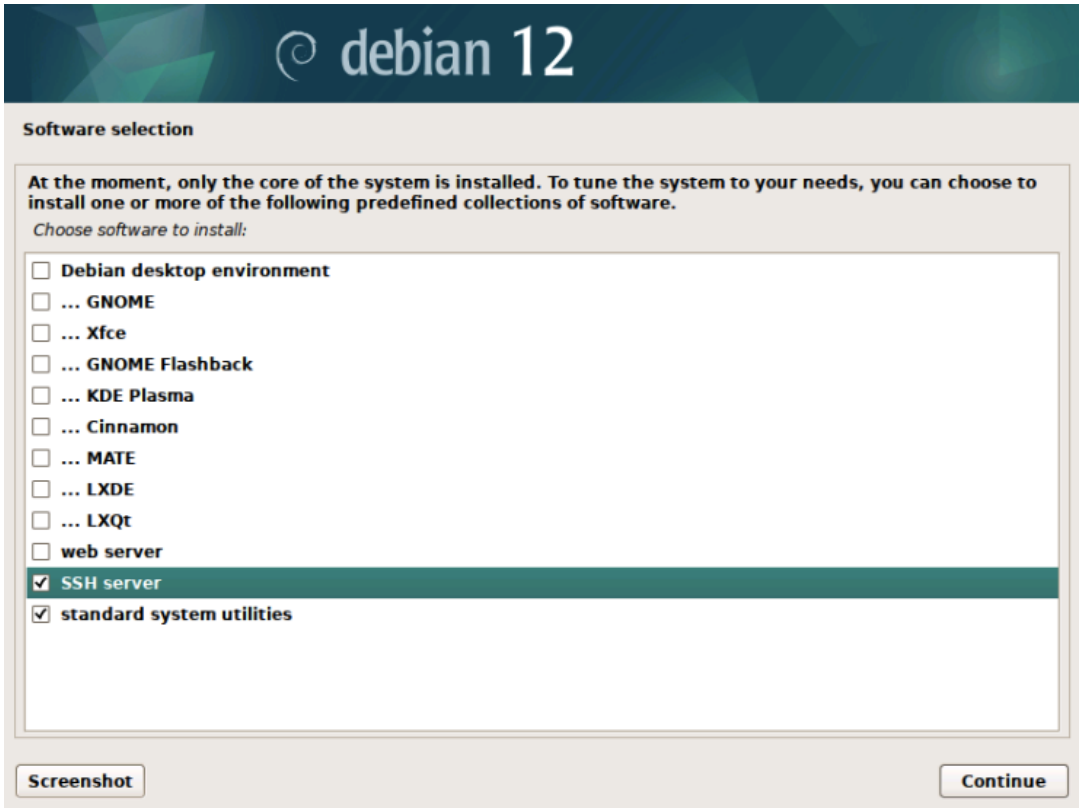


Figure 6: Debian 12 enabling the SSH server setting

Ensure the installation is performed using an Ethernet connection rather than Wi-Fi. Ludus does not currently support Wi-Fi and using it during installation may require reconfiguring the network interface afterward, which can be an unnecessary hassle.

3.5.2 Desktop Post Installation

When system boots up post Debian 12 installation, we will get following login screen, use the same username and its credentials that we have created during the installation.

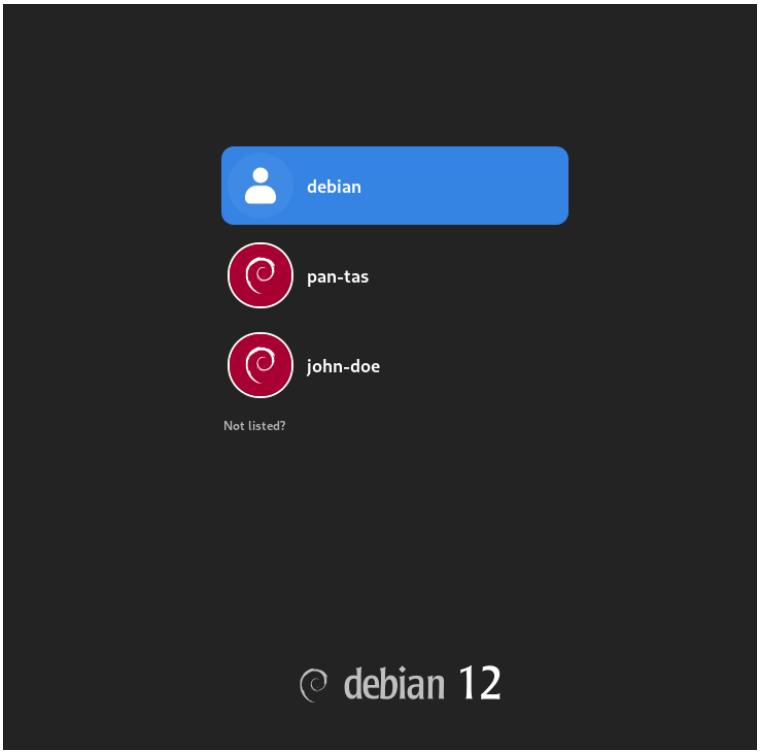


Figure 7: Debian 12 login screen

If we prefer to keep our server running without automatic suspension, we can disable auto-suspend with the following command:

```
systemctl mask sleep.target suspend.target hibernate.target hybrid-
sleep.target
```

3.5.3 Software Dependencies and Initial Setup

The installation of the Ludus Cyber Range environment requires the implementation of a unique set of software technologies to be primarily based on virtualization and automation solutions. This section explains the details of the software requirements and initial configurations executed on the Debian host system.

The Ludus tool uses an arsenal of powerful open-source tools in order to work properly. The core dependences of tools are:

1. qemu/qvm: The basis for hardware-based virtualization on Linux. The kvm module is the hypervisor, with qemu being the machine emulator.
2. libvirt: API and management tool to work with KVM and other virtualization solutions.
3. Docker: An application containerization tool that facilitates creating and executing application containers in isolated containers.
4. Python 3.9+: The main scripting language used in most of the components and management scripts of the platform.
5. Ansible: Open-source automation tool used for configuration management, software provision, and application deployment.

System Update and Upgrade

To install new packages on the server, it is necessary to perform the initial configuration process on top of all other processes on that server by making sure that it is updated before proceeding. Users can upgrade the installed packages to make updating easier with the apt tool, which is integrated into the server.

```
Step 1: Update and Upgrade Your System
sudo apt update && sudo apt upgrade -y

Step 2: Install Required Packages
sudo apt install -y qemu-kvm libvirt-daemon-system libvirt-clients virt-
manager \
python3 python3-pip git curl make unzip net-tools docker.io ansible

Step 3: Verify KVM Support
egrep -c '(vmx|svm)' /proc/cpuinfo

Output should be 1 or more. If 0, enable virtualization in BIOS/UEFI.
```

The command hunts in the CPU information text file '/proc/cpuinfo' for the flag SVM or VMX.

If the output is 1 or more, it shows that the CPU has hardware virtualization support and it is detectable by the operating system.

A value of 0 indicates that the CPU either does not support the feature or, in most cases, that the feature is disabled in the machine's UEFI/BIOS.

For KVM to function correctly, the host system's CPU must support hardware-assisted virtualization. This feature is known as Intel VT-x for Intel processors or AMD-V (SVM) for AMD processors.

A verification check is performed to confirm this support is available and enabled:

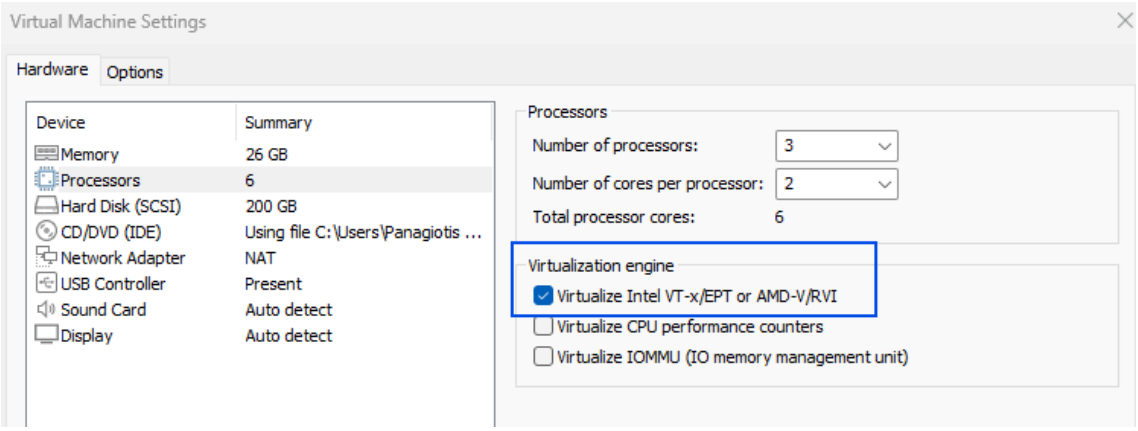


Figure 8: Virtual Machine Virtualization engine setting

3.6 Setting up and installing Ludus

Ludus is a lab automation tool that can deploy complex environments like Active Directory labs, penetration testing environments, and more with minimal effort. Setting up Ludus on Proxmox in the instructions by default involves creating a virtual machine, installing Ludus, and configuring it to deploy and manage our lab environments.

However, deploying Ludus directly on the Proxmox host works just as well. There is an easy one-liner to install Ludus on the host but be warned: installing Ludus this way will effectively take over the host, so make sure it is what we want to do.

There is an option to create a dedicated user for Ludus, followed by running the one-liner:

```
curl -s https://ludus.cloud/install | bash
```

The install.sh the script will install the ludus client, optionally shell completions, and then prompt to install the server. We just accept the default value and continue the installation. The installer will start and reboot the machine.

To install ludus, we use the installer script on a Debian 12 machine as shown below. It will extract files into /opt/ludus and walk through the configuration values during installation as seen below.

```
local:~$ ssh user@debian12

user@debian12:~$ su -
# Enter root password to elevate to root
root@debian12:~$ apt update && apt install curl sudo

# All-in-one command
root@debian12:~$ curl -s https://ludus.cloud/install | bash

# If you want to check out the install script
root@debian12:~$ curl https://ludus.cloud/install > install.sh
root@debian12:~$ cat install.sh
root@debian12:~$ chmod +x install.sh
root@debian12:~$ ./install.sh
```

Figure 9: Ludus Installation Process

```
curl -s https://ludus.cloud/install | bash
====================================

 _       _   _  ____    _   _  ____
| |     | | | ||  _ \  | | | |/ ___\
| |     | | | || | | || | | |\___ \
| |___  | |_| || |_| || |_| | ___) |
|_____/  \___/ |____/  \___/  \____/

====================================
[+] Client install prefix set to /usr/local/bin
[+] Created temp dir at /tmp/ludus-client.wXhGm0
[+] Architecture detected as x86_64
[+] OS detected as Linux
[+] Downloaded ludus-client_linux-amd64-1.7.3 into /tmp/ludus-client.wXhGm0
[+] Downloaded ludus checksums file into /tmp/ludus-client.wXhGm0
[+] Checksum of /tmp/ludus-client.wXhGm0/ludus-client_linux-amd64-1.7.3 verified
[+] Install prefix already exists. No need to create it.
[+] Installed ludus-client_linux-amd64-1.7.3 to /usr/local/bin/ as 'ludus'
[+] Ludus client installation complete
[+] Shell completions already installed
[+] Ludus server already installed in /opt/ludus
[?] Would you like to update the Ludus server on this host?
[?] (y/n): y
```

Figure 10: Ludus Installation Process

Once we select yes, the installer will auto-detect settings.

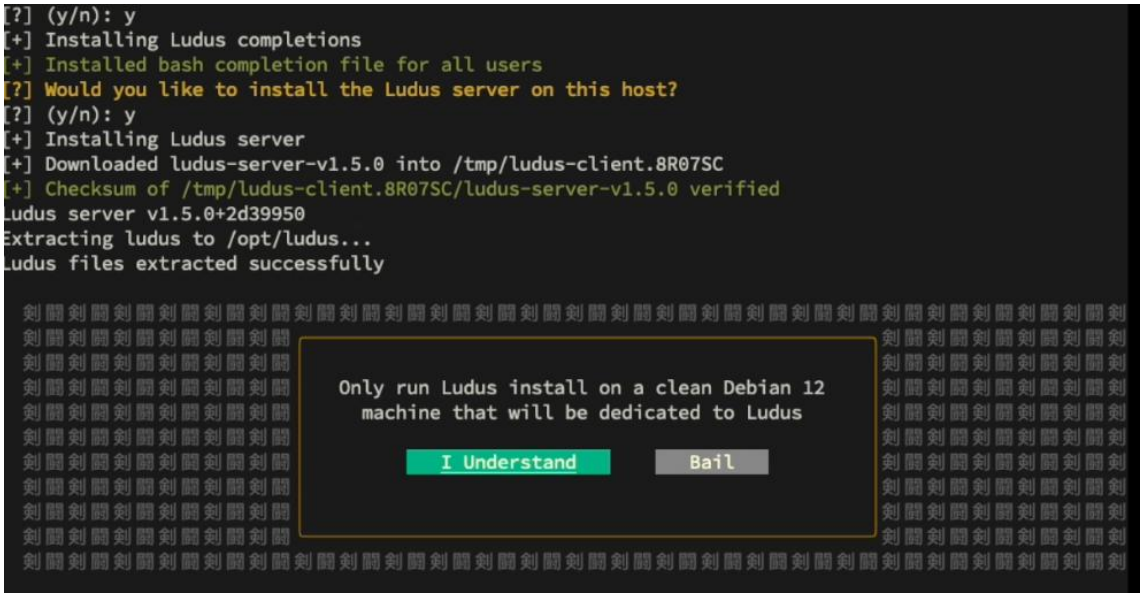


Figure 11: Ludus Installation Process

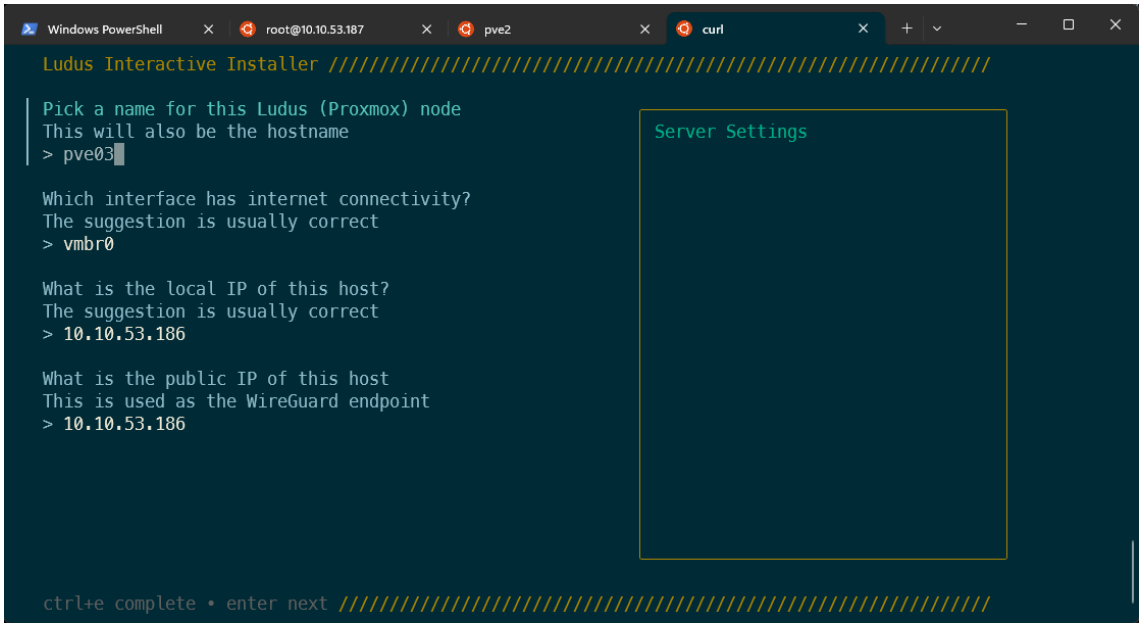


Figure 12: Ludus Installation Settings

Each setting we hit return on will build the server settings config, and we will see it populated on the right side as seen below:

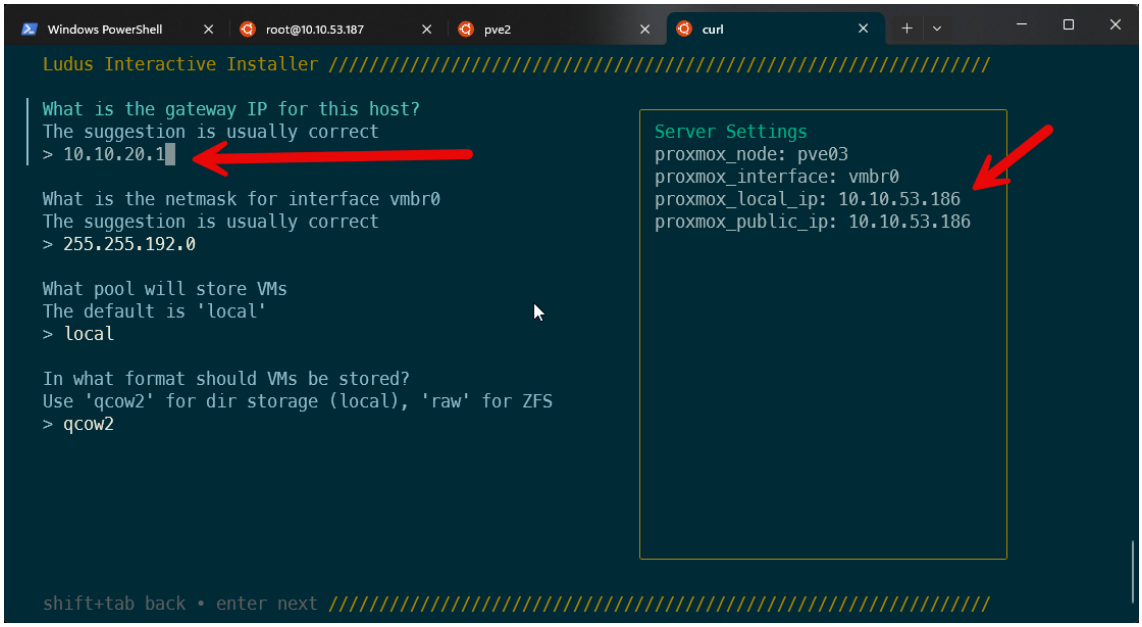


Figure 13: Ludus Installation Settings

Once we have gone through and built our config, Ludus will finally ask us if we are sure we want to install it with ready?

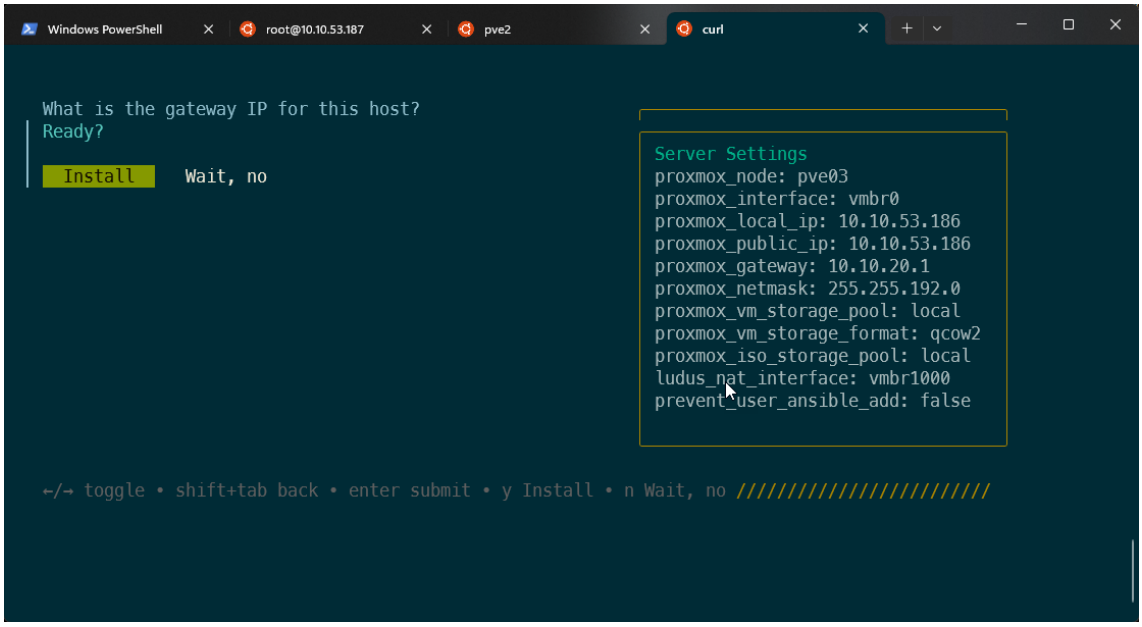


Figure 14: Ludus Installation Settings

The installer then kicks off a lot of Ansible that automagically installs the Ludus client and server to the host:

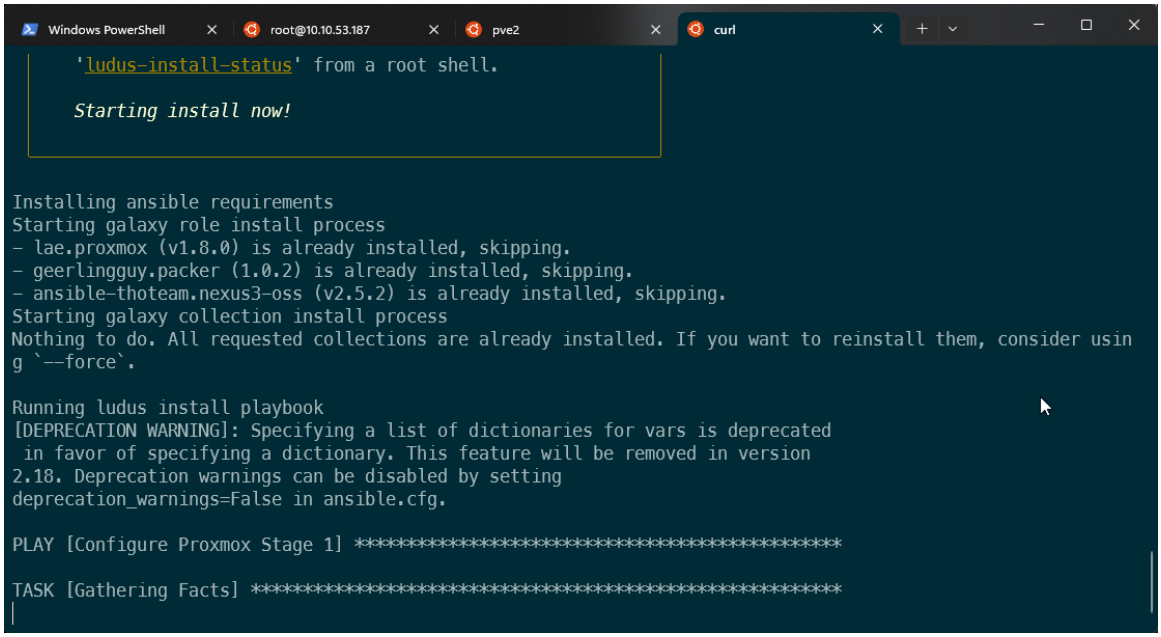


Figure 15: Ludus Installation

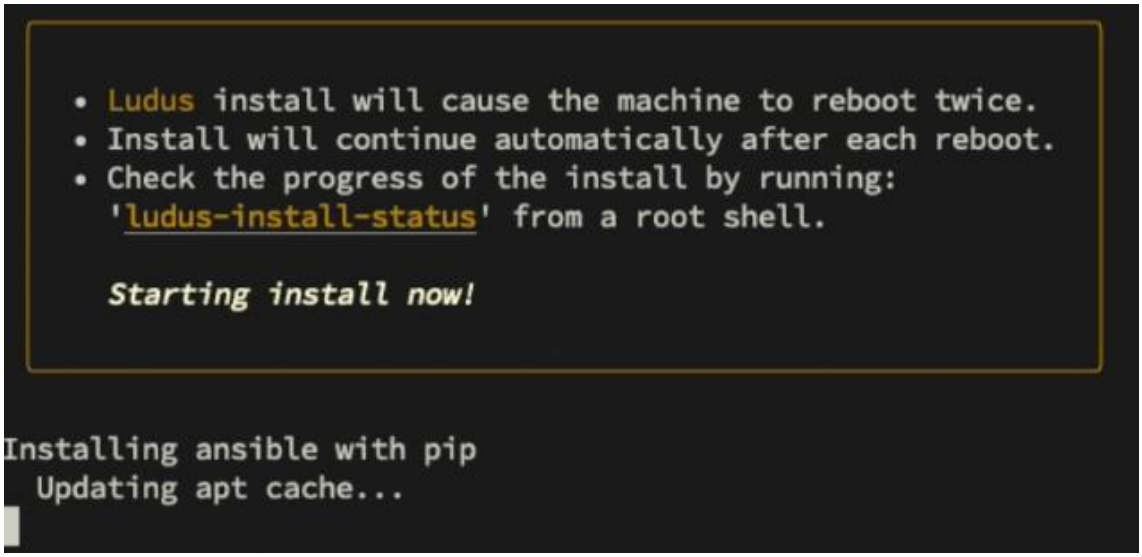


Figure 16: Ludus Installation

The install.sh script will install the ludus client, and optionally shell completions, and then prompt to install the server. The installer will start and reboot the machine. The server will reboot multiple times during the installation process.

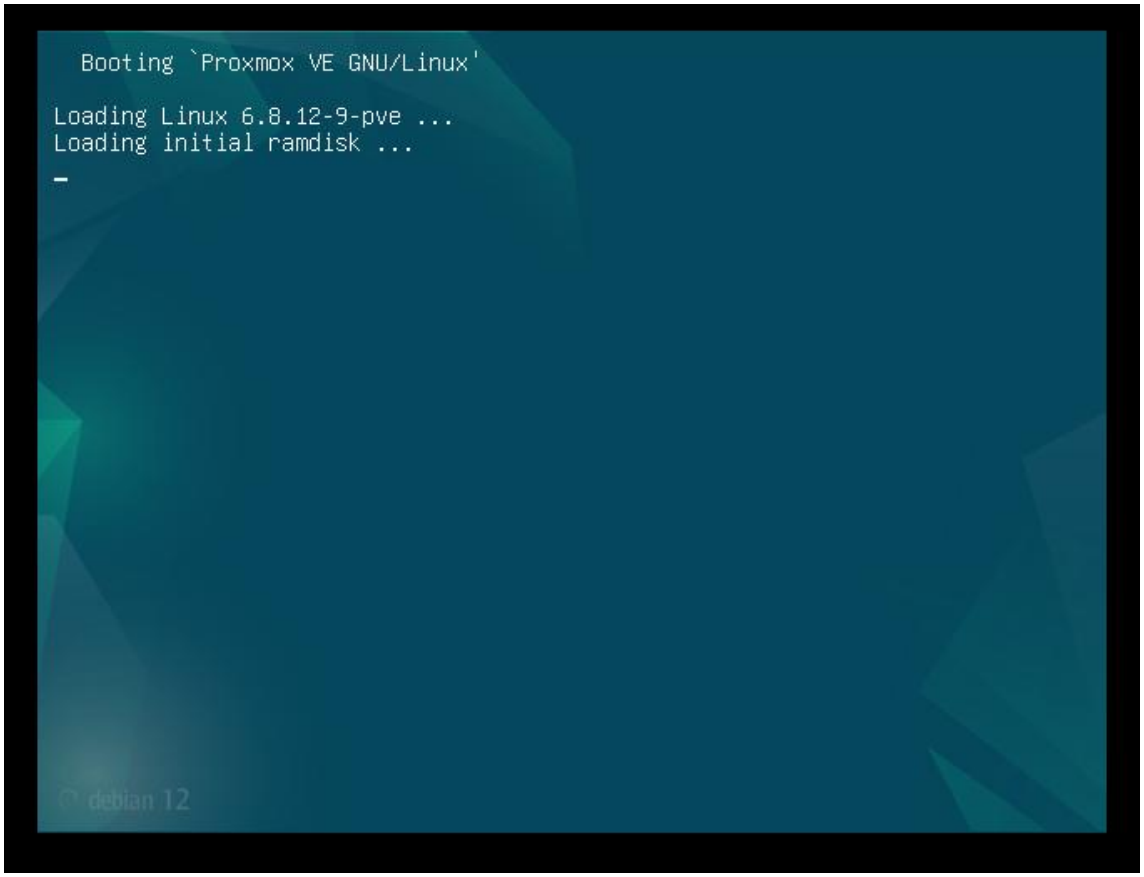


Figure 17: Ludus Installation

After the reboot, the installation will continue automatically. To monitor its progress, we perform ssh into the machine, elevate to root, and run:

```
>ludus-install-status
```

Ludus will not allow us to perform any action without creating a standard Ludus user.

```
debian@ludus:~$ ludus-install-status
You must be root to check the Ludus install status
debian@ludus:~$ su -
Password:
root@ludus:~# ludus-install-status
Ludus install completed successfully
Root API key: ROOT.BXYcpuOo2fKLBkYPLNfR0fE1b9PkbSM9e2pK592J
root@ludus:~#
```

When we run the install status command, the screen above indicates the installation succeeded. The next step is to create a Ludus user to delegate the installation and prevent us from using the root account.

3.7 Creating User

We are using the Ludus client to create a Ludus user. To perform user related actions, which modify the Ludus host as root, we must connect to the admin service which only listens on localhost.

On the Ludus host run ludus-install-status which will print the root API key.

```
debian@ludus:~$ ludus-install-status
You must be root to check the Ludus install status
debian@ludus:~$ su -
Password:
root@ludus:~# ludus-install-status
Ludus install completed successfully
Root API key: ROOT.BXYcpuOo2fKLBkYPLNfR0fE1b9PkbSM9e2pK592J
root@ludus:~#
```

3.7.1 Using Ludus client to create a Ludus user

Now we can create a ludus user. This user will be an admin as we specify --admin. Initials are commonly used for the userID. Prepend the LUDUS_API_KEY variable to the command to authenticate properly. It may take up to a minute to generate, but once it is, we will be presented with a new API key for our user “Pan Tas”.

```
debian@ludus:~$ LUDUS_API_KEY='ROOT.BXYcpuOo2fKLBkYPLNfR0fE1b9PkbSM9e2pK592J' \
> ludus user add --name "Pan Tas" --userid PT --admin --url https://127.0.0.1:8081

[INFO]  Adding user to Ludus, this can take up to a minute. Please wait.
+--------+-----------------+-------+-------------------------------------------+
| USERID | PROXMOX USERNAME | ADMIN |                API KEY                    |
+--------+-----------------+-------+-------------------------------------------+
| PT     | pan-tas         | true  | PT.vxxlctY@Uhui=1EcLQAbYee6N2cN9fcoAZfM3nBr |
+--------+-----------------+-------+-------------------------------------------+
```

3.7.2 Set the API Key + Get Proxmox Credentials

Using the key from the previous step, we will export the LUDUS_API_KEY variable so it is known to future commands. Ludus is built on top of the Proxmox hypervisor which has a web interface. It is available at **https://<ludus IP>:8006** and the credentials for the web GUI can be retrieved with ludus user creds get.

Once we have our key noted we can grab our credentials for proxmox for the newly created user with:

```
>ludus user creds get
```

```
debian@ludus:~$ export LUDUS_API_KEY='PT.vxxlctY@Uhui=1EcLQAbYee6N2cN9fcoAZfM3nBr'
debian@ludus:~$ ludus user creds get
+------------------+----------------------+
| PROXMOX USERNAME |   PROXMOX PASSWORD   |
+------------------+----------------------+
| pan-tas          | cBXw7eABJJ73mcnT4CbD |
+------------------+----------------------+
```

3.8 Logging into Proxmox

We can access the Proxmox web interface by opening the following URL in our browser:

```
https://LUDUSHOST-IP:8006/
```

Now that we have created the user and obtained our user creds for proxmox, we can build templates.

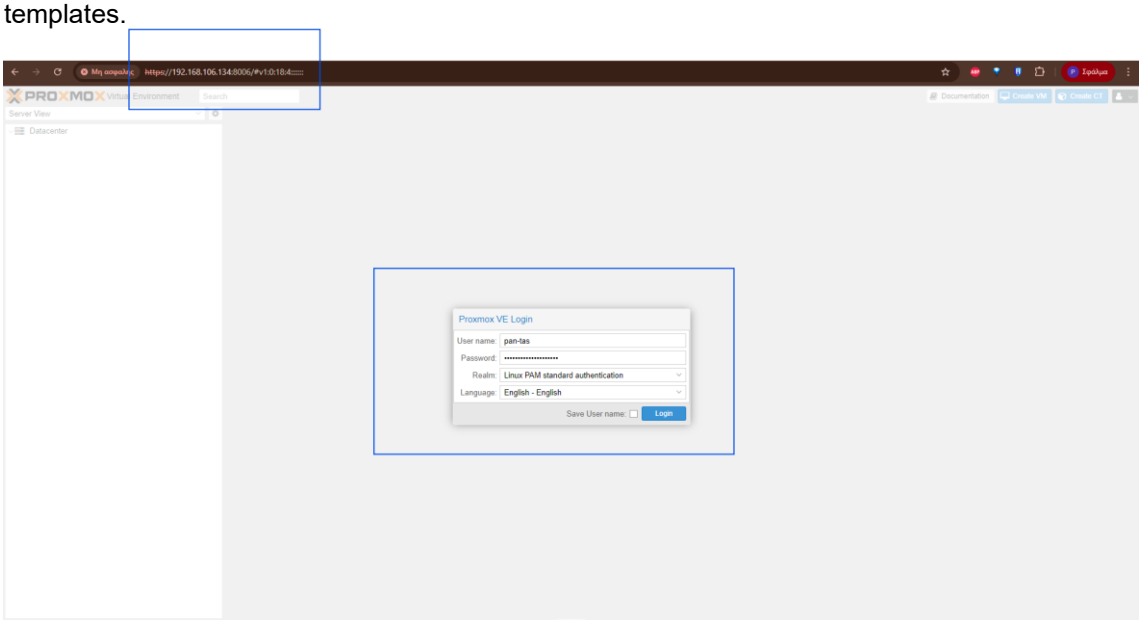
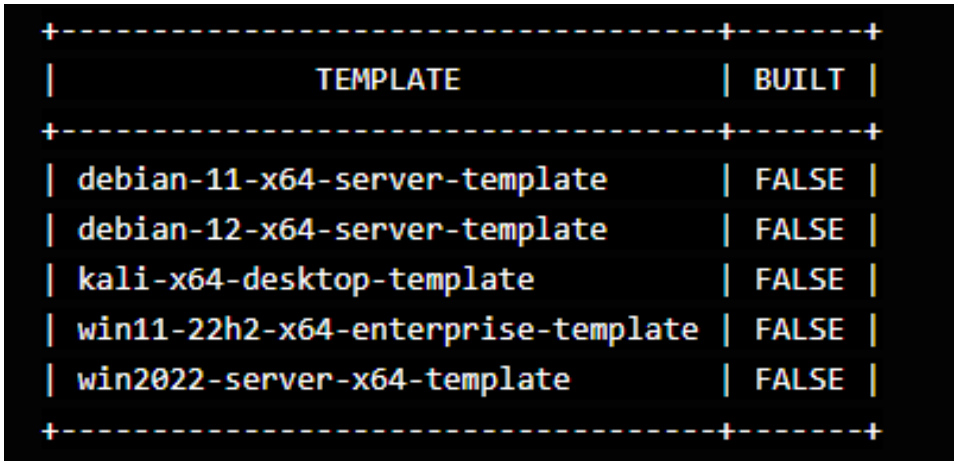


Figure 18: Proxmox Web User Interface login Screen

The next step in the chain is that we will want to build and add templates for the different lab environments in order to set them up; this can be achieved with the templates command:

```
>ludus templates list
```



TEMPLATE	BUILT
debian-11-x64-server-template	FALSE
debian-12-x64-server-template	FALSE
kali-x64-desktop-template	FALSE
win11-22h2-x64-enterprise-template	FALSE
win2022-server-x64-template	FALSE

The default list is shown below but if we want more, we can clone them from (https://gitlab.com/badsectorlabs/ludus).

3.9 Cyber Range Deployment

3.9.1 Initial Image Generation

Primarily, the process of establishing a custom environment has always been concerned with the combination of configuration, architecture, connectivity, and implementation aspects. The custom environment that has already been functional and deployed has been targeted by malicious users, and therefore, there should be predictability in terms of defending and monitoring the said environment. This explains why there is a need to develop automation tools and scripts so that the researcher can clone them in a controlled environment to perform their study.

As there is no cyber range environment to copy from, an initial deployment has to be performed to set up the original environment to be cloned later on. This process involves the installation of Virtual Resources necessary to set up the cyber range environment by hand on a hypervisor. This process is very simple and not relevant to the thesis, only involves setting up the hypervisor of choice, in our case Proxmox, and working through the processes involved in creating the Virtual Machine on the hypervisor.

Following the installation of the virtual machines, there is an initial basic setup and testing of the environment to ensure it behaves in the expected manner. All the images of the virtual machines can be downloaded from the hypervisor and saved for the re-deployment of the cyber range environment later on. Of course, if they were previously running in the production environment, the process would be the exact same, only with the difference that instead of downloading the storage images of the machines, they would need to be Cloned and then restored through the usage of the snapshots in another Virtual Machine instance, most possibly in a different node without affecting the current virtual machines.

In regard to Proxmox hypervisor, which uses the QEMU KVM engine, the Web User Interface allows the user to see all registered virtual machines right at the left side of the Web Interface, as can be viewed in the image below.

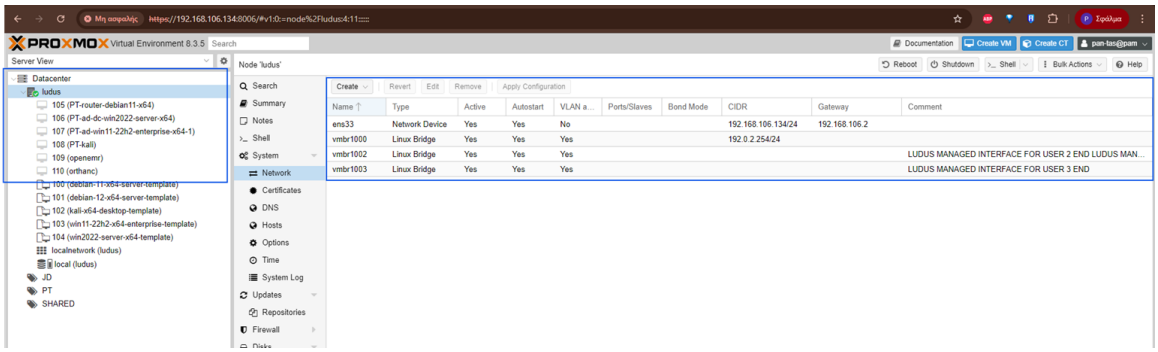


Figure 19: Proxmox Web User Interface

3.10 Ludus Templates

Templates are pre-configured virtual machines (base VMs) that can be used to deploy our environment. We can build the templates by running the following command:

```
>ludus templates build
```

3.10.1 Build Templates

Before we can deploy a range, we first must build the template VMs (base VMs without customization) that will be used in our range.

Templates are the basis of every VM deployed by Ludus. Unlike other solutions, Ludus templates are built from scratch (ISO), and by design do not contain any customization. This allows users to modify base templates into arbitrary VMs during a deploy without having to maintain a library of stale, customized VMs. This focus on infrastructure as code allows Ludus users to create fresh, up to date VMs every deployment.

The first step is to start the template build process. First, we can view the available templates.

```
+-----------------------------------+-------+
|              TEMPLATE             | BUILT |
+-----------------------------------+-------+
| debian-11-x64-server-template     | FALSE |
| debian-12-x64-server-template     | FALSE |
| kali-x64-desktop-template         | FALSE |
| win11-22h2-x64-enterprise-template | FALSE |
| win2022-server-x64-template       | FALSE |
+-----------------------------------+-------+
```

Image 1: Default available Ludus templates

We can build the templates by running the following command:

```
>ludus templates build

[INFO] Template building started - this will take a while. Building 1
template(s) at a time.
```

On a fresh installation, no templates are built. Ludus will build them from ISO files (with checksums) with the following command.

To monitor the progress of the build, execute this command to view the logs in real time:

```
>ludus templates logs -f
```

3.10.2 Adding new templates

We can easily add new templates from the Ludus repository. Here is a list of available templates:

```
- commando-vm
- debian10
- flare-vm
- remnux
- rocky-8-x64-server
- rocky-9-x64-server
- ubuntu-20.04-x64-server
- ubuntu-22.04-x64-server
- ubuntu-24.04-x64-desktop
- win10-22h2-x64-enterprise
- win11-23h2-x64-enterprise
- win2012r2-server-x64
- win2016-server-x64
- win2019-server-x64
- win2019-server-x64-no-security-updates
```

To add a template, first clone the Ludus repository, navigate to the template's directory, and then use the ludus command to add the desired templates:

```
>git clone https://gitlab.com/badsectorlabs/ludus
>cd ludus/templates
>ludus templates add -d win2016-server-x64
>ludus templates add -d win2019-server-x64
```

```
root@ludus:~# git clone https://gitlab.com/badsectorlabs/ludus
Cloning into 'ludus'...
warning: redirecting to https://gitlab.com/badsectorlabs/ludus.git/
remote: Enumerating objects: 4076, done.
remote: Counting objects: 100% (118/118), done.
remote: Compressing objects: 100% (78/78), done.
remote: Total 4076 (delta 61), reused 73 (delta 39), pack-reused 3958 (from 1)
Receiving objects: 100% (4076/4076), 78.45 MiB | 65.00 MiB/s, done.
Resolving deltas: 100% (3012/3012), done.
root@ludus:~# cd ludus/templates
root@ludus:~/ludus/templates# ludus templates add -d win2016-server-x64
[INFO]  Successfully added template
root@ludus:~/ludus/templates# ludus templates add -d win2019-server-x64
[INFO]  Successfully added template
root@ludus:~/ludus/templates# ludus templates build
[INFO]  Template building started - this will take a while. Building 1 template(s) at a time.
root@ludus:~/ludus/templates# ludus templates list
+-----------------------------------+-------+
|             TEMPLATE              | BUILT |
+-----------------------------------+-------+
| debian-11-x64-server-template     | TRUE  |
| debian-12-x64-server-template     | TRUE  |
| kali-x64-desktop-template         | TRUE  |
| win11-22h2-x64-enterprise-template | TRUE  |
| win2022-server-x64-template       | TRUE  |
| win2016-server-x64-template       | FALSE |
| win2019-server-x64-template       | FALSE |
+-----------------------------------+-------+
root@ludus:~/ludus/templates#
```

Figure 20: Cloning the Ludus repository and initiating the build for the win2019-server-x64 template.

We can also monitor template builds using the Proxmox web GUI. It is available at **https://<ludus IP>:8006** and the credentials for the web GUI can be retrieved with ludus user creds get.

Once all the templates have been built, we can deploy a range.

3.11 Deploying a range

Ludus ranges (environments) are defined by a single yaml configuration file. By default, a simple range configuration is provided to each user.

To view our current configuration, run the following command:

```
>ludus range config get
```

The configuration for this range is below:

```
debian@ludus:~$ ludus range config get
ludus:
  - vm_name: "{{ range_id }}-ad-dc-win2022-server-x64"
    hostname: "{{ range_id }}-DC01-2022"
    template: win2022-server-x64-template
    vlan: 10
    ip_last_octet: 11
    ram_gb: 8
    cpus: 2
    windows:
      sysprep: false
    domain:
      fqdn: ludus.domain
      role: primary-dc
  - vm_name: "{{ range_id }}-ad-win11-22h2-enterprise-x64-1"
    hostname: "{{ range_id }}-WIN11-22H2-1"
    template: win11-22h2-x64-enterprise-template
    vlan: 10
    ip_last_octet: 21
    ram_gb: 8
    cpus: 2
    windows:
      install_additional_tools: true
      office_version: 2019
      office_arch: 64bit
    domain:
      fqdn: ludus.domain
      role: member
  - vm_name: "{{ range_id }}-kali"
    hostname: "{{ range_id }}-kali"
    template: kali-x64-desktop-template
    vlan: 99
    ip_last_octet: 1
    ram_gb: 8
    cpus: 1
    linux: true
    testing:
      snapshot: false
      block_internet: false
network:
  inter_vlan_default: REJECT
  rules:
    - name: Only allow windows to kali on 443
      vlan_src: 10
      vlan_dst: 99
      protocol: tcp
      ports: 443
      action: ACCEPT
    - name: Only allow windows to kali on 80
      vlan_src: 10
      vlan_dst: 99
      protocol: tcp
      ports: 80
      action: ACCEPT
    - name: Only allow windows to kali on 8080
      vlan_src: 10
      vlan_dst: 99
      protocol: tcp
      ports: 8080
      action: ACCEPT
    - name: Allow kali to all windows
      vlan_src: 99
      vlan_dst: 10
      protocol: all
      ports: all
      action: ACCEPT
```

If we would like to make changes to this file, save it to disk, modify it, and set it with the following commands.

```
>ludus range config get > ludus-range-config.yml
<vim|nano> config

>ludus range config set -f ludus-range-config.yml
[INFO]  Your range config has been successfully updated.
```

Once we are satisfied with our range configuration, deploy it with range deploy.

```
>ludus range deploy

[INFO]  range deploy started
```

Just like with templates, we can tail the ansible logs for range deployment with :

```
user@ludus:~$ ludus range logs -f
PLAY [Pre run checks] ***************************************************************

TASK [Acquire session ticket] *******************************************************
changed: [localhost]

TASK [Check for valid dynamic inventory] ********************************************
ok: [localhost] => {
    "changed": false,
    "msg": "Dynamic inventory loaded!"
}
...
```

We can also check the status of our range at any time with range status

```
debian@ludus:~$ ludus range status
+---------+---------------+------------------+---------------+-------------------+-----------------+
| USER ID | RANGE NETWORK | LAST DEPLOYMENT  | NUMBER OF VMS |                   | TESTING ENABLED |
+---------+---------------+------------------+---------------+-------------------+-----------------+
|   PT    |  10.3.0.0/16  | 2025-04-05 03:54 |       5       |                   |      TRUE       |
+---------+---------------+------------------+---------------+-------------------+-----------------+
+------------+-----------------------------------+-------+------+
| PROXMOX ID |              VM NAME              | POWER |  IP  |
+------------+-----------------------------------+-------+------+
|    105     | PT-router-debian11-x64            |  Off  | null |
|    106     | PT-ad-dc-win2022-server-x64       |  Off  | null |
|    107     | PT-ad-win11-22h2-enterprise-x64-1 |  Off  | null |
|    108     | PT-kali                           |  Off  | null |
|    109     | openemr                           |  Off  | null |
+------------+-----------------------------------+-------+------+
```

After starting the machines via our ProxMox UI:

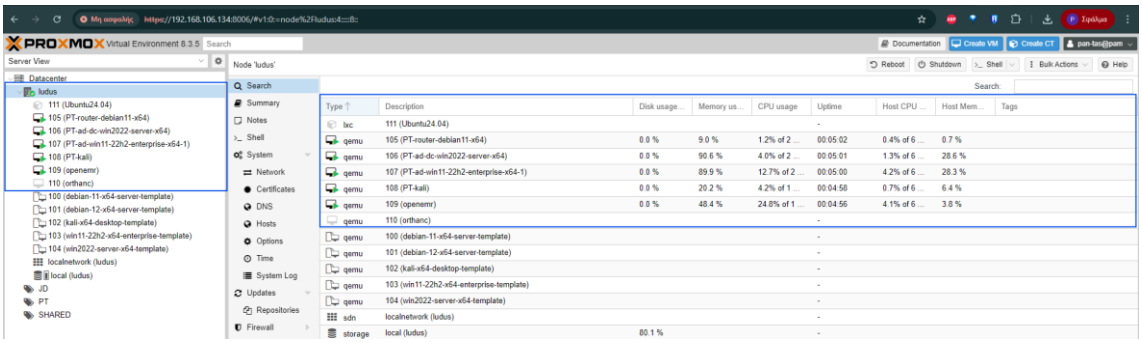


Figure 21: Proxmox VE 'ludus' node summary, showing running VMs and their resource consumption.

Once complete, if we run ludus range status, we will see an output with Deployment Status as SUCCESS. If we get errors, we can run ludus range errors, and it will print out what it failed on.

```
debian@ludus:~$ ludus range status
+---------+---------------+------------------+---------------+--------------------+------------------+
| USER ID | RANGE NETWORK | LAST DEPLOYMENT  | NUMBER OF VMS |                    | TESTING ENABLED |
+---------+---------------+------------------+---------------+--------------------+------------------+
|   PT    |  10.3.0.0/16  | 2025-04-05 03:54 |       5       |      SUCCESS       |       TRUE       |
+---------+---------------+------------------+---------------+--------------------+------------------+
+------------+-------------------------------------+-------+-------------+
| PROXMOX ID |              VM NAME                | POWER |     IP      |
+------------+-------------------------------------+-------+-------------+
|    105     | PT-router-debian11-x64              |  On   | 10.3.10.254 |
|    106     | PT-ad-dc-win2022-server-x64         |  On   | 10.3.10.11  |
|    107     | PT-ad-win11-22h2-enterprise-x64-1   |  On   | 10.3.10.21  |
|    108     | PT-kali                             |  On   | 10.3.99.1   |
|    109     | openemr                             |  On   | null        |
+------------+-------------------------------------+-------+-------------+
```

3.12 Testing the servers of Deployment

After a successful installation, we can log in to DC01 using the username: domainadmin and the password: password. Once logged in, we navigate to the Exchange Center.

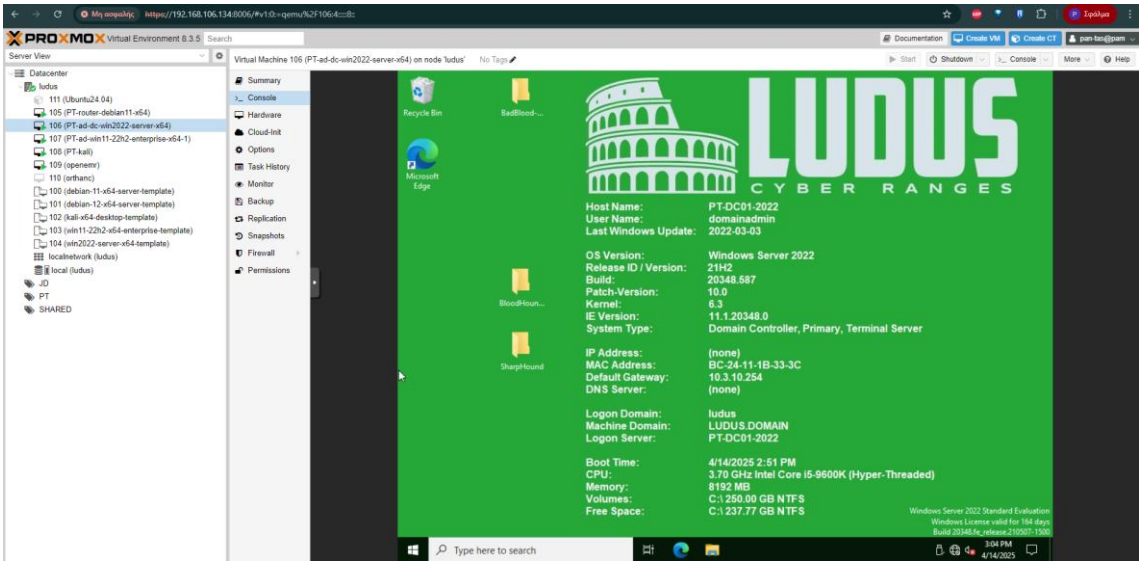


Figure 22: Proxmox VE console for the Active Directory Domain Controller (VM 106) displaying the Ludus environment system details.

While in testing mode, the desktop wallpaper for Windows machines will change from red to green. The green wallpaper indicates that network traffic is being blocked. This is accomplished by a PowerShell script that checks if:

1. There is a ping response from 8.8.8.8
2. There is a 200 response from a GET to http://captive.apple.com
3. There is a 200 response from a GET to https://google.com

If all checks fail, the background is set to green.
This PowerShell script is set to run on logon
in HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run as the task bginfo.

3.13 Resource Footprint and Scalability

The environment was deployed on a Debian 12 host configured with 26 GB of RAM, six CPU cores and 200 GB of storage, as described in Section 3.1.1. It is worth noting that this allocation is below the 32 GB of RAM recommended for Ludus, and that the host itself was executed as a virtual machine rather than on bare metal, which introduces an additional layer of virtualization overhead. The environment nevertheless deployed and operated successfully, which is itself an indication of the modest demands of the architecture. As reported by the range status output and the node summary presented in Figure 21, the deployed range comprised five virtual machines. Their individual allocations and observed consumption are summarized in Table 3.1.

VM (Proxmox ID)	Role	vCPU	Share of host RAM
PT-router-debian11-x64 (105)	Gateway and inter-VLAN routing	2	0.7%
PT-ad-dc-win2022-server-x64 (106)	Domain Controller (ludus.domain)	2	28.6%
PT-ad-win11-22h2-enterprise-x64-1 (107)	Enterprise workstation	2	28.3%
PT-kali (108)	Attacker node	1	6.4%
openemr (109)	Clinical application server	1	3.8%
Total (5 VMs)	Complete range instance	8	≈ 67.8%

Table 3.1: Measured resource footprint of a single range instance (source: Figure 21).

Three properties of the deployment follow these measurements. First, the aggregate memory footprint of a complete range instance is approximately 68% of the 26 GB host, corresponding to roughly 17.6 GB, or an average of some 3.5 GB per virtual machine. Second, the eight allocated virtual CPUs exceed the six physical cores of the host, representing an over-commitment ratio of approximately 1.33 to 1; this is a standard virtualization practice, viable because the workloads are idle between exercises, and it demonstrates that processor capacity is not the binding constraint in this configuration. Third, and most significantly for scaling, the local datastore was observed at 80.1% utilization, corresponding to approximately 160 GB of the available 200 GB once the base templates and the five virtual machine disks are accounted for. Storage, rather than memory or processing capacity, is therefore the resource that saturates first.

From this baseline an indicative projection can be constructed for the operation of multiple concurrent instances of the range. The projection assumes that memory and processor requirements scale approximately linearly with the number of instances, while storage grows sub-linearly, since the base templates are provisioned once and shared across all instances, with only the per-instance virtual machine disks, estimated at approximately 100 GB per instance, being replicated. Table 3.2 presents the resulting estimates together with an indicative host specification for each tier.

Concurrent instances	VMs	RAM (GB)	vCPU	Storage (GB)	Indicative host specification
1 (validated)	5	≈ 18	8	≈ 160	26–32 GB RAM, 6 cores, 200 GB — as deployed
2	10	≈ 35	16	≈ 260	64 GB RAM, 8–12 cores, 512 GB NVMe
4	20	≈ 70	32	≈ 460	128 GB RAM, 16–24 cores, 1 TB NVMe
8	40	≈ 140	64	≈ 860	256 GB RAM, 32–48 cores, 2 TB NVMe

Table 3.2: Indicative projection of infrastructure requirements for concurrent range instances.

Two qualifications must accompany this projection. It is a first-order derived from a single measured instance, not an empirically validated multi-instance benchmark, and the actual figures would be expected to deviate in both directions. Several mechanisms would reduce consumption below the linear estimate: Kernel Same-page Merging deduplicates identical memory pages across virtual machines running the same operating system, which is precisely the case when

multiple copies of the same Windows-based range are executed concurrently, while thin provisioning and linked clones further reduce the effective storage cost of each additional instance. Conversely, storage input and output contention constitutes the most probable practical bottleneck, since the simultaneous boot of several dozen virtual machines produces a demanding random-access workload, which is the reason a fast NVMe device is specified in every tier of Table 3.2 rather than conventional rotational storage.

The practical significance of these figures is that the entire environment operates within the capabilities of a single workstation-class machine, and that supporting a small class of trainees, in the order of four to eight concurrent instances, remains within the range of a single mid-tier server rather than requiring datacenter infrastructure. This substantiates, in quantitative terms, the accessibility argument advanced in Section 1.10. It should nevertheless be stated explicitly that the multi-instance configurations presented in Table 3.2 were not deployed and measured within the scope of this work. The empirical validation of concurrent instances, through the deployment of multiple range configuration files and the measurement of deployment time, memory deduplication efficiency and storage input and output under simultaneous load, is identified as the immediate continuation of this work in Section 5.1.

4. Testing

Having completed the deployment of the Cyber Range infrastructure in the previous chapter, this chapter focuses on the practical validation of the environment. We populate the Active Directory with realistic data, execute attack scenarios covering the full cyber kill chain, and verify that the detection mechanisms correctly identify and log malicious activity.

4.1 Populating the Active Directory Environment

Having the core infrastructure deployed successfully within the Ludus cyber range, with the infrastructure including the Domain Controller (DC) and an effective Active Directory (AD), the next step would be to move the lab from a clean environment, an out-of-the-box installation, to an environment reflective of an established, active corporate network infrastructure.
When installed in the default mode of Active Directory, there tend to be very few users and groups. However, such an installation does not represent reality, since the environment of the company, which might be in operation for years, tends to be complex with respect to permissions, group membership, and legacy, with thousands of objects such as users, computers, and groups.
To reproduce such realistic levels of complexity, the **BadBlood** tool was used [29]. BadBlood happens to be a PowerShell tool designed specifically for the purpose of importing a massive random object set into an Active Directory domain. Not only this, but the tool also generates thousands of users, groups, and Organizational Units (Ous), the most important aspect being the injection of a multitude of dangerous Active Directory misconfigurations, which instantly turns the pristine lab environment into a high-fidelity target, reflecting the levels of "organizational drift" seen within prod environments.

Cyber Attack Phases

The primary objective of this population step is the development of a valid target environment for assessment and analysis of contemporary attack themes. The misconfigurations and intricate relationships spawned by BadBlood are not random; they are crafted to be identified and exploited and thus directly support a multitude of phases in the cyberattack process after the initial access phase.
Such an environment becomes ready for an attack simulation, especially concentrating on the phases below:

1. **Reconnaissance (Internal):** This represents the first step an attacker takes after having established the first level of their presence inside the target's infrastructure, for example a compromised user account. In the context of an AD infrastructure, such an activity entails the utilization of tools such as BloodHound and PowerView for the purposes of mapping the domain[35]. BadBlood represents the very information necessary for the execution of such tools since such information would never be established within an AD infrastructure.
2. **Privilege Escalation:** The intricate pattern of group permissions, nested group membership, and improper ACLs established through BadBlood makes it easy for an attacker to achieve privilege escalations. A very easy example would be the presence of a lower privilege level username that belongs to a group having the right to write permissions for the higher privilege level group/user object. An attacker would be eager to identify and utilize such routes, an integral component of this thesis's research.
3. **Lateral Movement:** Once privilege escalation has occurred, the attacker's intention is to move from computer to computer in order to locate high-value information manifested by seizing the control of the entire domain. Information obtained from BadBlood, for example, the presence of certain users within local administrators' groups in various workstations, serves as the 'map' the attacker uses during lateral movement, for example, accessing the workstation of the server's administrative computer.

4.2 BadBlood Tool

BadBlood is a PowerShell tool that populates our Active Directory with thousands of randomized users, groups, computers, and complex permissions (ACLs). This simulates an active, complex, real-world corporate environment, which is perfect for testing detection, enumeration (like with BloodHound), and attack paths.

At this moment we have established the Cyber Range with Domain controller let it populate with BadBlood. We run BadBlood on a domain so that security analysts and engineers can practice using tools to gain an understanding and prescribe to securing Active Directory. Each time this tool runs, it produces different results. The domain, users, groups, computers and permissions are different.

The output of BadBlood is a domain similar to one found in the real world.

Use it to:

1. Start a journey into privileged identity threat hunting.
2. Build a test domain
3. Test vendor software

Getting started:

1. Download BadBlood from https://github.com/davidprowe/BadBlood
2. Extract
3. Run BadBlood ./badblood/invoke-badblood.ps1

The Invoke-BadBlood.ps1 PowerShell script is at the root of the BadBlood repo. A few prompts appear designed to ensure this is only run in a testing or educational environment.

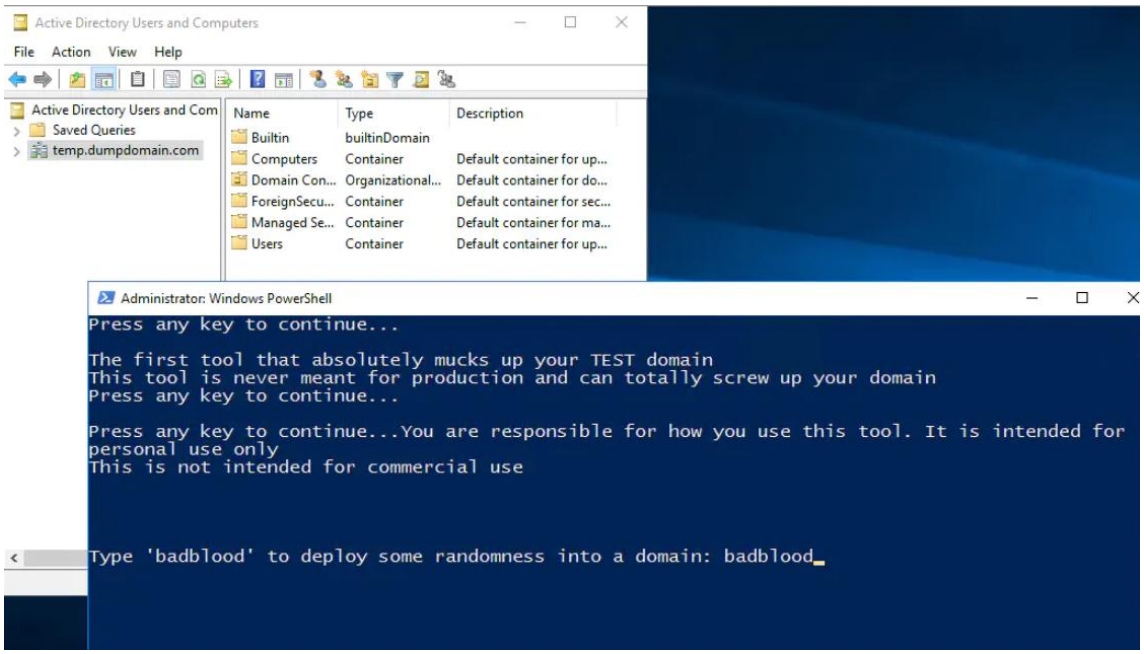


Figure 23: Initializing the BadBlood tool to populate Active Directory domain.

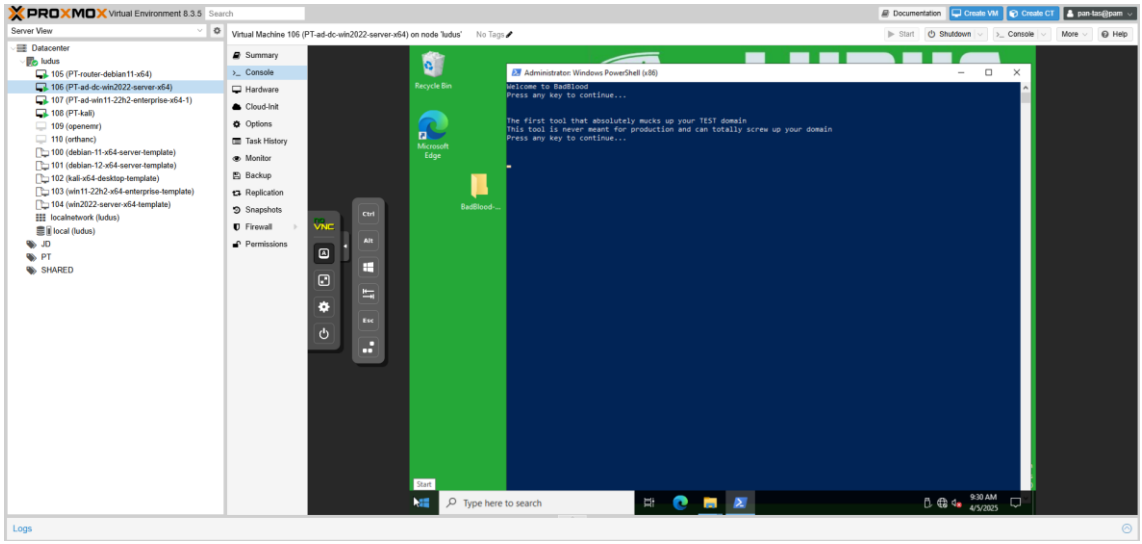


Figure 24: Preparing BadBlood tool execution on the Active Directory within Windows Server 2022

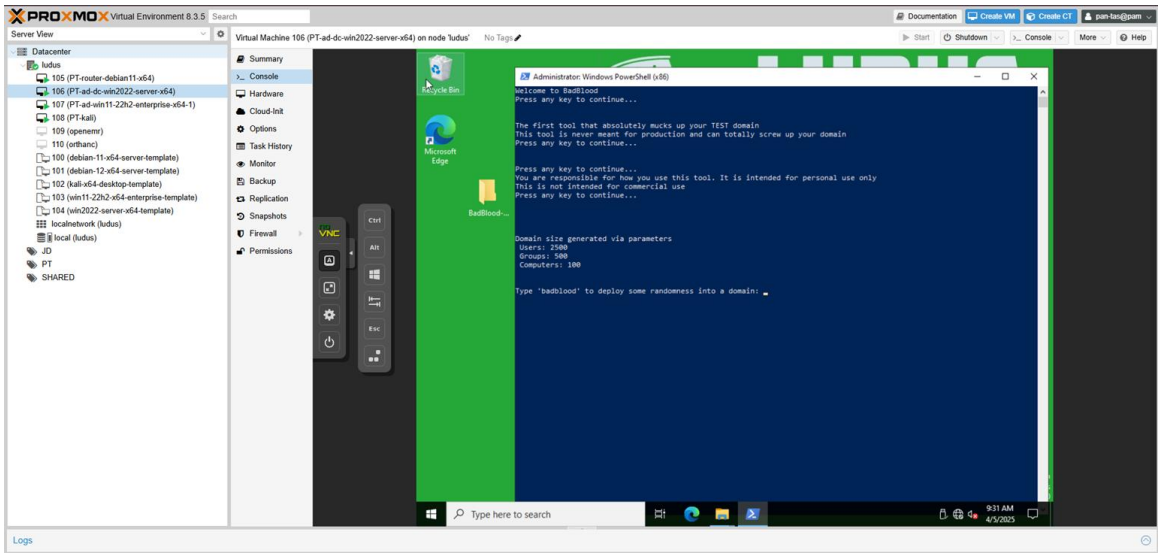


Figure 25: Preparing the BadBlood tool execution on the Active Directory

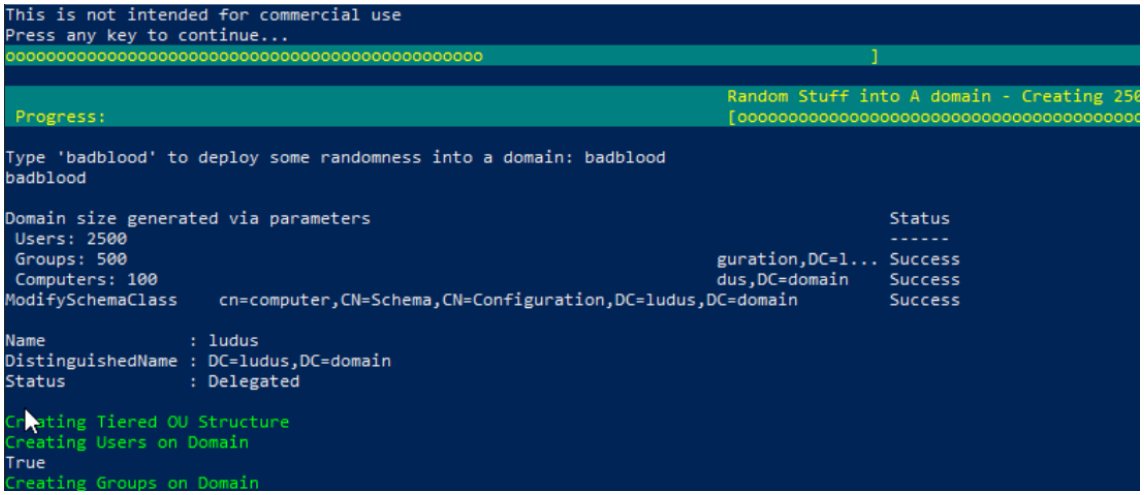


Figure 26: Execution progress of the BadBlood tool

4.2.1 LAPS installed

Badblood begins by expanding the current domain's schema by installing LAPS:

```
AD_Laps_install\InstallLAPSSchema.ps1
```

LAPS is Microsoft's free tool designed to remediate lateral movement across a domain by randomly generating passwords for the admin user on computer objects.

4.2.2 Organizational Unit Structure Created

After extending the schema, BadBlood creates a new OU structure on our domain. The full details of this expansion can be viewed in the file:

```
AD_OU_CreateStructure\CreateOUStructure.ps1
```

BadBlood can customize the OUs that are added to the OUs I call "Top Level OUs". The detailed sub-structure of the OUs can be expanded by modifying the variables in the createoustructure.ps1. The first release of the tool creates an OU structure similar to the one depicted below.
BadBlood uses this new OU structure to randomize the placement of objects and permissions.

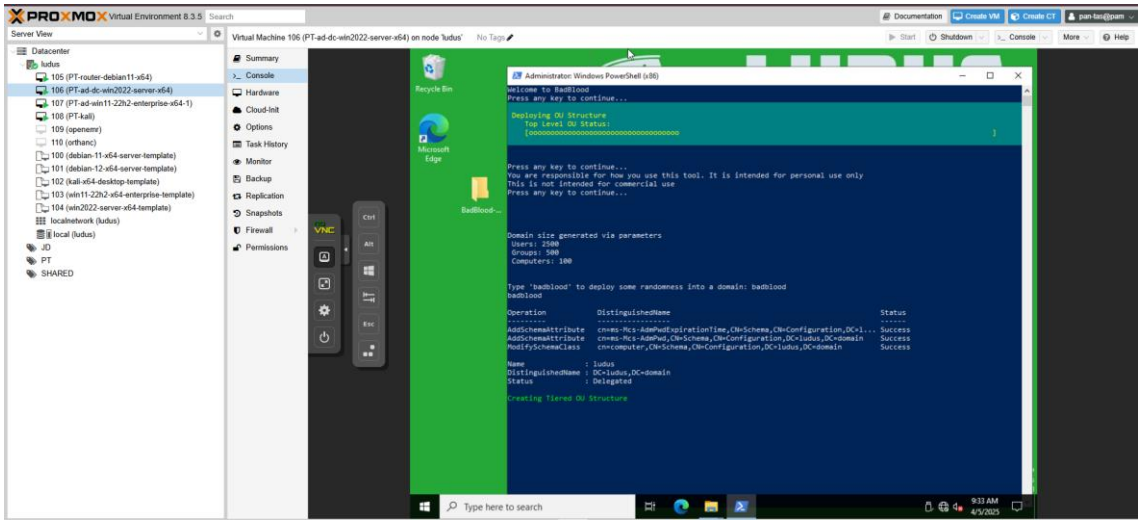


Figure 27: Monitoring the BadBlood tool's execution via the Proxmox VNC console

4.2.3 Users Created

After the structure is complete, BadBlood begins to create a random number of users (500-10000) into the domain. This process is designed in the:

```
AD_Users_Create\CreateUsers.ps1
```

During creation of each user BadBlood randomly selects an OU or container and places that person in a random path. The tool generates very random male and female users based on the text files located in the \Names folder under the AD_Users_Create folder.
Also, each user is created with a unique password.
The tool fills in a bit of information on the user account and moves onto the next step: Groups.

4.2.4 Groups Created

After the users are complete, the tool moves on to creating the groups in the domain. This is performed by:

```
AD_Groups_Create\CreateGroups.ps1
```

The groups are randomly named in this script pulling information from a hotmail.txt file located in the same folder as the script. The groups, just like the users, are randomly placed in random OUs and Containers in the domain.

4.2.5 Computers Created

The tool now moves onto creating computers by calling:

```
AD_Computers_Create\CreateComputers.ps1
```

These computers also have randomly generated names. Like the other objects previously created the computers are placed in random OUs.

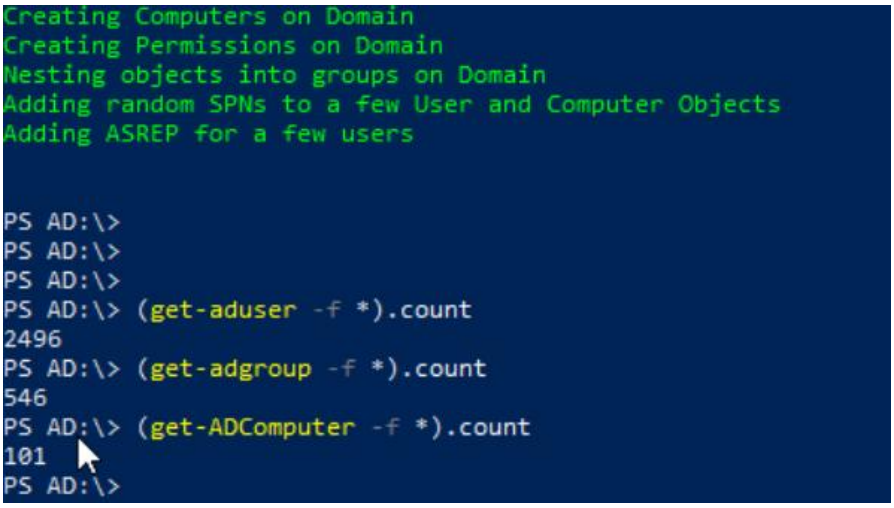


Figure 28: Verifying Active Directory object counts with PowerShell after BadBlood execution.

4.2.6 ACLs Generated

This is the Active Directory DACLs permission part and a generator to create random permissions every time this tool is run. No matter what, this tool adds very random and very invasive permissions.

The permission generator script is :

```
AD_Permission_Randomizer\GenerateRandomPermissions.ps1
```

This script calls and imports the functions from the folder AD_OU_SetACL. There are many scripts in this folder, with many functions inside of each script. They can be used to automate a lot of Active Directory permission tasks an admin might encounter during the workday.
In BadBlood, I call upon these scripts to populate and stress the Active Directory domain.

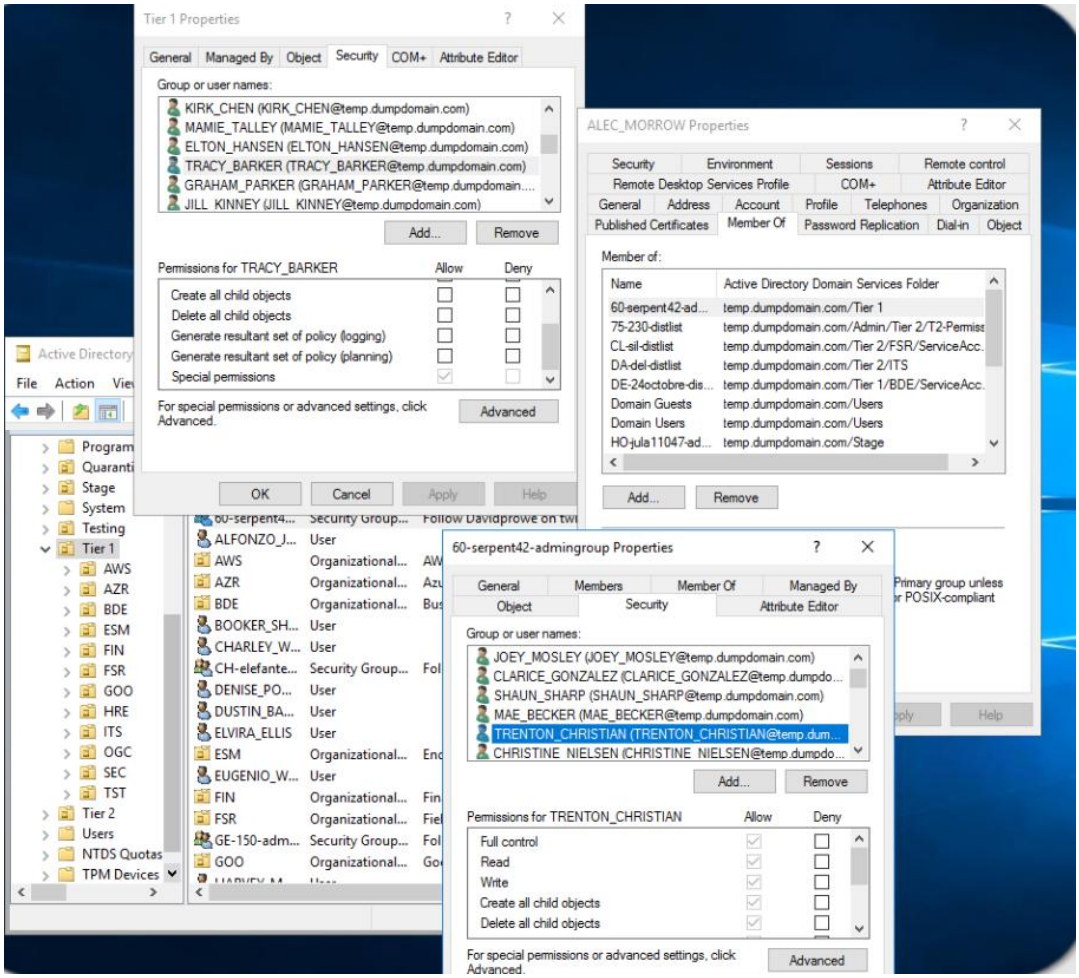


Figure 29: Inspecting the complex Active Directory OUs, users, and group memberships generated by BadBlood.

We have successfully:

- Created the Infrastructure: Our Ludus DC.
- Populated the Environment: Our BadBlood run created 2,496 users, 546 groups, and 101 computers.

Our next step is to visualize the attack paths we just created. BadBlood's entire purpose is to create hidden relationships and misconfigurations. The standard tool to find them is **BloodHound**.[35]

4.3 BloodHound

Once we have finished with the above step, our Active Directory is no longer a clean, empty lab. It is now a realistic, complex, and vulnerable target, suitable for the objective of this thesis. Our next step after this will be to run an enumeration tool (like BloodHound) against our DC to visualize all the attack paths that BadBlood just created. [35]
BloodHound is an attack path enumeration tool which makes a lot requests to Active Directory. It uses graph theory to map all the relationships within Active Directory. For our thesis, it does three critical things:

1. It visualizes Attack Paths: Rather than our own efforts to locate a susceptible user out of 2,500, what the BloodHound does is allow us to view the actual graphical representation. We can clearly locate the start (a low-privileged user) and the destination (the “Domain Admins” group) and all the intervening hops. This graphical representation is essentially the evidence we need to support our theory.

2. It Identifies "Hidden" Vulnerabilities: A real-world network is not hacked by one single "critical" vulnerability. It is hacked by a *chain* of small, overlooked misconfigurations. BadBlood *created* these misconfigurations, and BloodHound *finds* them.

   Examples include:

   - A regular user who is in a group that has admin rights on a specific computer.
   - A computer that has a domain admin currently logged into it (letting us steal their credentials).
   - A user who has permission to change the password of a more privileged group.
   - Users vulnerable to "Kerberoasting" or "AS-REP Roasting" (which BadBlood specifically added).

3. It automates the Attacker's Hardest Job: Before tools like BloodHound, an attacker would have to spend days or weeks manually running commands (net user, net group, etc.) to find these relationships. BloodHound does it in minutes. We are simulating *exactly* what a modern, real-world attacker would do the moment they get inside a network.

4. What is the Essential Attack Phase?

Running BloodHound is an essential component of the Post-Exploitation phase of the assault process.

Using the MITRE ATT&CK Framework, the major tool for the Discovery (TA0007) category within BloodHound is Discovery.

Here starts the Internal Reconnaissance or Discovery phase. The tool used for this response to the third question is BloodHound. The environment is where the architecture scans for discovery (T1087 - Account Discovery, T1069 - Permission Groups Discovery, T1482 - Domain Trust Discovery), and then the route for Privilege Escalation (TA0004) and Lateral Movement (TA0008).[30]

So, we have run BadBlood to *create* the attack paths. We are now running BloodHound to *find* those attack paths. This allows us to realistically simulate an attacker's entire internal campaign, from a low-privilege foothold to full Domain Admin.

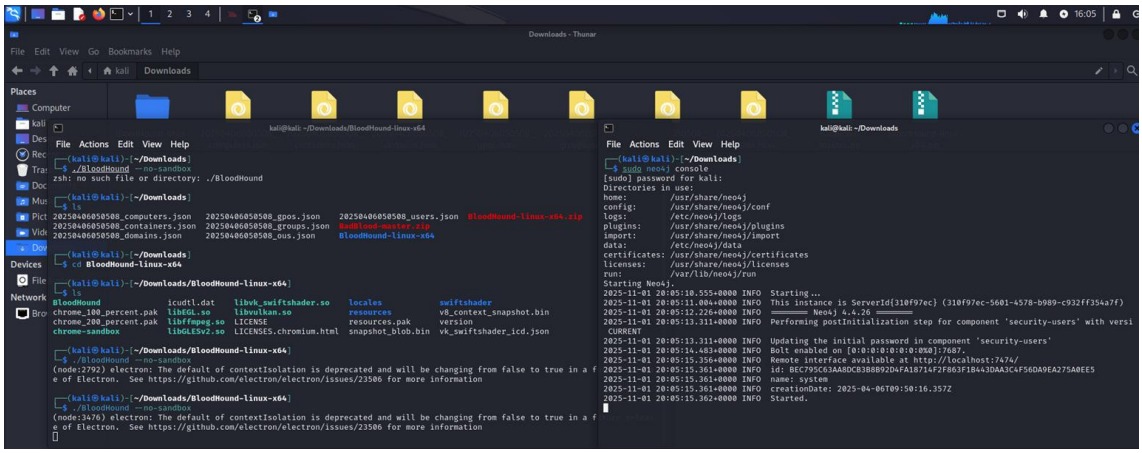


Figure 30: Launching the BloodHound application and its Neo4j database backend on the Kali Linux attacker VM.

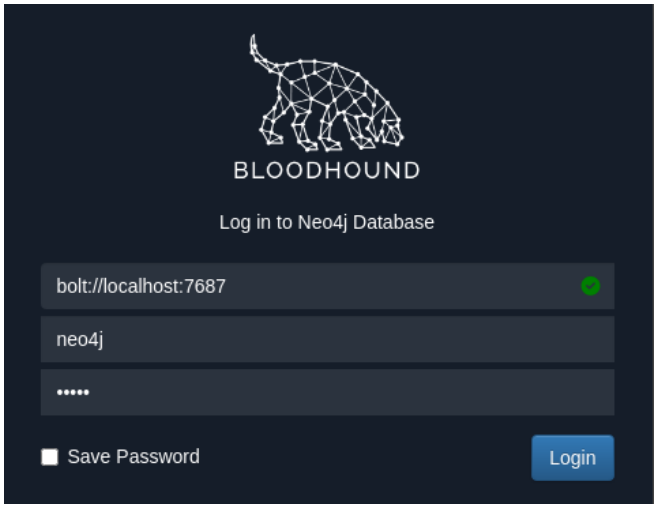


Figure 31: Neo4j database backend of BloodHound

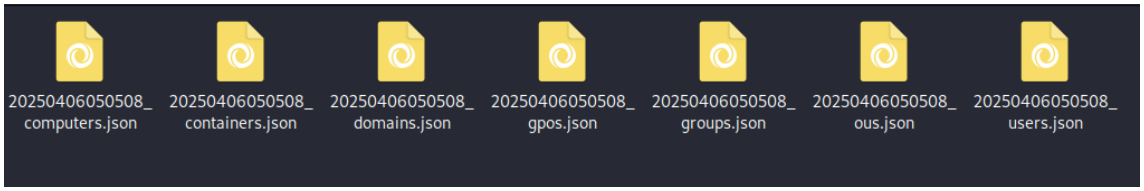


Figure 32: Neo4j database backend of BloodHound

Those exported data is the raw material for the next and most critical phase of the attack: Attack Path Analysis. These files will be ingested by the BloodHound Graphical User Interface (GUI), which will parse this data using graph theory to visually reveal the hidden and often unintended privilege escalation paths that BadBlood created.

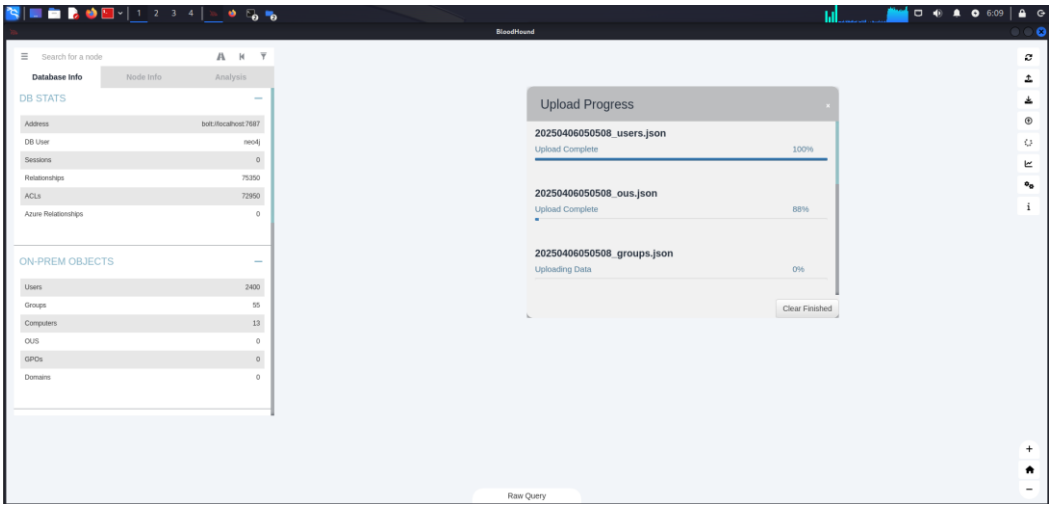


Figure 33: Uploading the collected Active Directory data (users, OUs, groups) into the BloodHound GUI for analysis.

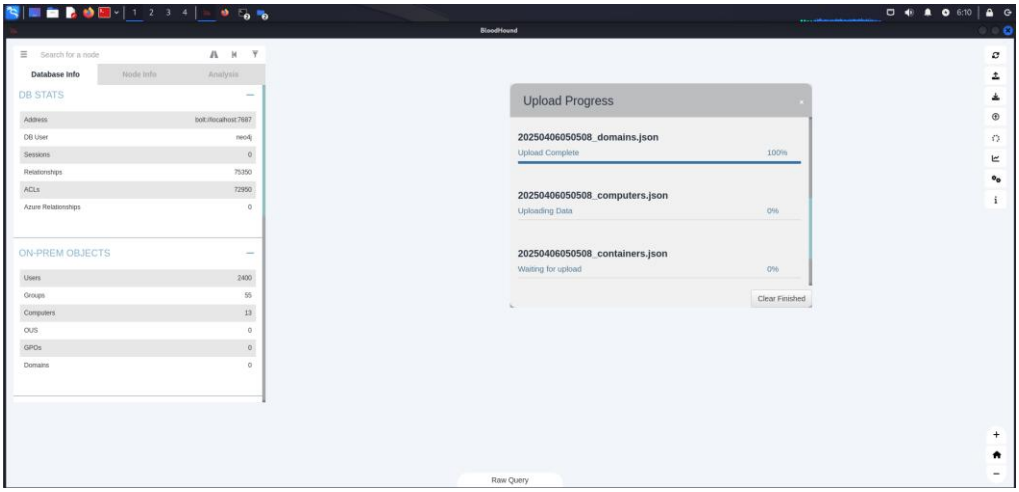


Figure 34: Importing the domains.json and computers.json data files into the BloodHound analysis database.

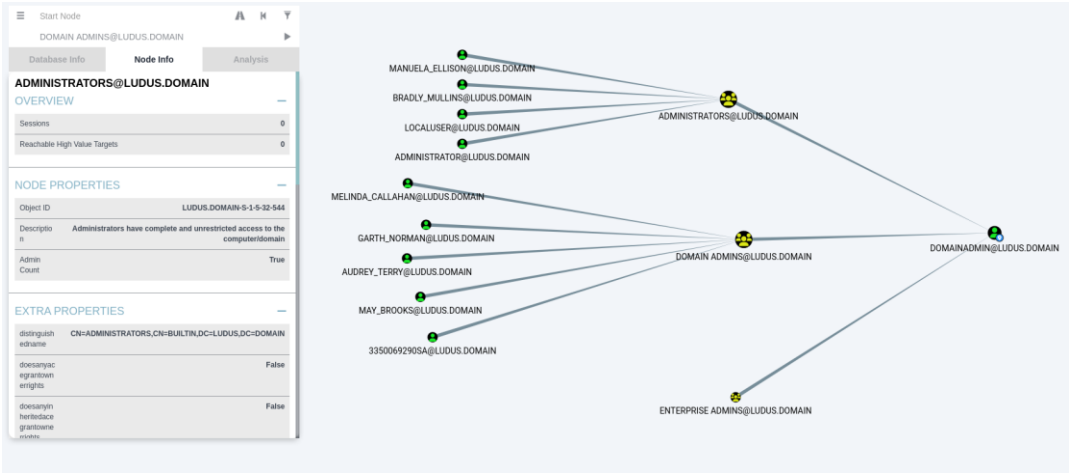


Figure 35: BloodHound visualization of privileged group relationships in the ludus.domain.

This above image represents the successful completion of the Internal Reconnaissance and Discovery phase of the attack. The data exported from the BadBlood-populated Domain Controller has been ingested into the BloodHound GUI, providing a visual guidance of the domain's structure.

Key Insights:

1. Visualizing the "Crown Jewels": The graph clearly maps the relationships between the LUDUS.DOMAIN's most critical-privilege groups, such as DOMAIN ADMINS (the ultimate target), ENTERPRISE_ADMINS, and the built-in ADMINISTRATORS group.

2. Identifying High-Value Targets: The core insight is the visualization of all members of these powerful groups. The BadBlood tool has successfully simulated a realistic, complex environment by populating these groups with numerous users, including MANUELA.ELLISON, BRADLY.MULLINS, GARTH.NORMAN, and many others.

3. Defining the Attacker's Goal: From an attacker's perspective, the objective is no longer abstract. This graph confirms that gaining control of any single one of these user accounts (Manuela, Bradly, etc.) is equivalent to becoming a Domain Admin.

In summary, this graph provides the complete attack plan. We have successfully mapped the "keys to the kingdom" and identified a wide attack surface of high-value user accounts. The next step is to use BloodHound's query engine to find the *shortest and easiest* attack path from a compromised, low-privilege user to any one of these targets.

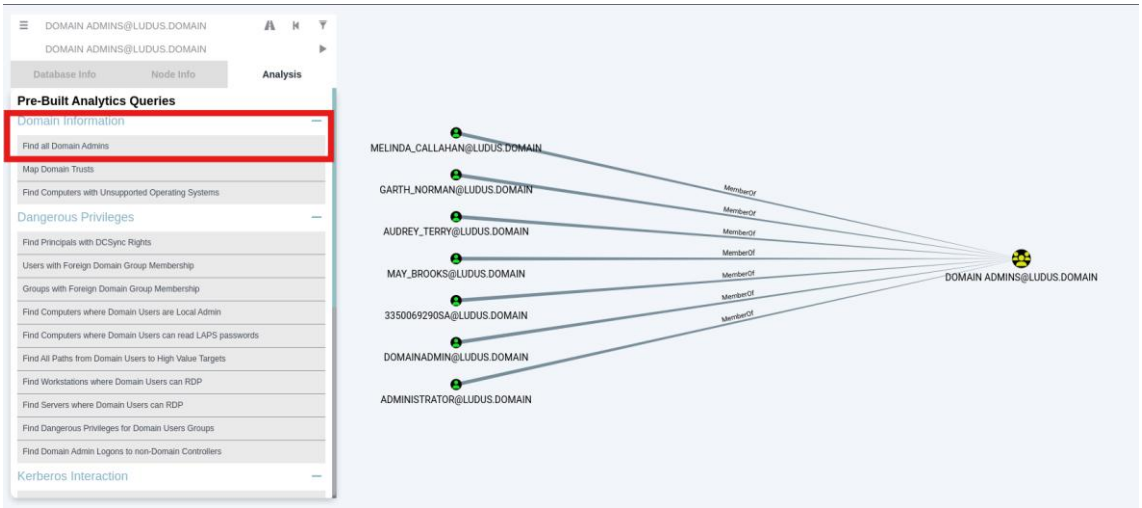


Figure 36: Using BloodHound's "Find all Domain Admins" query to map privileged users.

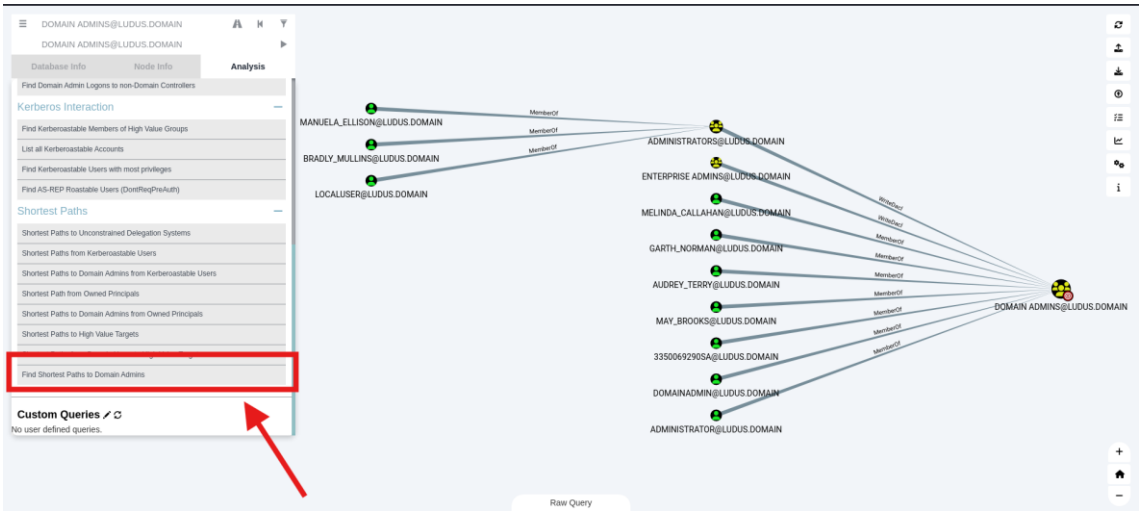


Figure 37: Selecting the "Find Shortest Paths to Domain Admins" query in BloodHound for attack path analysis.

Having successfully ingested the environment's data and identified the high-value targets (the members of the DOMAIN ADMINS group), the attack progresses to the Analysis phase. The objective is no longer just to see who a domain admin is, but to find a concrete, exploitable path from a position of low privilege to one of those high-value accounts.

In other words, the image depicts an Active Directory attack path analysis. The highlighted section (“Find Shortest Paths to Domain Admins”) shows a graph-based visualization that identifies the shortest privilege escalation routes from regular domain users to the Domain Admins group. Each node represents an account or group, while edges represent relationships or permissions (e.g., membership, delegation, or administrative rights).This query instructs BloodHound to analyze the entire graph database tens of thousands of relationships, permissions (ACLs), and user sessions created by BadBlood and calculate the most efficient privilege escalation chain available.

The underlying mindset is offensive security accomplished by means of the defender's awareness of enemy patterns in the manner of enemy analysis. Thus, the detection of high-privileged paths will allow attackers to focus their efforts on path-hardening activities, such as the reduction of unnecessary high-privileges.

Analysis of this kind is relevant to the objectives of a cyber range or cybersecurity training, in that it establishes the ways in which graph theory and attack path modeling can be utilized in simulating attacker movement in order to enhance Active Directory security.

4.4 Attack Vector Identification : Kerberoasting

Figure 38: Identifying Kerberoastable Attack Vectors in BloodHound

This image is utilized to represent the environment in which the Active Directory (AD) Kerberos interaction analysis is done, in relation to the query “List all Kerberoastable Accounts.” Each node in the above image symbolically represents the Active Directory user accounts with Service Principal Name (SPN) susceptible to the attack known as Kerberoasting. These accounts might be employed by hackers to obtain the Kerberos service tickets, which in turn might be brute-forced off-line to result in the plain text login details.

The conceptual approach in relation to visualization involves proactive threat hunting and the assessment of privilege exposure. This can be considered to represent the attacker’s viewpoint during the process of credential attack surface identification, offering defenders operational intelligence in terms of securing AD configurations. From the defender’s position, the analysis offered by the exercise helps in identifying weak or erroneous service accounts, advising the administrator on how to implement tougher password policies, control SPNs, or change the credentials periodically. This specific exercise in the cyber range offers practical application involving offensive security software such as BloodHound in relation to Kerberos.

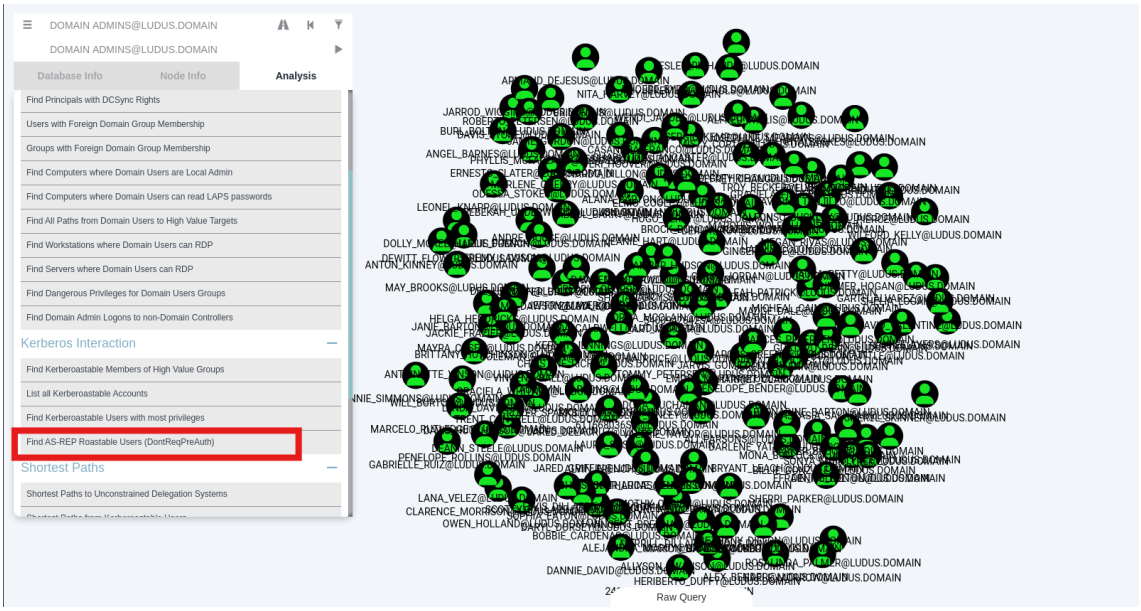


Figure 39: Identifying AS-REP Roastable Accounts in BloodHound

4.5 Attack Vector Identification : AS-REP Roasting

Prior to traversing the complex attack path, the attacker must first establish an initial foothold or find a means of obtaining escalated privileges in their current low-privileged environment. One of the most popular means of doing just that is by finding the misconfiguration in the Kerberos environment.

Referring to the image below, an in-built query in BloodHound, **"Find AS-REP Roastable Users (DontReqPreAuth),"** was run. This search returns all accounts in the domain with the "Kerberos Pre-authentication" attribute disabled.

Why This Is a Critical Vulnerability (T1558.004):

When pre-authentication is disabled, an attacker can ask for a Kerberos ticket (AS-REP) on behalf of any such user without providing any credentials. There is also a part of the ticket encrypted with the user's hashed NTLM password.

This offers the attacker a major opportunity:

1. **Hash Extraction:** The attacker can then utilize a tool, such as Impacket's GetNPUsers.py, in order to ask for the AS-REP tickets of all the above-mentioned users.

2. **Offline Cracking:** These obtained hashes of the tickets can be stored in a file and then cracked in an offline environment, by utilizing the Hashcat tool. These attacks do not involve direct interactions with the network; thus, they cannot cause potential lockouts.

3. **Gaining Foothold:** If any of the users in the current list have weak passwords, their password will be successfully cracked, yielding the first set of valid login credentials.

This stage in the attack chain is indeed very important, since it represents the "entry point" from which the more complex "Shortest Path to Domain Admins" attack is deployed.

4.6 Alternative Lab Provisioning tool LudusHound

LudusHound is a lab replication tool that transforms **BloodHound** data back into a fully functional, deployable **Ludus** range. The standard workflow for an attacker is to run SharpHound/BloodHound against a target network to collect data and find attack paths.

A major concern in establishing scalable cyber ranges is related to the automation of realistic and complex “target” network simulations and models. The initial step in any red team operation has to include establishing an Active Directory specifically, one with users, groups, and complete native misconfigurations through manual settings, which would not only consume far too many precious cycles but clearly fail to scale efficiently to meet the needs of on-demand net defense training requirements in any realistic or timely way. This is decidedly not sufficient, since any modern model of net defense needs to simulate real-world corporate networks' complex patterns of attack, not fictional ones not even remotely representative of real-world corporate network data sets in any realistic manner.

The key to overcoming current concerns with regard to the automation of realistic and complex “target” network simulations and models would be to identify and apply relevant automation tools to better facilitate real-world net defense models in direct accordance with real-world corporate network data sets. There is one key automation tool in particular, LudusHound (bagelByt3s, 2024), that very directly addresses said concerns between “Theoretical Breakthroughs” and “Simulation Implementation Detail” by directly assisting in the automation process in very real-world scalable terms to specifically simulate real-world corporate network data sets on an appropriately realistic attack graph in accordance with real-world corporate networks and cybersecurity professionals' needs. The Ludus lab automation framework integrates “BloodHound” data concerning real-world attack graph mappings to automatically set up “A complete Active Directory environment” with full functionalities in scalable terms for either an entire prod Active Directory network or isolated sections of it on an attack graph level directly reflective of real-world defense simulations.

The workflow for LudusHound is:

1. **Data Acquisition:** A security team (red or blue) runs SharpHound on a *real production* network and collects the resulting .zip file.
2. **Configuration Generation:** This .zip file, which is just a map of the network's relationships, is fed into LudusHound. The tool parses this data and automatically generates a complete Ludus configuration file(.yml).
3. **Lab Replication:** A new configuration file is then used to establish a new Ludus scale. This creates an isolated, functional, and safe Active Directory environment, which is an exact replica of the real environment in terms of users, groups, permissions, and most importantly, the attack paths.

The importance of LudusHound lies in its ability to enable safe, realistic, and specific "what-if" scenarios for both offensive and defensive teams.

- For the Red Team (attackers): An attacker can harvest data of the live target and then create a functional Cyber Range of the targeted network with the help of LudusHound. They can then practice their attacks in a safe environment and confirm if their attack vector will succeed without even getting detected in the production environment. They will be able to simulate and refine their attack vectors before executing them in the production environment.
- For Blue Teams (Defenders): This is perhaps even more powerful. A blue team can run BloodHound on their own network, feed the data to LudusHound, and build a replica of their *own* environment. In this replica, they can:
    - Validate Attack Paths: Ensure the attack path identified by BloodHound as possible in theory is actually exploitable.

- Test Remediation: Attempt to remediate the attack path (for example, “What if we remove this user from this group?”) and then run BloodHound to quickly confirm the attack path is eliminated.
- Train Incident Responders: Utilize the replica lab to train analysts on identifying attacks relevant to the company's specific misconfigurations.

4.6.1 Installation

Clone the LudusHound repo and install Ansible Roles.

```
git clone https://github.com/bagelByt3s/LudusHound /opt/LudusHound

ludus ansible collection add
https://github.com/bagelByt3s/LudusHound/raw/refs/heads/main/Collections/bagelByt3s-
ludushound-1.0.0.tar.gz
```

Build the LudusHound binary.

```
cd LudusHound
go build
```

Running LudusHound

The following is an example of how to run the tool. LudusHound must have network access to our BloodHound CE server.

```
./LudusHound
--Server 127.0.0.1 \
--User neo4j \
--Pass password \
--Output LudusRanges/LudusHound.yml \
--AliveComputers
Titan.Ghost.Local,Eclipse.Child.Ghost.Local,Avalanche.Specter.Local,Raven.Ghost.Local,Nova.Chi
ld.Ghost.Local,Mirage.Specter.Local
```

This is a breakdown of the command arguments:

```
--Server: The IP address of the pupulated BloodHound server
--User: The username of a valid BloodHound account
--Pass: The password to the BloodHound user
--Output: The Ludus range configuration file that will be generated
--AliveComputers: The computers that will be created and deployed as live systems
```

This is what it looks like to run the tool.

```
root@ludus:/opt/LudusHound# ./LudusHound
--Server 127.0.0.1 \
--User neo4j \
--Pass password \
--Output LudusRanges/LudusHound.yml \
--AliveComputers
Titan.Ghost.Local,Eclipse.Child.Ghost.Local,Avalanche.Specter.Local,Raven.Ghost.Local,Nova.Chi
ld.Ghost.Local,Mirage.Specter.Local

Server: 127.0.0.1
User: neo4j
Password: password
Output: LudusRanges/LudusHound.yml
Directory to save BloodHound Files: ./Tmp/2025-06-06_20-16-53
Domain Identified: SPECTER.LOCAL
Retreiving AD Objects for SPECTER.LOCAL
Domain Identified: GHOST.LOCAL
Retreiving AD Objects for GHOST.LOCAL
Domain Identified: CHILD.GHOST.LOCAL
Retreiving AD Objects for CHILD.GHOST.LOCAL
Retrieving object relationship mapping for all domains and saving to:
./Tmp/2025-06-06_20-16-53/Relationships/Relationships.json
Data successfully saved to ./Tmp/2025-06-06_20-16-53/filesMap.json
Generating Ludus YML Config:
Creating config for TITAN.GHOST.LOCAL
Creating config for ECLIPSE.CHILD.GHOST.LOCAL
Creating config for AVALANCHE.SPECTER.LOCAL
Creating config for RAVEN.GHOST.LOCAL
Creating config for NOVA.CHILD.GHOST.LOCAL
Creating config for MIRAGE.SPECTER.LOCAL
Writing Ludus Range to LudusRanges/LudusHound.yml
```

The next step is to provide the generated Ludus range configuration file to Ludus and deploy the range.

```
ludus range config set -f LudusRanges/LudusHound.yml
ludus range deploy
```

After a couple of hours, we will have a Ludus range with the same domain trusts, user, computer, and group objects, GPO's, OU's, sessions and relationships as the production environment.

4.7 Detection Mechanism Wazuh

For the detection and monitoring phase of this research, we have selected **Wazuh**, a unified open-source Security Information and Event Management (SIEM) and Extended Detection and Response (XDR) platform. Wazuh was chosen for its comprehensive ability to aggregate and analyze log data from Active Directory and Linux machine where open source medical OpenEMR has been hosted. By integrating Wazuh with **Sysmon** (System Monitor), we can capture granular telemetry required to identify the specific identity-based attacks simulated in this study, such as Kerberoasting and AS-REP Roasting and also capturing RCE (remote code execution in OpenEMR virtual machine) . Its rule-based engine allows for the customization of detection logic, enabling the precise identification of malicious indicators and the correlation of events across the simulated network structure.

Critical Rationale for the Method of Choice:

- **Unified Capabilities:** Offers the capabilities of log analysis, file integrity monitoring, and intrusion detection in one agent.
- **Open Source:** Available for research purposes and largely utilized in the industry. This increases the relevance of the research findings.
- **Customizability:** Facilitates the development of tailored rules (in XML-based) to identify precise artifacts produced by tools like BadBlood and BloodHound.

Logs Data Analysis

Analysis of log data refers to a basic process consisting of the evaluation or extraction of meaningful information from the log files generated by different systems, applications, or devices. The log files contain events providing information necessary for the evaluation of troubleshooting, security analysis, or monitoring of processes. Alternatively, analysis of log data refers to the best practice necessary for a secured efficient and robust information technology environment.

The system gathers, analyzes, and stores the logs. The Wazuh agent running on the targeted machines gathers the system/application logs. Additionally, the system logs can be sent to the system server for analysis. There are processes by which the messages can be sent to the server. These include syslog or third-party API.

Data Logs Acquisition

Additionally, the system provides the capability to collect logs from a wide range of sources. In this case, readers who may not be conversant with how the system analyzes the collected log information from the monitored systems can gain detailed information about the process in the Log Data Collection document. Some of the examples of log sources supported by Wazuh include:

Operating system logs: The Wazuh system supporting multiple operating systems includes Linux, Windows, and macOS.

Wazuh can collect syslog, auditd, application logs, and others from Linux endpoints.

Wazuh collects logs on Windows endpoints using the Windows event channel and Windows event log format. By default, the Wazuh agent monitors the System, Application, Security Windows event channels on Windows endpoints and from the Microsoft-Windows-Sysmon/Operational event channel on a Windows endpoint.

The Wazuh solution is based on the Wazuh agent, which is deployed on the monitored endpoints, and on three central components: the Wazuh server, the Wazuh indexer, and the Wazuh dashboard.

- The Wazuh dashboard is the web user interface for data visualization and analysis. It includes out-of-the-box dashboards for threat hunting, regulatory compliance (e.g., PCI DSS, GDPR, CIS, HIPAA, NIST 800-53), detected vulnerable applications, file integrity monitoring data, configuration assessment results, cloud infrastructure monitoring events, and others. It is also used to manage Wazuh configuration and to monitor its status.

- Wazuh agents are installed on endpoints such as laptops, desktops, servers, cloud instances, or virtual machines. They provide threat prevention, detection, and response capabilities. They run operating systems such as Linux, Windows, macOS, Solaris, AIX, and HP-UX.

4.7.1 Deployment of Wazuh Detection Platform

For the realization of the detection capabilities identified in the methodology, the Wazuh platform was installed in the laboratory environment on a dedicated virtual machine running under the Proxmox hypervisor **(Node Name: 'ludus').** The installation was carried out in **All-in-One architecture** configuration to ensure the simplicity of the test environment for the installation of the Wazuh Indexer, the server, and the dashboard.

The deployment process consisted of the following stages:

1. **Environment Preparation:** A virtual machine was provisioned within the Ludus environment (VM ID 100). The installation was carried out using the automated **Wazuh Installation Assistant** script (wazuh-install.sh), which simplifies the deployment by managing dependencies and system configurations automatically.

2. **Execution & Configuration:** As shown in the command-line interface below , the installation script was executed with the -a (assistant) flag. This process automatically execute several critical tasks:

    - **Repository Management:** Including the official Wazuh repositories in the system.

    - **Certificate Generation:** Creation of the Root CA , Admin, Indexer, and Filebeat certificates to ensure encrypted communication between both sides of the components.

    - **Installation of Components:** Step-by-step installing and startup the **Wazuh Indexer** (for data storage), the **Wazuh Server** (for agent management and analysis), and the **Wazuh Dashboard** (for visualization).

3. **Verification and Access:** After the successful execution of the script, the installer presented unique administrative login credentials (username admin and a strong password) to the test participant. The functionality of the platform was further verified by gaining access to the web graphical interface utilizing a browser to verify the functionality of the central management console that was ready to collect telemetry information.

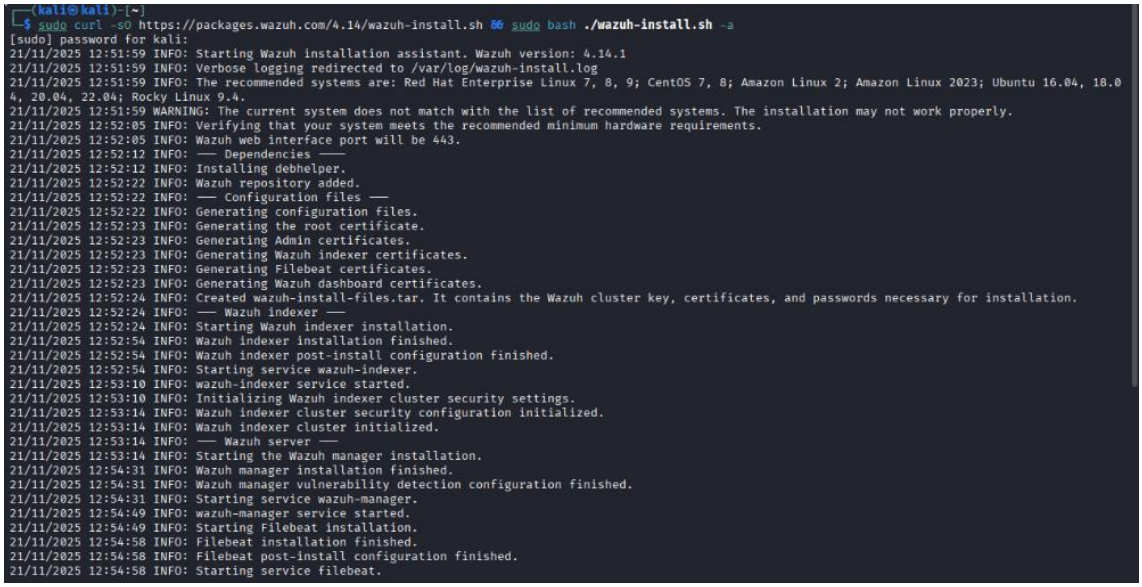


Figure 40: Execution of the Wazuh automated installation script on the Proxmox virtual machine.

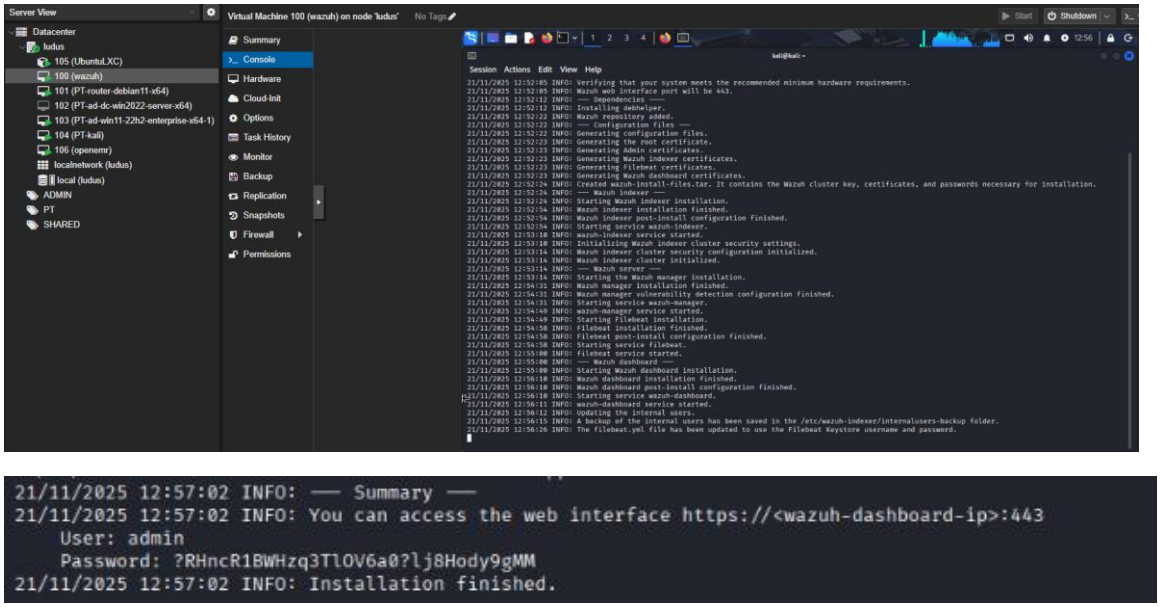


Figure 41: Completion of the installation process showing the generated administrative credentials.

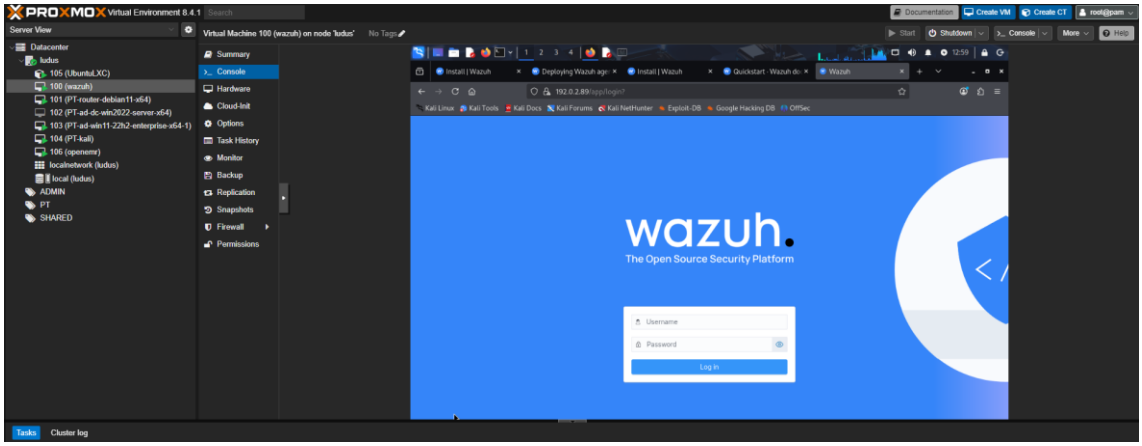


Figure 42: Successful initial login to the Wazuh Dashboard web interface.

4.7.2 Endpoint Agent Deployment (Windows Domain Controller)

After the deployment of the server and initialization of the Wazuh server, the next step was the implementation of the Active Directory environment for collecting the telemetry information. The main aim of the deployment was the Domain Controller named PT-DC01-2022 running Microsoft Windows Server 2022.

Agent deployment procedure

Agent Configuration: The “Deploy new agent” functionality of the Wazuh Dashboard was used to create a customized PowerShell installation script. The installation script was already setup to contain the IP of the Wazuh Manager (192.0.2.83) as well as the necessary authentication credentials to allow for instant connectivity.

Installation via PowerShell: Accessing the Domain Controller via the Proxmox console, an administrative PowerShell session was initiated. The generated command was executed to download the Wazuh Agent MSI package (via Invoke-WebRequest) and install it silently using msiexec.

Service Initialization: Post-installation, the agent service was manually started using the command NET START Wazuh. The console output confirmed the successful startup of the service, establishing the encrypted channel between the Domain Controller and the Wazuh Manager for real-time log forwarding.

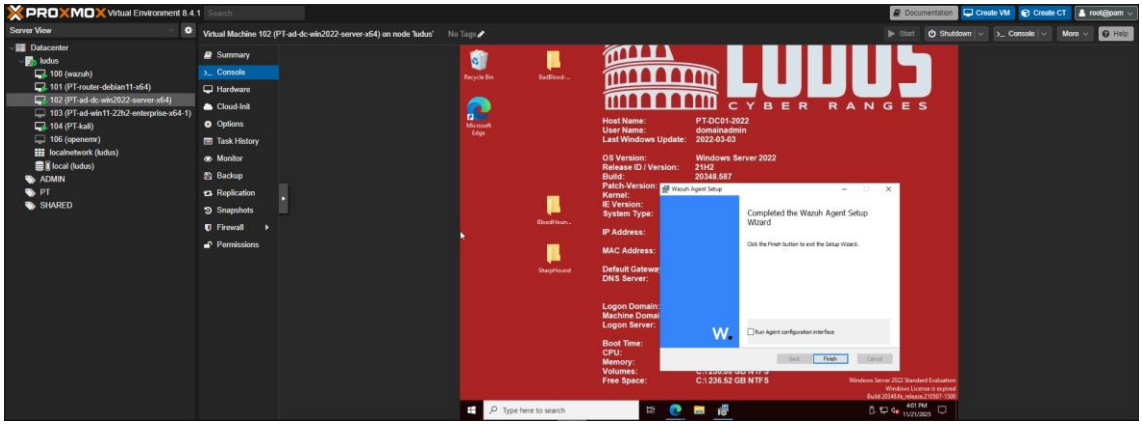


Figure 43: Generation of the Windows Agent deployment command within the Wazuh Dashboard.

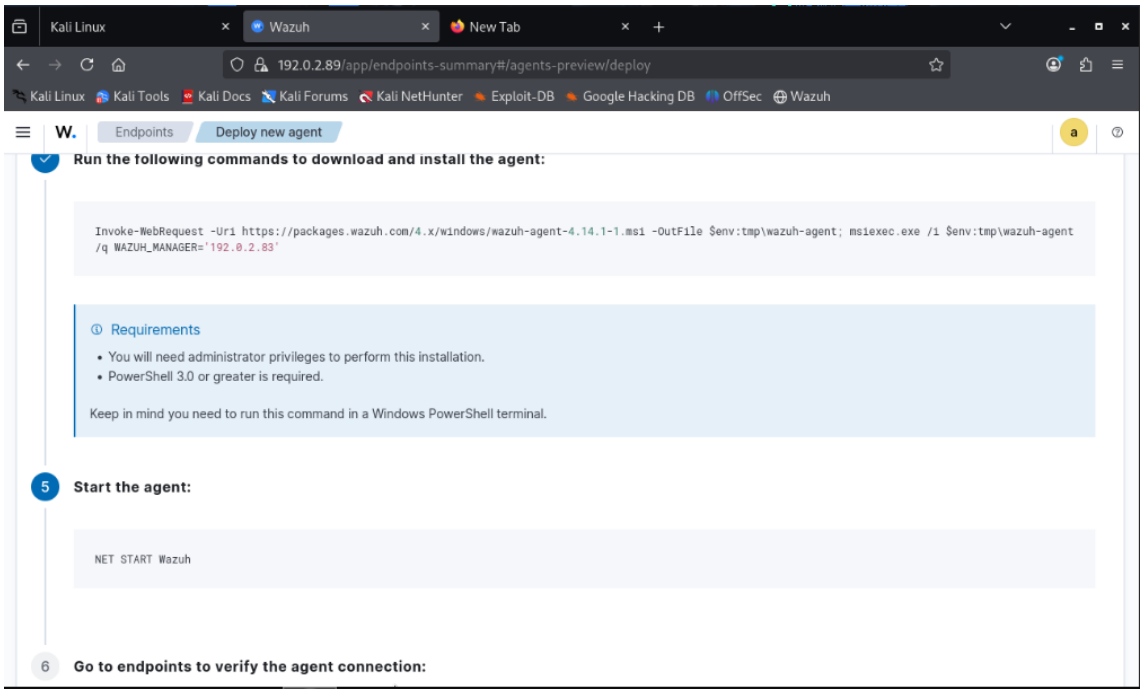


Figure 44: Execution of the installation script via PowerShell on the Windows Server 2022 Domain Controller.

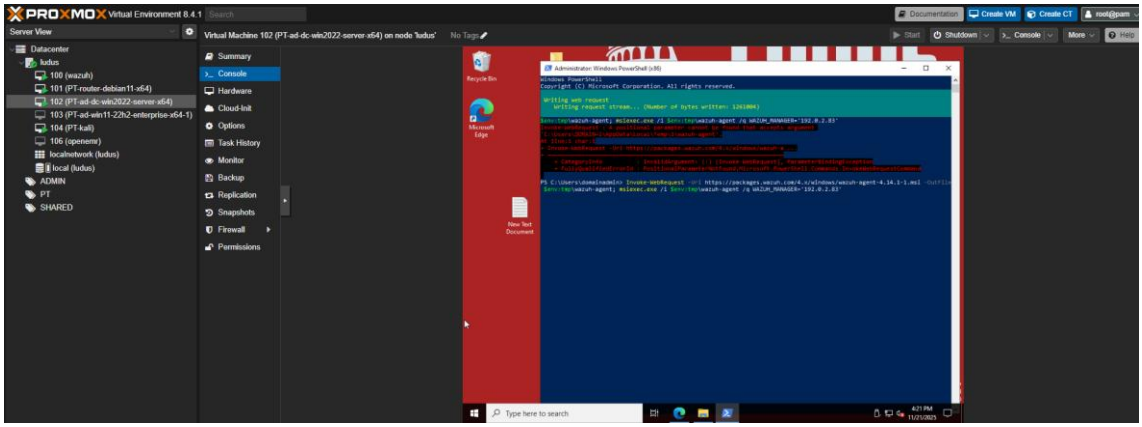


Figure 45: Execution of the installation script via PowerShell on the Windows Server 2022 Domain Controller.

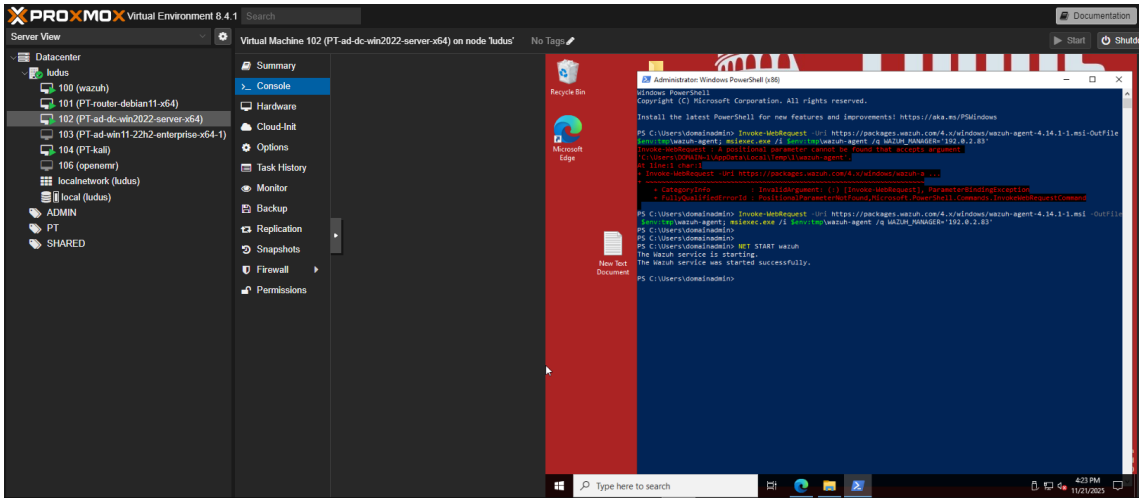


Figure 46: Successful initialization of the Wazuh Agent service verifying endpoint connectivity.

4.7.3 Agent Verification and Connectivity Validation

After the installation of the Wazuh Agent on the Domain Controller, a verification process was performed to validate the functionality of the service to communicate effectively with the central manager.

1. **Process Execution:** The successful initialization of the service was validated by the Task Manager. Notice in figure below that the main process of the service was running under the context of the SYSTEM account (4936). This was necessary to allow the service to have the requisite permissions to query the Window Security Event Log for events.

2. **Connectivity & Log Analysis:** To validate the functionality of the secured communications link, the internal log file of the Ossec Agent (ossec.log) was reviewed. The records validated the successful import of authentication keys and the launch of a TCP connection to the Wazuh Manager running on IP Address 192.0.2.89 & Port Number 1514. Furthermore, the graphical tool for the Agent Manager of the Wazuh framework also validated the "Running" status of the Domain Controller to stream the telemetry information to the detection system.

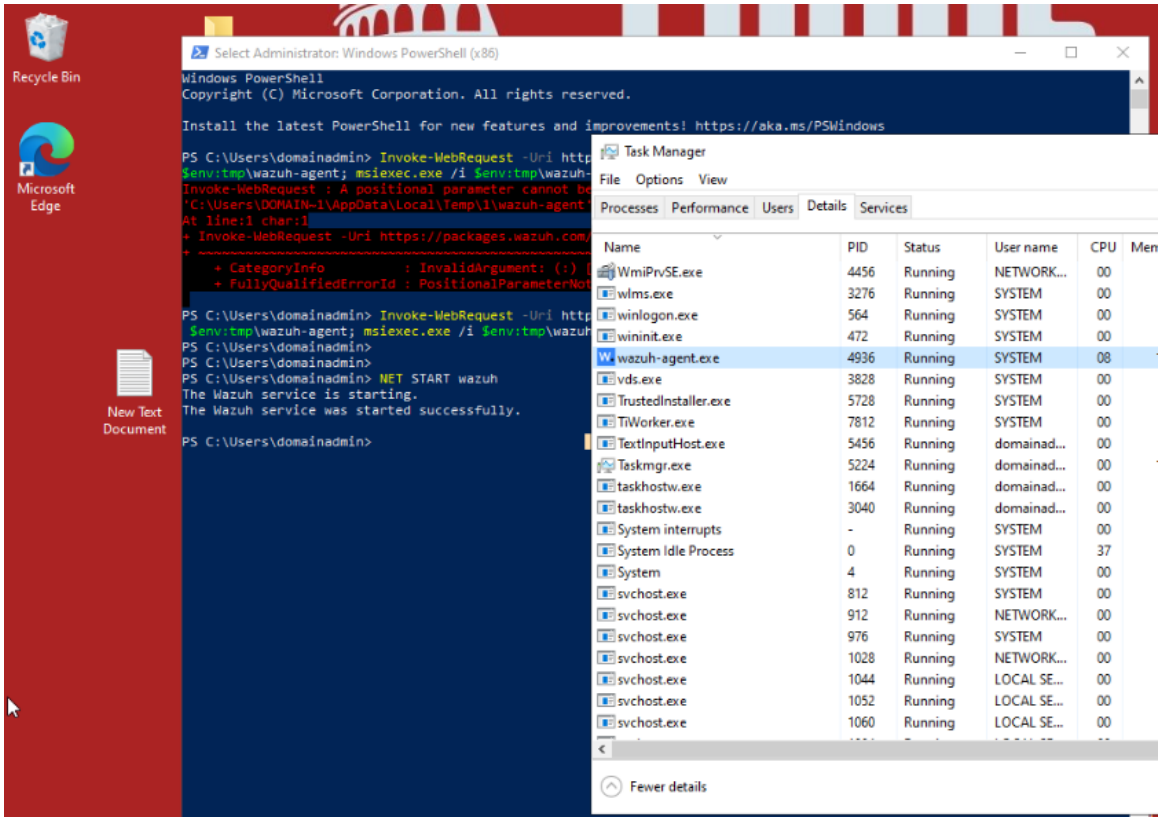


Figure 47: Validation of the running wazuh-agent.exe process with SYSTEM privileges via Windows Task Manager.

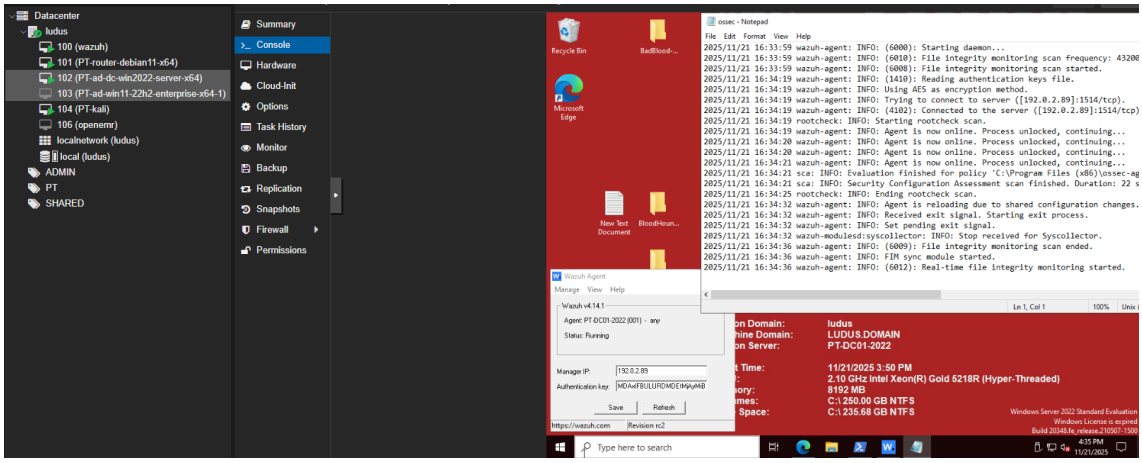


Figure 48: Analysis of the ossec.log file confirming successful connection to the Wazuh Manager and the Agent GUI status indicator.

4.8 Wazuh Dashboard

Following the agent deployment, the final validation step involved confirming the endpoint's registration within the Wazuh Central Component. Accessing the web-based dashboard verified that the Domain Controller was successfully communicating with the manager.

Agent Registration: As illustrated in Figure below, the "Endpoints" summary explicitly lists the target machine (PT-DC01-2022) with the assigned IP address 192.0.2.83. The status indicator is marked as "Active," confirming that the encrypted channel is stable and the agent is sending "keep-alive" beacons to the manager.

Telemetry Ingestion: The "Overview" dashboard provides further confirmation that log ingestion has commenced. The "Last 24 Hours Alerts" panel displays a volume of initial events (classified primarily as Medium and Low severity). This indicates that the agent is not only connected but is actively harvesting Windows Event Logs, parsing them against the default ruleset, and populating the SIEM database in real-time.

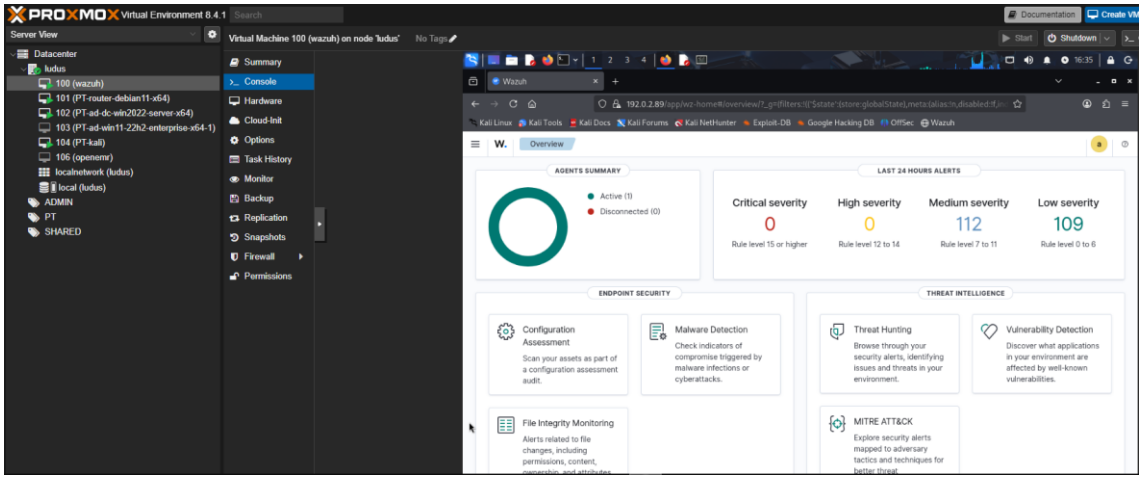


Figure 49: Wazuh "Endpoints" view confirming the active status of the Windows Domain Controller agent.

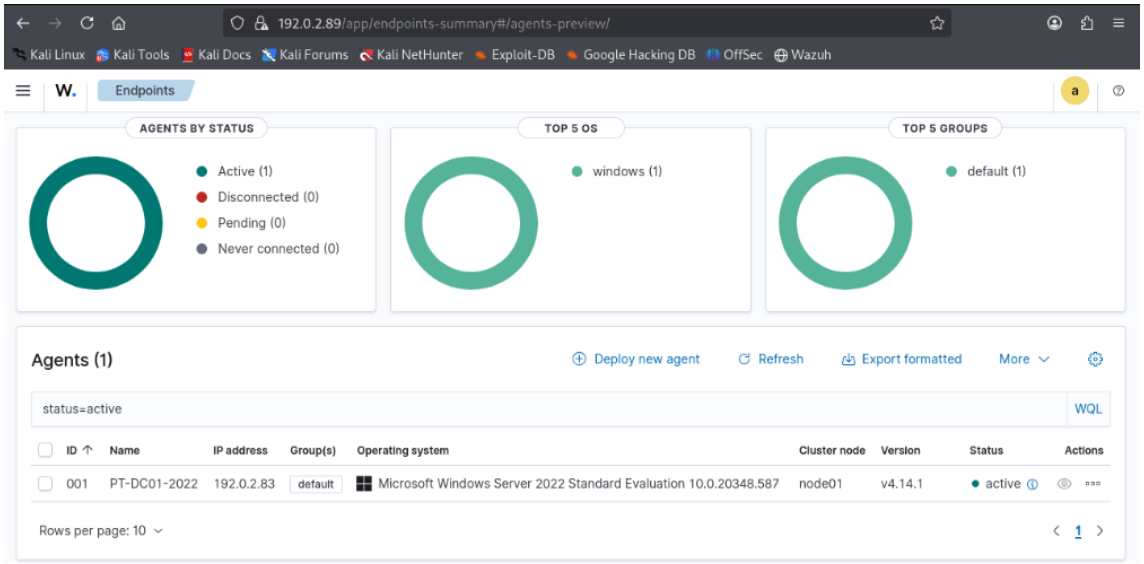


Figure 50: Wazuh "Overview" dashboard showing the successful ingestion of initial telemetry and security alerts from the monitored environment.

4.9 Active Directory Enumeration with SharpHound

To identify complex attack paths and relationship flaws within the generated Active Directory environment, we utilized **BloodHound**, utilizing its data collector component, **SharpHound**. This tool serves as the ingestor, performing comprehensive enumeration of the domain to map trust relationships, group memberships, and permission structures (ACLs).

1. **Execution:** As shown below, the SharpHound.exe binary was executed from the Domain Controller. In a typical assessment, this would be run from a compromised host, but for the purpose of this lab simulation, it was executed locally to ensure full visibility of the LUDUS.DOMAIN structure.

2. **Data Processing:** The console output indicates the active collection phases. The logs explicitly show the **ACL Processor (ACLProc)** identifying specific GUIDs for Access Control rights (such as ms-mcs-admpwd which relates to LAPS) and the **LDAP Query** engine interacting with the Domain Controller.

3. **Completion:** The status messages "Producer has finished" and "LDAP channel closed" confirm that the enumeration completed successfully without errors. This process generated a timestamped ZIP archive (visible on the desktop as 20251122...) containing the JSON files (**Users, Groups, Computers, GPOs**) required for the graphical analysis in the BloodHound interface.

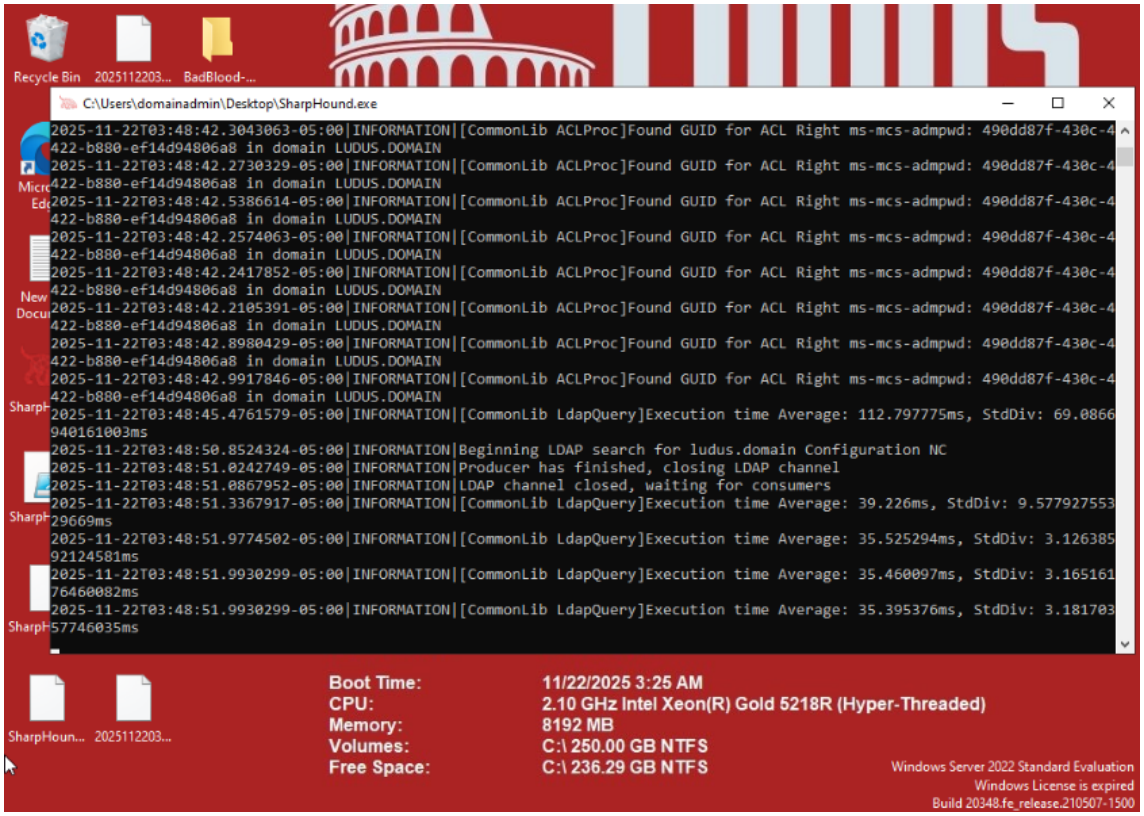


Figure 51: Execution of the SharpHound collector showing real-time LDAP querying and ACL enumeration within the Ludus domain.

4.9.1 Implementation of Advanced Telemetry (Sysmon)

To overcome the limitations of the traditional Windows Event Logging capabilities of being unable to record command line arguments and detailed network connections, the Microsoft Sysinternals tool named Sysmon (version 15.15) was installed on the Domain Controller. Sysmon is a driver installed in the kernel.

To facilitate high-fidelity data gathering while reducing the effects of “log noise” and spurious messages, the deployment was set up utilizing the SwiftOnSecurity Sysmon Configuration. This industry-standard configuration profile was designed to efficiently filter out reputable background tasks while focusing on known activities related to the MITRE ATT&CK framework. These tasks include dumping credentials and laterally moving.

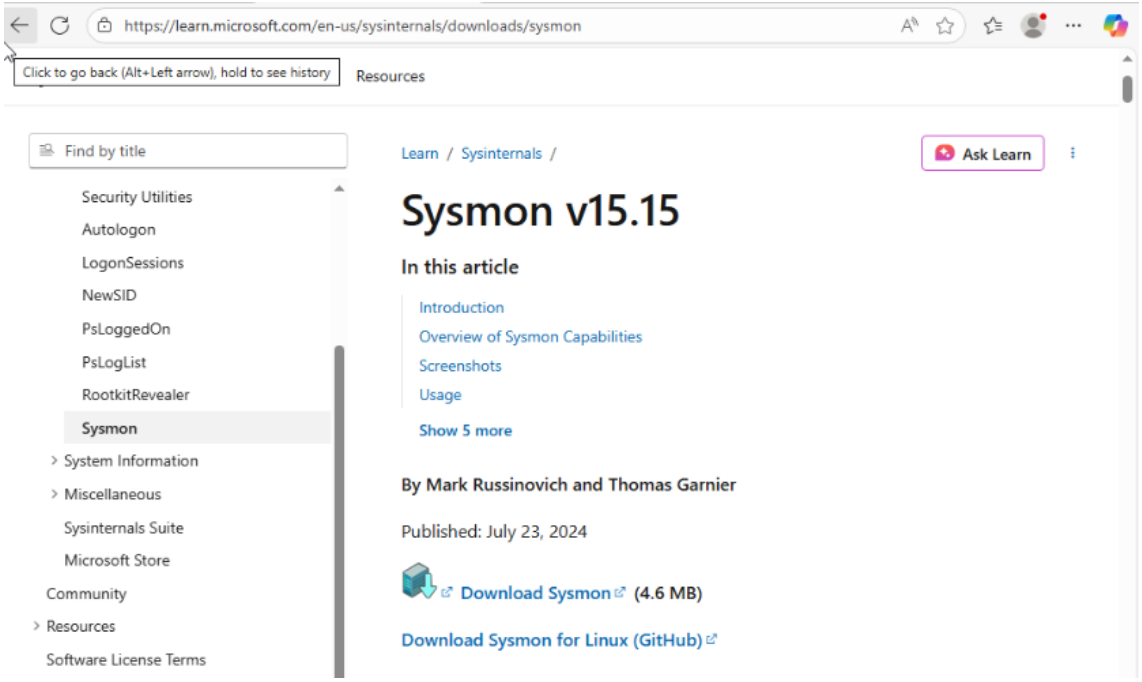


Figure 52: Sourcing the Microsoft Sysmon (v15.15) binary for advanced endpoint monitoring.

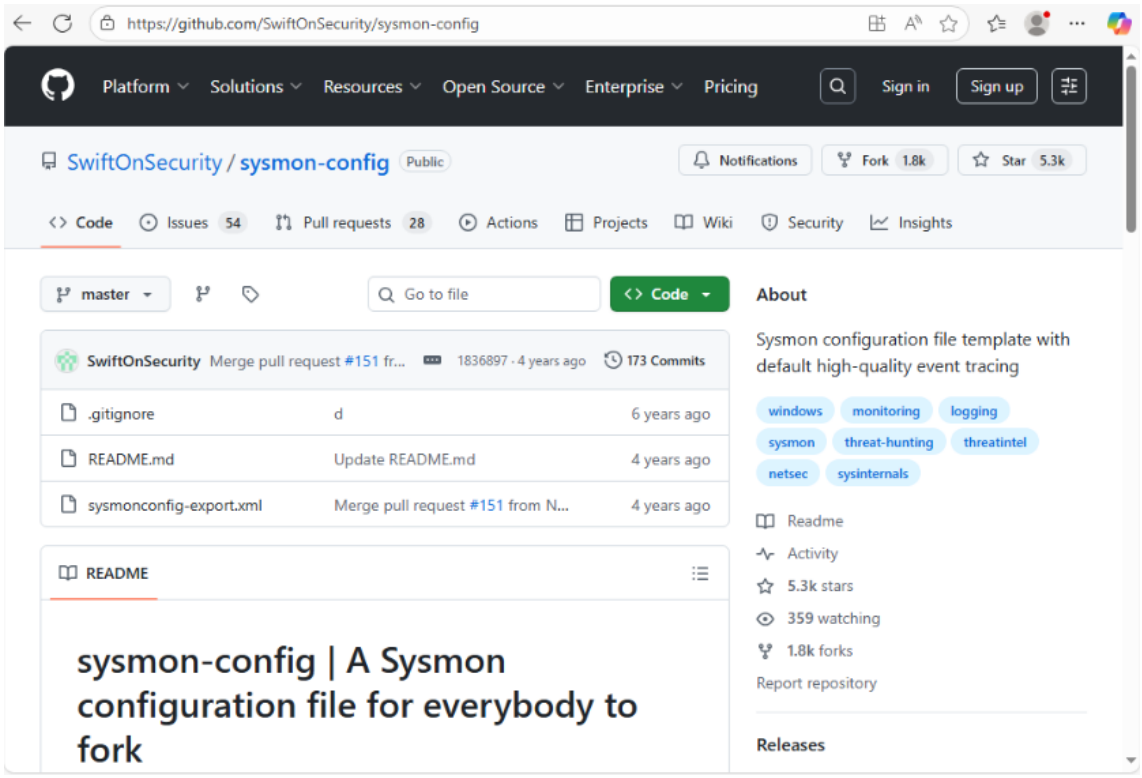


Figure 53: Selection of the SwiftOnSecurity configuration XML to optimize threat detection and reduce telemetry noise.

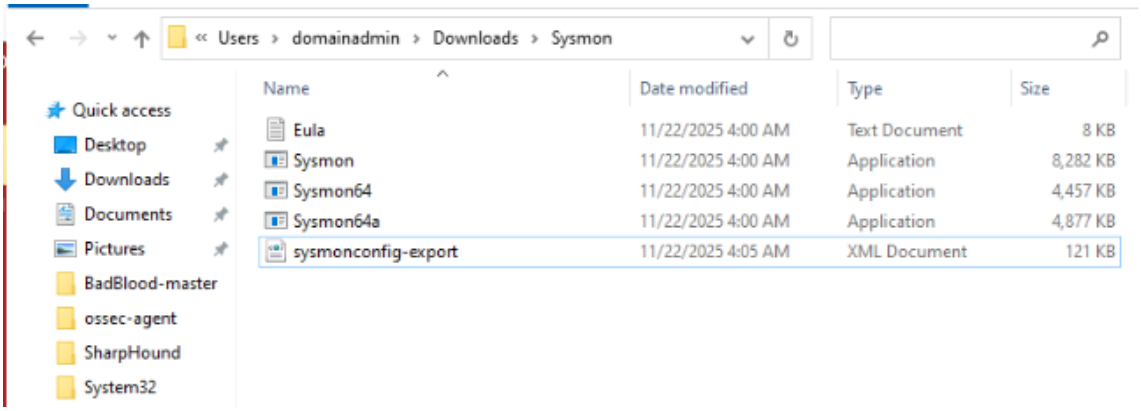


Figure 54: Sysmon Installation Files Downloaded to the Domain Controller.

4.9.2 Deployment and Initialization of Sysmon

With the configuration file prepared, the Sysmon binary was deployed to the Domain Controller. The installation was executed via an administrative command prompt using the syntax **.\Sysmon64.exe -i sysmonconfig-export.xml**. This command instructs the installer to register the service and immediately apply the granular filtering rules defined in the **SwiftOnSecurity** XML file.

As shown below, the installer checked the schema for the configuration (version 4.90) against the binary. After agreeing to the End User License Agreement (EULA) terms, the process was successful in registering the system driver (SysmonDrv) and the background service (Sysmon64) to initialize the immediate start of advanced telemetry gathering.

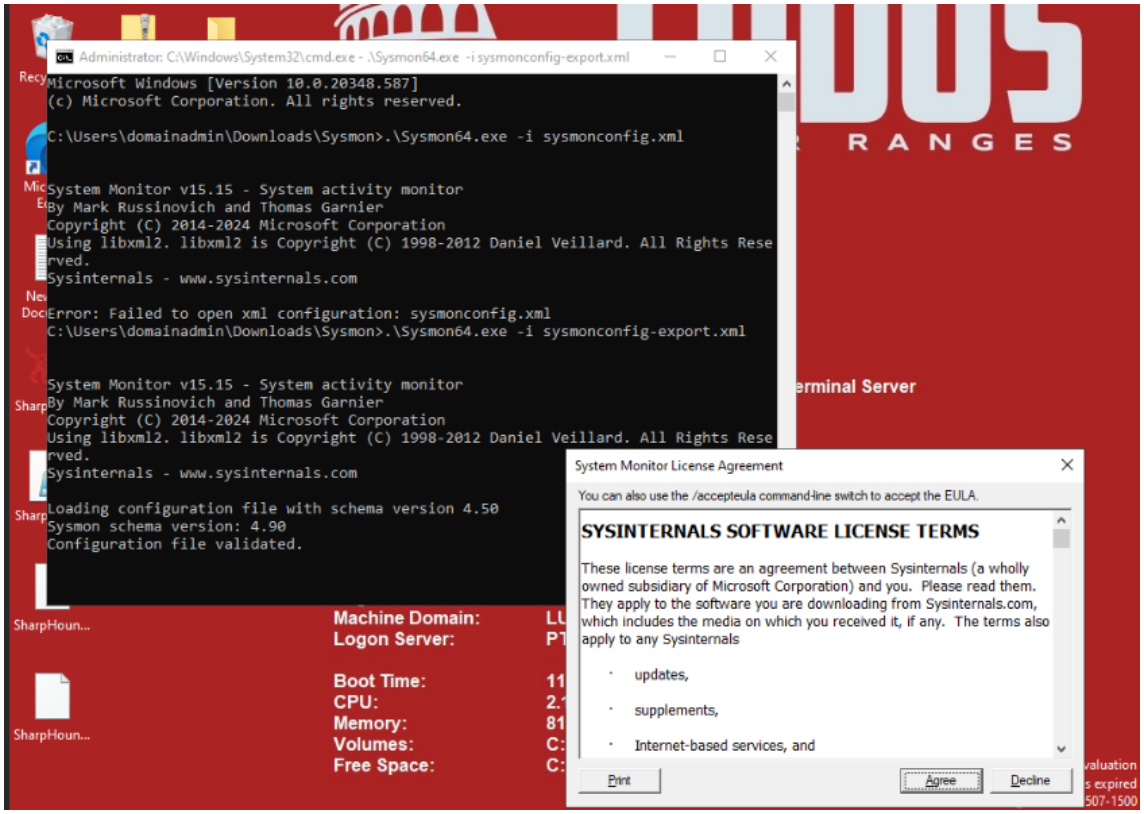


Figure 55: Execution of the Sysmon installation command referencing the external configuration file.

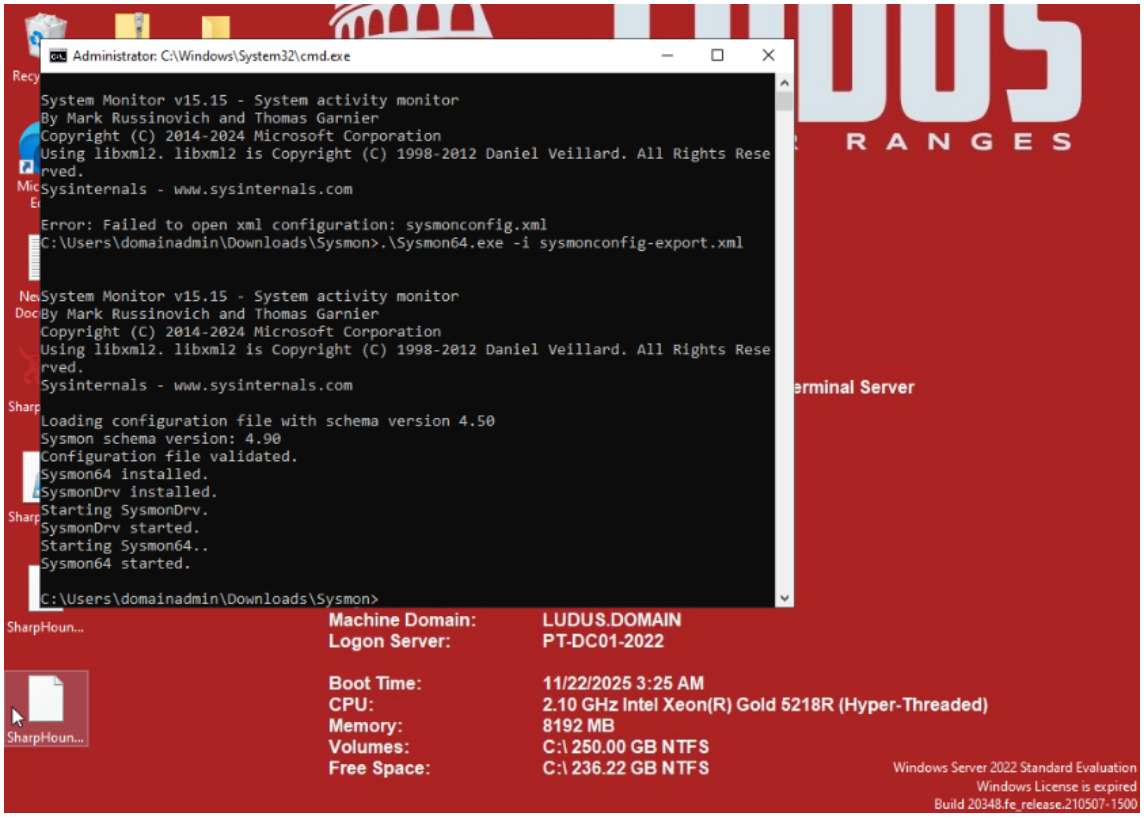


Figure 56: Successful validation of the XML schema and initialization of the Sysmon system driver.

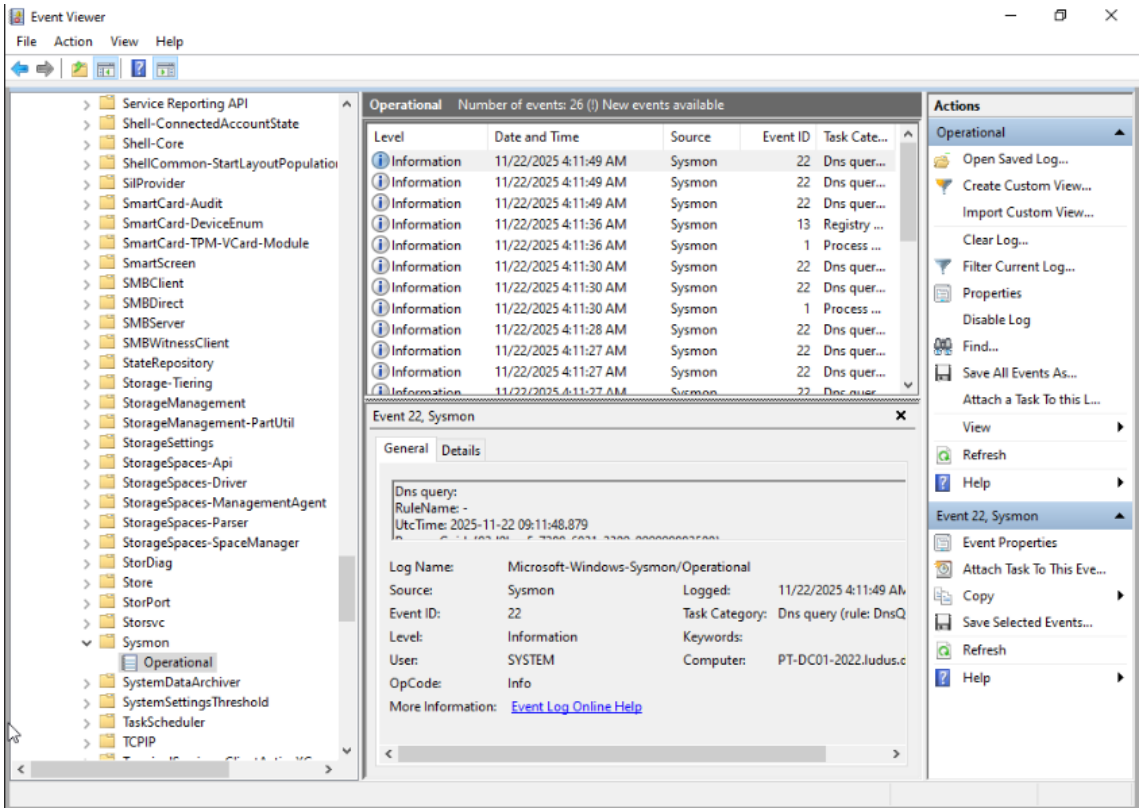


Figure 57: Verification of local Sysmon event generation (ID 1 and ID 22) within the Windows Event Viewer.

4.9.3 Integration of Sysmon Telemetry with Wazuh Agent

Although Sysmon was successfully installed, the Wazuh agent does not ingest Sysmon logs by default; it requires explicit configuration to monitor the specific Event Channel. The integration process involved three distinct steps:

1. **Validation of Local Telemetry:** To verify the presence of locally recorded logs in the system, the Event Viewer of the Windows operating system was checked. As can be viewed in figure below, the log channel for Microsoft-Windows-Sysmon/Operational was filled up with high-value events of Event ID 1 (Process Creation) & Event ID 22 (DNS Query).

2. **Configuration Modification:** To transmit these logs to the Wazuh Manager, the configuration file of the agent (ossec.conf) in the path C:\Program Files (x86)\ossec-agent\ was modified.

3. **Channel Definition:** A new <localfile> was added to the configuration. This adds the Microsoft-Windows-Sysmon/Operational channel to the configuration in the required eventchannel format. After the configuration change was made, the service for the Wazuh agent was restarted to implement the change and allow the data to feed from the clients to the central server.

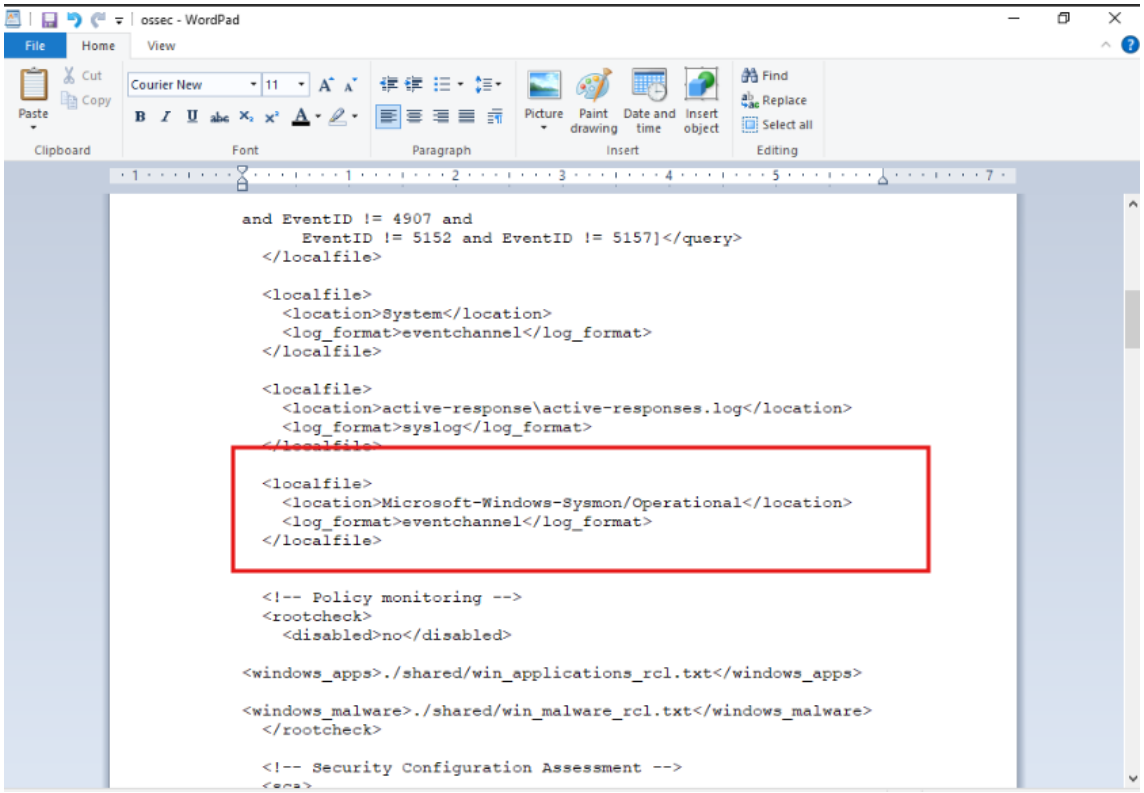


Figure 58: Configuration of the <localfile> XML block enabling the ingestion of the Sysmon.

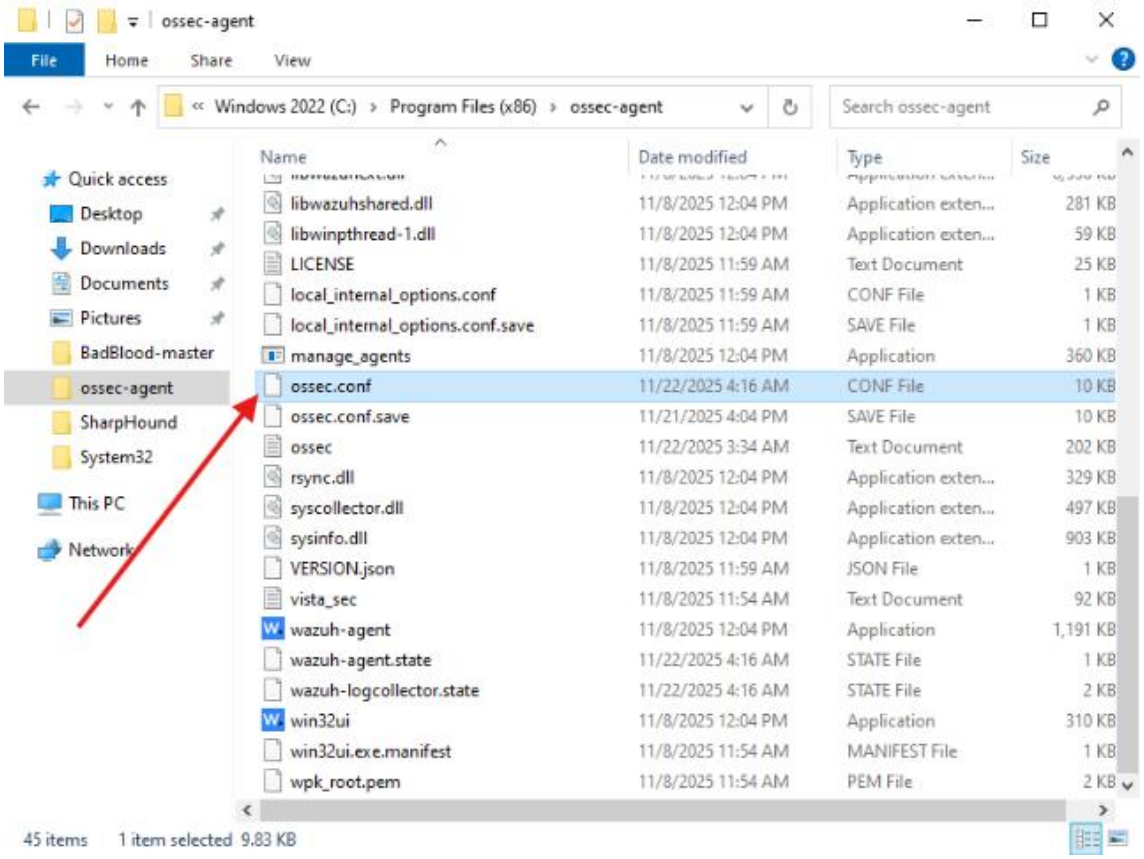


Figure 59: Location of the ossec.conf configuration file within the agent installation directory.

4.10 Validation of Telemetry Generation (SharpHound)

To validate the relevance of the configuration of the tool Sysmon, the reconnaissance tool SharpHound was launched on the Domain Controller. Its main intention was to validate the capability of the endpoint sensor to monitor the targeted behavioral indicators for Active Directory enumeration.

Event Viewer analysis on the local system verified the production of high-fidelity telemetry:

1. **Network Visibility (Event ID 3):** Sysmon has recorded the network connection made by the SharpHound.exe process. This establishes a connection to the Domain Controller over port 389, known as LDAP. This follows the standards for querying a directory service.

2. **Process Lifecycle (Event ID 5)**: The Process Terminated event was recorded by the system. This recorded information enabled the recreation of the time correlated to the recon window.

3. **Forensic Artifacts (Event ID 13):** Notably, the change made to the Registry (AppCompatFlags) by the SharpHound binary was identified by the Sysmon tool. The above-mentioned forensic artifact evidence makes the execution of the tool difficult to remove even if the tool was deleted immediately after execution.

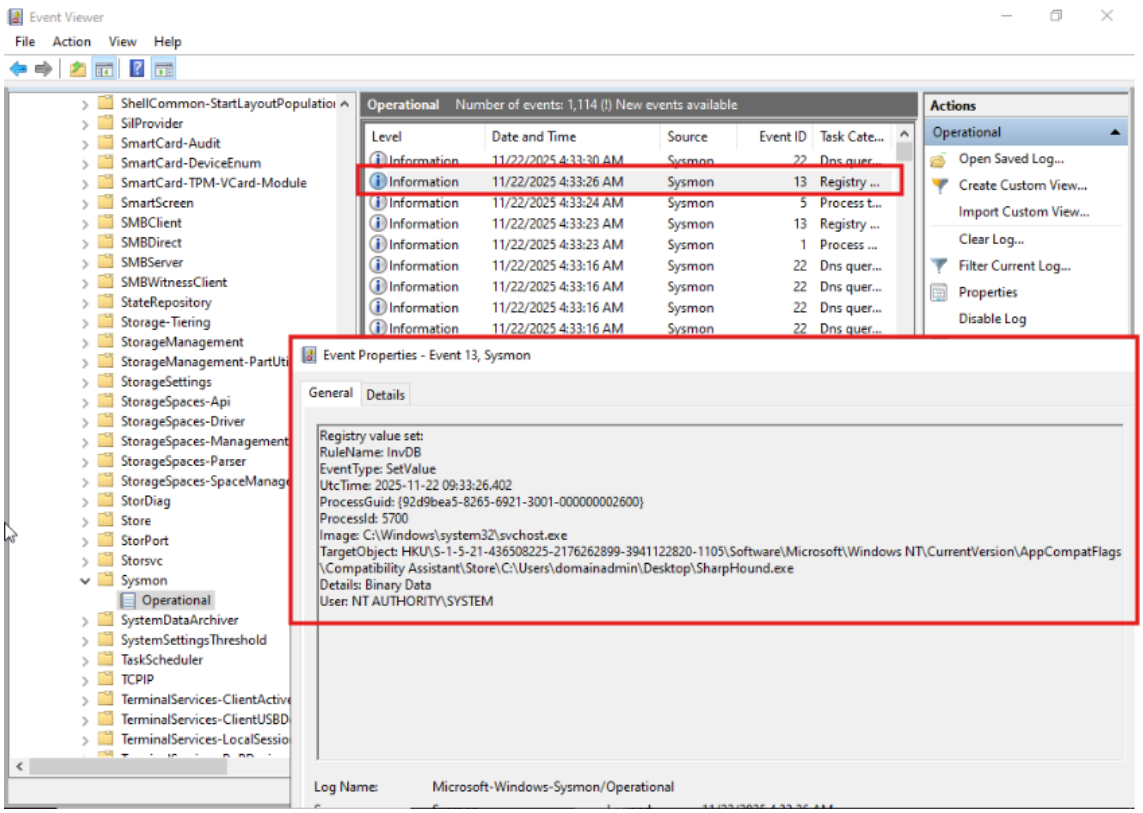


Figure 60: Sysmon Event ID 13 showing process termination and the creation of forensic registry artifacts.

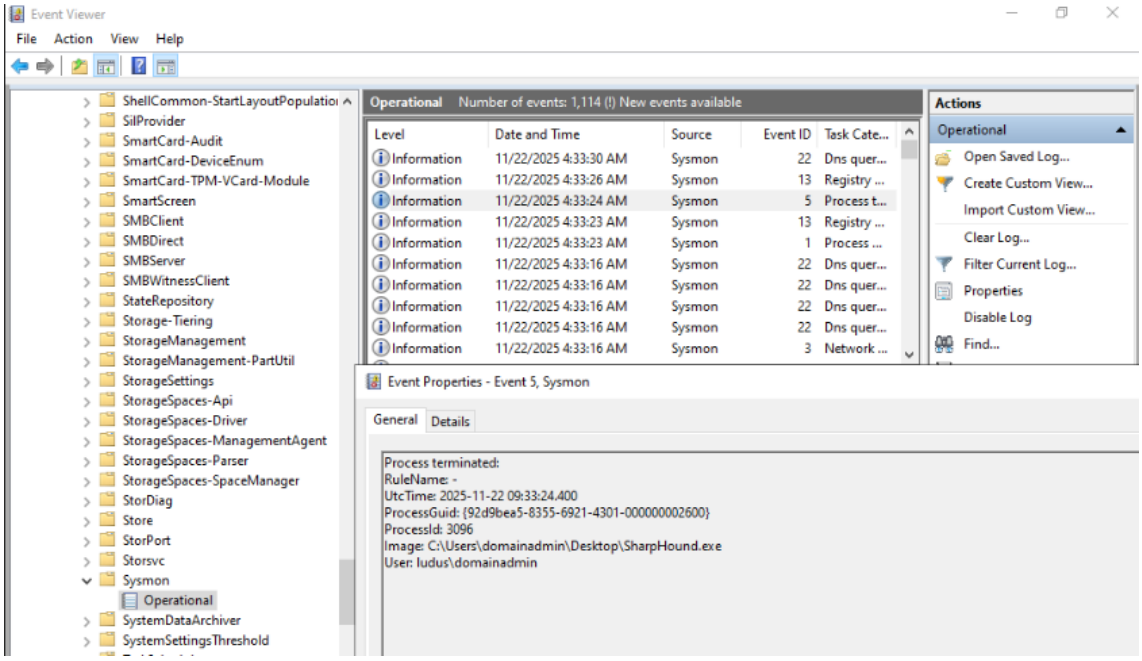


Figure 61: Sysmon Event ID 5 and Event ID 13 showing process termination and the creation of forensic registry artifacts.

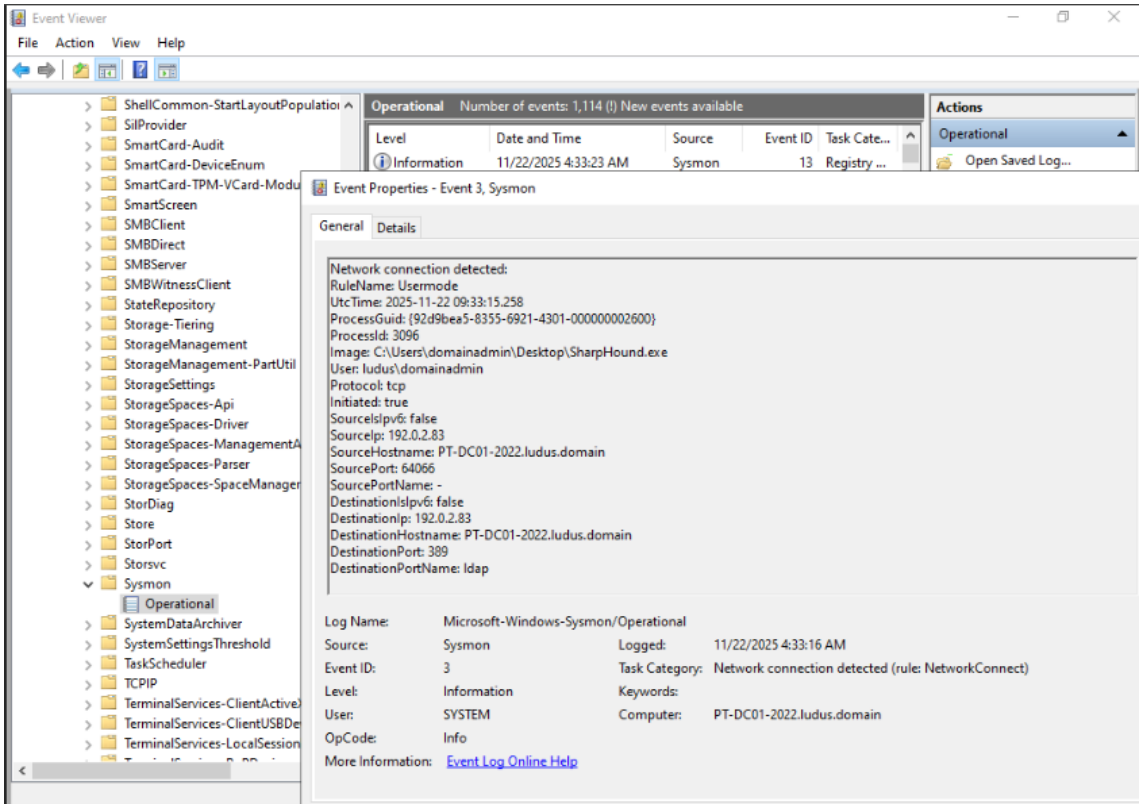


Figure 62: Sysmon Event ID 3 capturing the LDAP network connection (Port 389) initiated by the SharpHound binary.

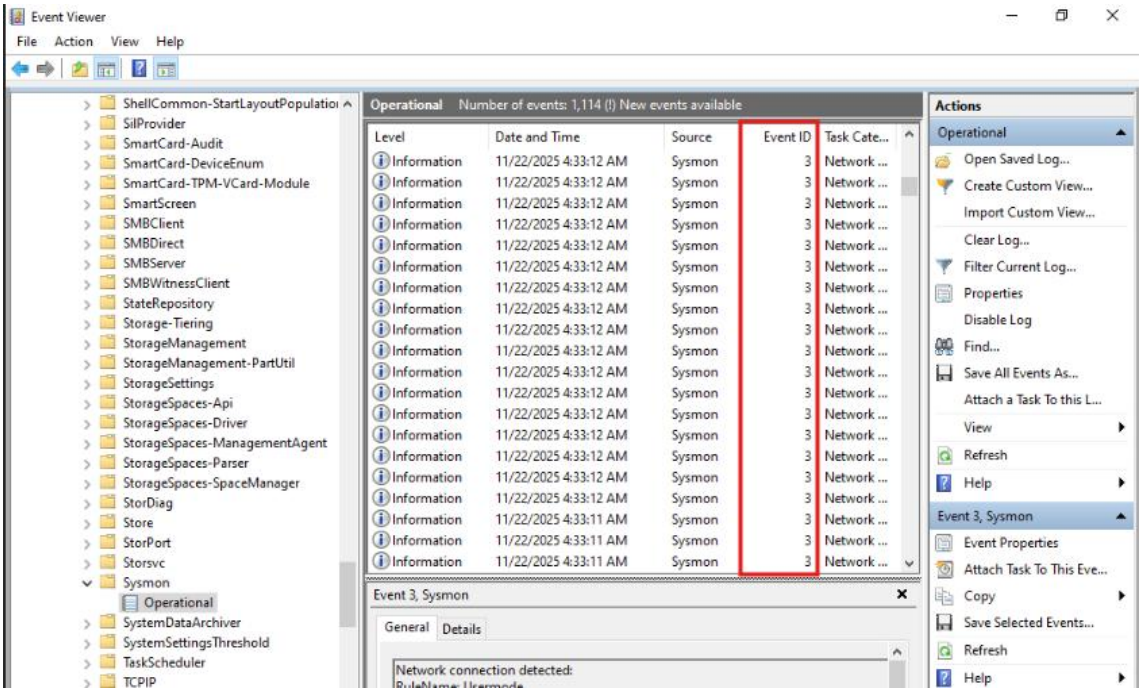


Figure 63: Sysmon Event ID 3 Entries Showing Network Connection Detections in Event Viewer

Why a Tool Like Sysmon Is Required

Wazuh focuses on the analysis of common events in the Windows operating system. There is a tool named Sysmon developed by Microsoft. It works at the kernel level. The tool logs high-fidelity events. Some of them include:

- Process Creation (Event ID 1) : This records the entire command line for running the SharpHound.exe or SharpHound.ps1 file.
- Network Connections (Event ID 3): Tracks the connections being made to the LDAP (port 389) and SMB (port 445) services when SharpHound searches the domain, producing a very telling sign of enumeration.

  Absent the functionality of Sysmon, these vital pieces of information would not be recorded or would lack the depth of detail necessary for the detection rules of the W

4.11 Verification of Centralized Telemetry Aggregation

Following the agent configuration, we validated the integrity of the data pipeline by accessing the **Wazuh Discovery** interface. This phase confirmed that the central server was successfully ingesting and normalizing logs from the PT-DC01-2022 endpoint.

1. **Log Indexing and Volumetrics:** As demonstrated below, the "Discover" tab indicated a steady influx of security events (327 hits in a 30-minute window). The presence of these logs confirms that the encrypted channel between the Agent and the Manager is stable and capable of handling the event throughput generated by the lab environment.
2. **Field Parsing and Normalization: T**he "Expanded" view of the ingested logs. Crucially, this shows that the Wazuh decoder is correctly parsing raw Windows Event Logs into structured JSON fields (e.g., data.win.eventdata.targetUserName, authenticationPackageName: Kerberos). This normalization is a prerequisite for the

correlation engine to function, as detection rules rely on these specific field values rather than raw text blocks.

3. **Severity Classification:** The **Overview** dashboard confirmed that the alerting engine is active. The interface highlights **6 Critical Severity alerts** (Rule Level 15+), indicating that the system is not merely archiving logs but is actively evaluating them against its threat intelligence ruleset in real-time.

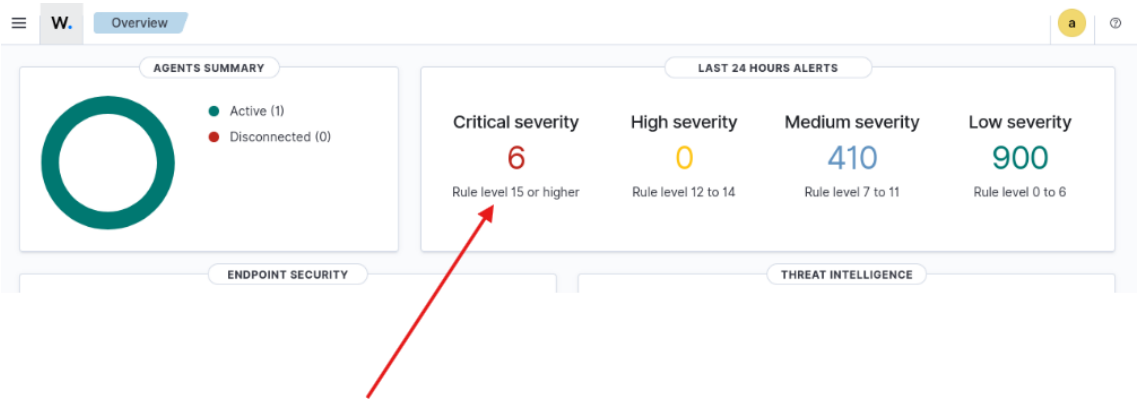


Figure 64: The Security Events Overview dashboard highlighting the detection of "Critical" severity incidents.

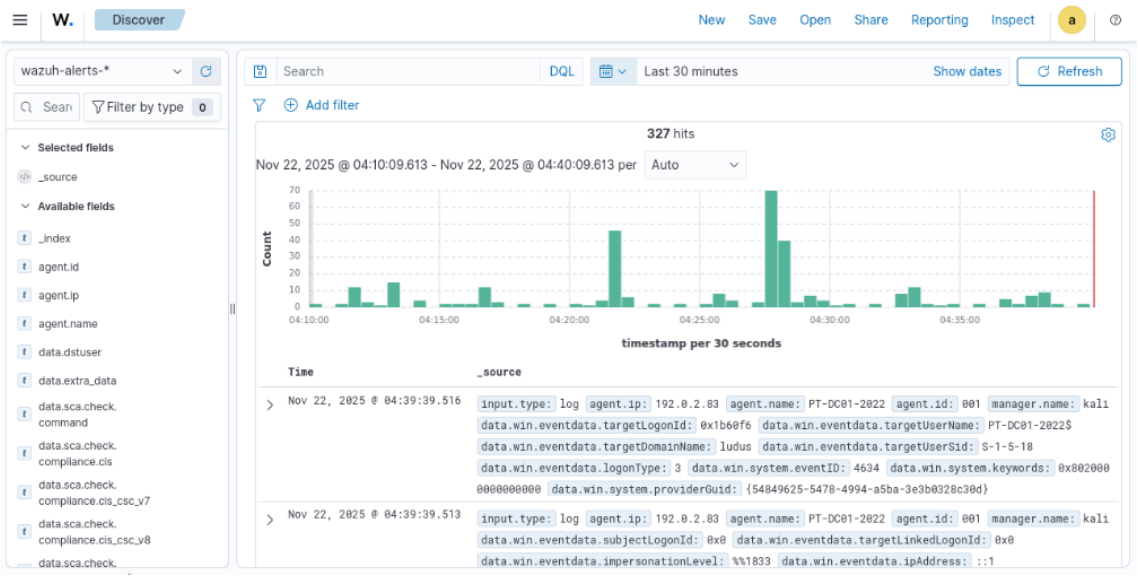


Figure 65: Wazuh Discovery tab showing the real-time ingestion of security events from the Domain Controller.

Figure 66: detailed JSON view of a parsed Windows Security log, confirming correct field extraction for Kerberos authentication events.

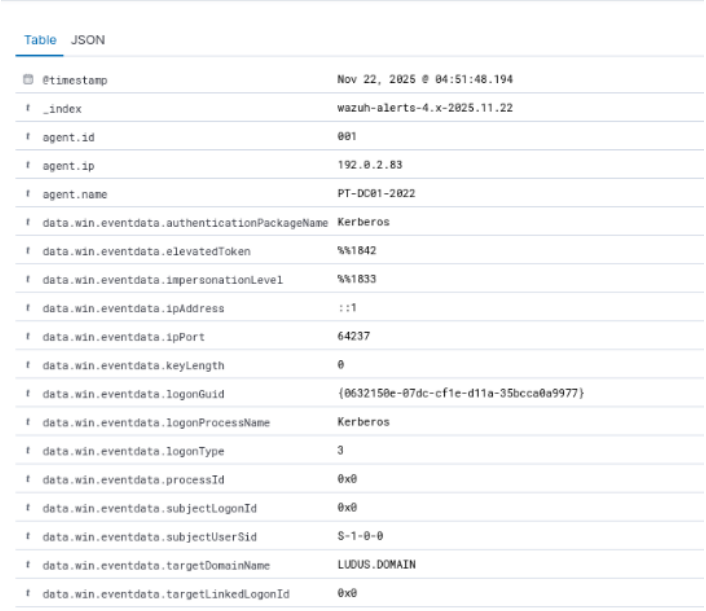
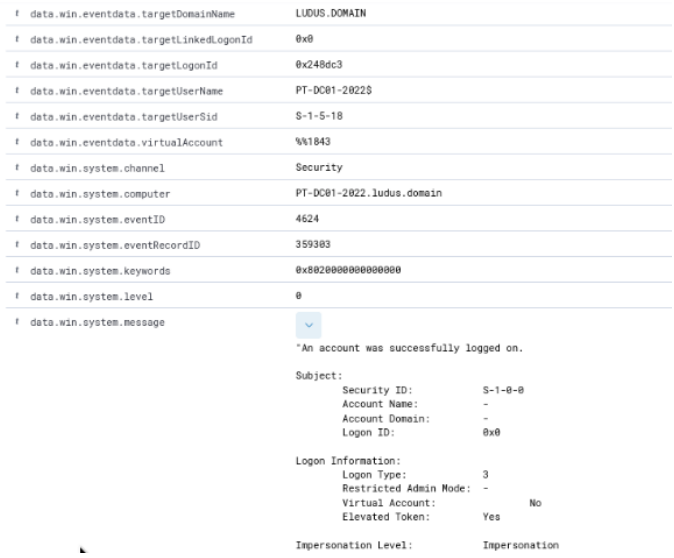


Figure 67: Detailed view of the "Logon Information" validating the Kerberos authentication package and Network logon type.



t data.win.eventdata.targetDomainName	LUDUS.DOMAIN
t data.win.eventdata.targetLinkedLogonId	0x0
t data.win.eventdata.targetLogonId	0x248dc3
t data.win.eventdata.targetUserName	PT-DC01-2022$
t data.win.eventdata.targetUserSid	S-1-5-18
t data.win.eventdata.virtualAccount	%%1843
t data.win.system.channel	Security
t data.win.system.computer	PT-DC01-2022.ludus.domain
t data.win.system.eventID	4624
t data.win.system.eventRecordID	359303
t data.win.system.keywords	0x8020000000000000
t data.win.system.level	0
t data.win.system.message	"An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Account Domain: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes Impersonation Level: Impersonation

Figure 68: JSON representation of a Windows 4624 Event, showing the successful extraction of network port and user identity fields.

t data.win.system.providerName	Microsoft-Windows-Security-Auditing
t data.win.system.severityValue	AUDIT_SUCCESS
t data.win.system.systemTime	2025-11-22T09:51:38.0006473Z
t data.win.system.task	12544
t data.win.system.threadID	6852
t data.win.system.version	2
t decoder.name	windows_eventchannel
t id	1763805108.3157918
t input.type	log
t location	EventChannel
t manager.name	kali
t rule.description	Windows Logon Success
# rule.firedtimes	159
t rule.gdpr	IV_32.2
t rule.gpg13	7.1, 7.2
t rule.groups	windows, windows_security, authentication_success
t rule.hipaa	164.312.b
t rule.id	60106
# rule.level	3
◕ rule.mail	false
t rule.mitre.id	T1078
t rule.mitre.tactic	Defense Evasion, Persistence, Privilege Escalation, Initial Access

Figure 69: Wazuh alert metadata demonstrating the automated mapping of the event to MITRE ATT&CK Tactic T1078

Validation of Event Parsing and Contextual Enrichment

To ensure the detection engine could effectively correlate disparate events, we validated the **normalization process**. As illustrated in the collected data, the Wazuh decoder successfully transformed raw Windows Event Logs into structured JSON objects, enabling precise querying and automated analysis.

1. **Field Extraction:** Figures above demonstrate the granular mapping of **Event ID 4624 (Successful Logon)**. Critical attributes were correctly parsed into specific fields, such as:
    - data.win.eventdata.targetUserName: **PT-DC01-2022$** (identifying the machine account).
    - data.win.eventdata.logonType: **3** (Network Logon, indicating a remote connection).
    - data.win.eventdata.ipPort: **64237** (Dynamic ephemeral port used for the connection).
2. **Automated Enrichment:** Beyond simple parsing, the platform automatically enriched the telemetry with metadata. The system appended the rule.mitre.id field with value **T1078** ("Valid Accounts"). This proves that the monitoring solution effectively maps raw activity to the **MITRE ATT&CK framework** in real-time, a capability that significantly aids in the threat hunting phase by categorizing events based on adversary tactics (Defense Evasion, Persistence, Privilege Escalation).

4.12 Offensive Operations and Detection on Linux Infrastructure

Following the verification of the monitoring pipeline, the research proceeds to the active exploitation phase targeting the application layer. In this stage, we focus on the OpenEMR medical records system hosted on the Linux virtual machine.

The objective is to execute specific offensive techniques primarily focusing on Remote Code Execution (RCE) to simulate a critical compromise of the medical data server. By exploiting known vulnerabilities or misconfigurations within the OpenEMR application, we aim to gain unauthorized shell access to the underlying Linux operating system. Simultaneously, we will analyze the telemetry captured by the Wazuh agent to validate its capability to detect the exploitation chain, including suspicious process spawning, unauthorized network connections, and post-exploitation persistence mechanisms.

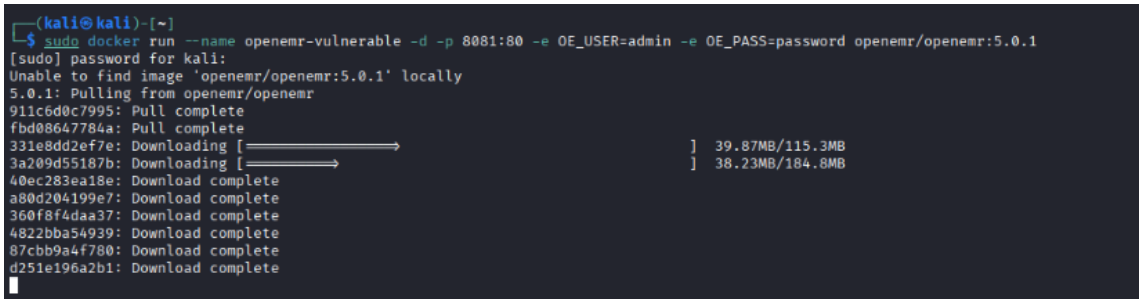


Figure 70: Creation of the dedicated Docker network and deployment of the MySQL database backend.

```
┌──(kali㉿kali)-[~]
└─$ sudo docker network create openemr-net
6580e147f99656baf175d38557553550abf0df64f211468aa41668d11e260e4b

┌──(kali㉿kali)-[~]
└─$ sudo docker run --detach \
  --name mysql \
  --network openemr-net \
  --env "MYSQL_ROOT_PASSWORD=root" \
  --env "MYSQL_USER=openemr" \
  --env "MYSQL_PASSWORD=openemr" \
  --env "MYSQL_DATABASE=openemr" \
  mysql:5.7
b859b05d02514d8812188f20e8daaff1a6713831524c02c4f3449ef41e29bb79
```

Figure 71: Creation of the dedicated Docker network and deployment of the MySQL database backend.

4.12.1 OpenEMR Environment Provisioning

To simulate a realistic application server environment, the **OpenEMR** platform was deployed using containerization technology on the Linux host. First, a dedicated Docker bridge network (openemr-net) was created to ensure isolated communication between the application and its database.

Subsequently, the backend was provisioned using the mysql:5.7 container image, configured with the necessary environment variables for database authentication. Once the database was active, the vulnerable application instance **OpenEMR version 5.0.1**, was deployed and linked to the dedicated network. Port forwarding was configured to expose the web interface on port 8081, making it accessible for the subsequent vulnerability assessment and exploitation.

```
Session  Actions  Edit  View  Help

┌──(kali㉿kali)-[~]
└─$ sudo docker run --detach \
  --name mysql \
  --network openemr-net \
  --env "MYSQL_ROOT_PASSWORD=root" \
  --env "MYSQL_USER=openemr" \
  --env "MYSQL_PASSWORD=openemr" \
  --env "MYSQL_DATABASE=openemr" \
  mysql:5.7
Unable to find image 'mysql:5.7' locally
5.7: Pulling from library/mysql
20e4dcae4c69: Pull complete
1c56c3d4ce74: Pull complete
e9f03a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de6e: Pull complete
ae71319cb779: Pull complete
ffc89e9dfd88: Pull complete
43d05e938198: Pull complete
064b2d298fba: Pull complete
df9a4d85569b: Pull complete
Digest: sha256:4bc6bc963e6d8443453676cae56536f4b8156d78bae03c0145cbe47c2aad73bb
Status: Downloaded newer image for mysql:5.7
9b552c1377b7dc43cb11bae0731ea23d957bb84492c869abc3af3ebce9f7ad0e

┌──(kali㉿kali)-[~]
└─$ sudo docker run --detach \
  --name openemr-vulnerable \
  --network openemr-net \
  --publish 8090:80 \
  --env "MYSQL_HOST=mysql" \
  --env "MYSQL_ROOT_PASS=root" \
  --env "MYSQL_USER=openemr" \
  --env "MYSQL_PASS=openemr" \
  --env "OE_USER=admin" \
  --env "OE_PASS=password" \
  openemr/openemr:5.0.1
c8b120ba41a86d0a8f5f5db18100d02850f0deca4eb9929f5a43f559de7fbd37

┌──(kali㉿kali)-[~]
└─$
```

Figure 72: Pulling and initialization of the vulnerable OpenEMR 5.0.1 container image.

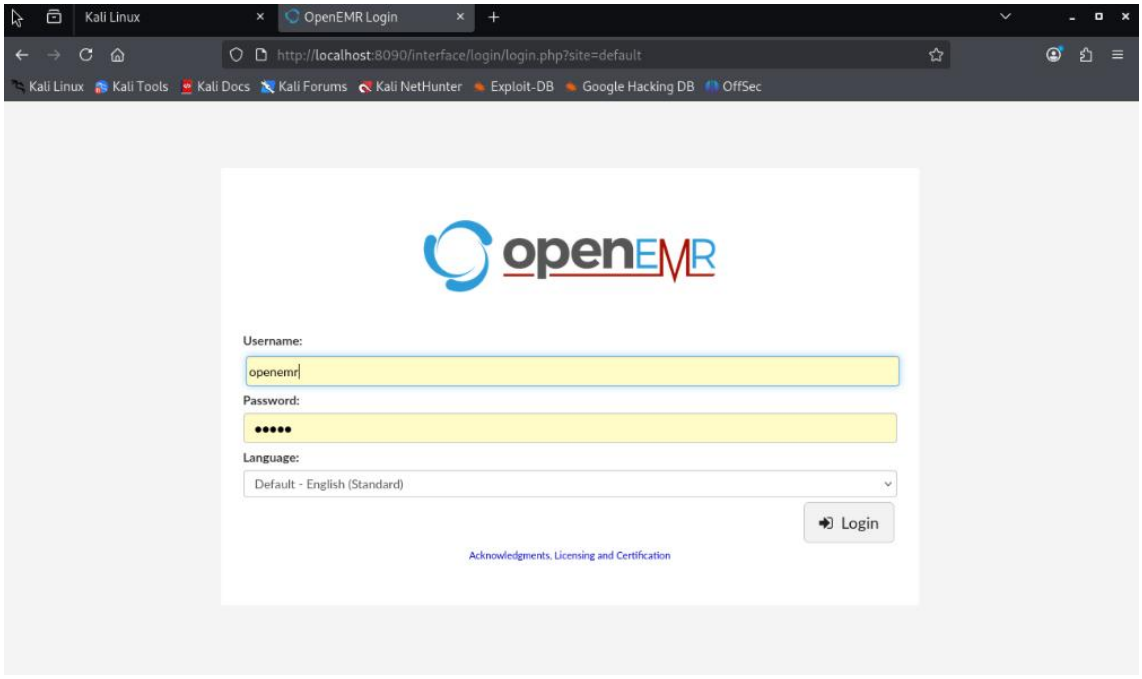


Figure 73: Verification of the OpenEMR login portal.

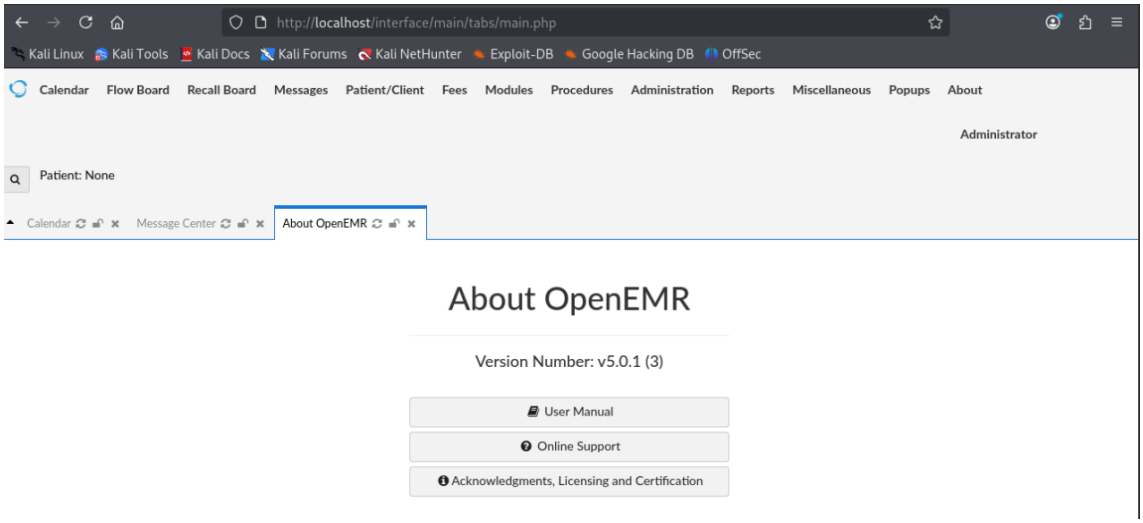


Figure 74: OpenEMR v5.0.1 (3) Web Interface Accessed via Localhost.

```
└─$ searchsploit openemr
 ---------------------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                                           |  Path
 ---------------------------------------------------------------------------------------- ---------------------------------
OpenEMR - 'site' Cross-Site Scripting                                                    | php/webapps/38328.txt
OpenEMR - Arbitrary '.PHP' File Upload (Metasploit)                                      | php/remote/24529.rb
OpenEMR 2.8.1 - 'fileroot' Remote File Inclusion                                         | php/webapps/1886.txt
OpenEMR 2.8.1 - 'srcdir' Multiple Remote File Inclusions                                 | php/webapps/2727.txt
OpenEMR 2.8.2 - 'Import_XML.php' Remote File Inclusion                                   | php/webapps/29556.txt
OpenEMR 2.8.2 - 'Login_Frame.php' Cross-Site Scripting                                   | php/webapps/29557.txt
OpenEMR 3.2.0 - SQL Injection / Cross-Site Scripting                                     | php/webapps/15836.txt
OpenEMR 4 - Multiple Vulnerabilities                                                     | php/webapps/18274.txt
OpenEMR 4.0 - Multiple Cross-Site Scripting Vulnerabilities                              | php/webapps/36034.txt
OpenEMR 4.0.0 - Multiple Vulnerabilities                                                 | php/webapps/17118.txt
OpenEMR 4.1 - '/contrib/acog/print_form.php?formname' Traversal Local File Inclusion     | php/webapps/36650.txt
OpenEMR 4.1 - '/Interface/fax/fax_dispatch.php?File' 'exec()' Call Arbitrary Shell Command | php/webapps/36651.txt
OpenEMR 4.1 - '/Interface/patient_file/encounter/load_form.php?formname' Traversal Local F | php/webapps/36649.txt
OpenEMR 4.1 - '/Interface/patient_file/encounter/trend_form.php?formname' Traversal Local  | php/webapps/36648.txt
OpenEMR 4.1 - 'note' HTML Injection                                                      | php/webapps/38654.txt
OpenEMR 4.1.0 - 'u' SQL Injection                                                        | php/webapps/49742.py
OpenEMR 4.1.1 - 'ofc_upload_image.php' Arbitrary File Upload                             | php/webapps/24492.php
OpenEMR 4.1.1 Patch 14 - Multiple Vulnerabilities                                        | php/webapps/28329.txt
OpenEMR 4.1.1 Patch 14 - SQL Injection / Privilege Escalation / Remote Code Execution (Met | php/remote/28408.rb
OpenEMR 4.1.2(7) - Multiple SQL Injections                                               | php/webapps/35518.txt
OpenEMR 5.0.0 - OS Command Injection / Cross-Site Scripting                              | php/webapps/43232.txt
OpenEMR 5.0.0 - Remote Code Execution (Authenticated)                                    | php/webapps/49983.py
OpenEMR 5.0.1 - 'controller' Remote Code Execution                                       | php/webapps/48623.txt
OpenEMR 5.0.1 - Remote Code Execution (1)                                                | php/webapps/48515.py
OpenEMR 5.0.1 - Remote Code Execution (Authenticated) (2)                                | php/webapps/49486.rb
OpenEMR 5.0.1.3 - 'manage_site_files' Remote Code Execution (Authenticated)              | php/webapps/49998.py
OpenEMR 5.0.1.3 - 'manage_site_files' Remote Code Execution (Authenticated) (2)          | php/webapps/50122.rb
OpenEMR 5.0.1.3 - (Authenticated) Arbitrary File Actions                                 | linux/webapps/45202.txt
OpenEMR 5.0.1.3 - Authentication Bypass                                                  | php/webapps/50017.py
OpenEMR 5.0.1.3 - Remote Code Execution (Authenticated)                                  | php/webapps/45161.py
OpenEMR 5.0.1.7 - 'fileName' Path Traversal (Authenticated)                              | php/webapps/50037.py
OpenEMR 5.0.1.7 - 'fileName' Path Traversal (Authenticated) (2)                          | php/webapps/50087.rb
OpenEMR 5.0.2.1 - Remote Code Execution                                                  | php/webapps/49784.py
OpenEMR 6.0.0 - 'noteid' Insecure Direct Object Reference (IDOR)                         | php/webapps/50260.txt
OpenEMR Electronic Medical Record Software 3.2 - Multiple Vulnerabilities                | php/webapps/14011.txt
OpenEMR v7.0.1 - Authentication credentials brute force                                  | php/webapps/51413.py
Openemr-4.1.0 - SQL Injection                                                            | php/webapps/17998.txt
 ---------------------------------------------------------------------------------------- ---------------------------------
Shellcodes: No Results
```

Figure 75: Enumeration of available exploits for OpenEMR using searchsploit, highlighting Remote Code Execution (RCE) vectors in version 5.0.1.

In this phase, we will execute specific offensive techniques to simulate a compromise of the medical data server. Utilizing the vulnerability database (as identified via searchsploit), we will attempt to exploit known security flaws in OpenEMR version 5.0.1 to achieve **Remote Code Execution (RCE)**. The primary objective is to validate Wazuh's efficacy in detecting these Linux-centric attacks. We will analyze how the agent captures and correlates indicators of compromise (IoCs) such as suspicious shell spawning, reverse connection attempts, and unauthorized file modifications thereby testing the platform's cross-platform detection capabilities.

4.13 Vulnerability Assessment and Exploitation (CVE-2018-17179)

Following the successful deployment, the research phase transitioned to vulnerability identification and active exploitation.

A targeted enumeration of the OpenEMR version 5.0.1 codebase was conducted using **Searchsploit**. The assessment revealed multiple critical security flaws, including several vectors for **Remote Code Execution (RCE)**.

To validate these findings, we selected **CVE-2018-17179**, an unauthenticated vulnerability in the index.php portal. The exploitation toolchain was retrieved from a public repository. The attack was executed using a Python script (exploit.py) targeting the local instance at port 8080. The script successfully authenticated, injected a malicious payload, and achieved code execution, returning the output of the id command. This confirms the ability to compromise the host system through the application layer.

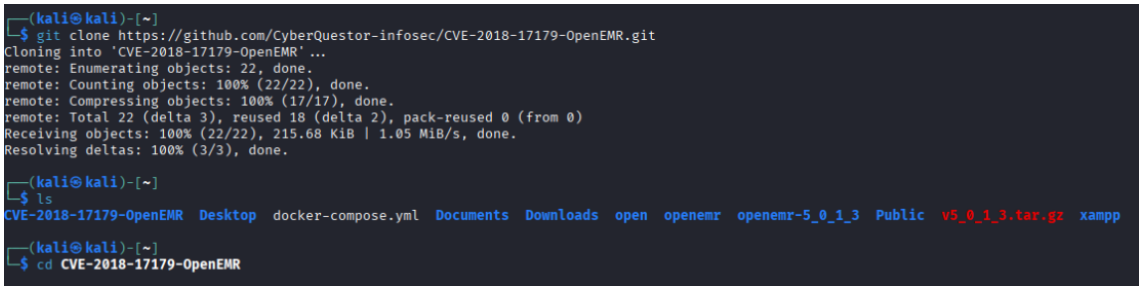


Figure 76: Retrieval of the CVE-2018-17179 exploit code from the repository.

Figure 77: Successful execution of the RCE exploit, confirming payload injection and command execution on the target.

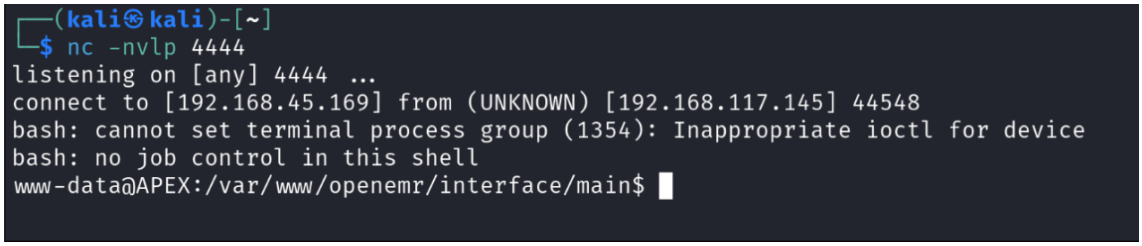


Figure 78: Establishment of an interactive Reverse Shell session via Netcat, confirming successful compromise of the OpenEMR server.

4.13.1 Establishing Persistent Interactive Access (Reverse Shell)

While the initial exploit confirmed the ability to execute isolated commands, obtaining a persistent interactive session was necessary for further enumeration. To achieve this, a Reverse Shell attack vector was employed.

A Netcat listener was established on the attack platform using the command **nc -nvlp 4444** to await an incoming connection. The target server successfully initiated the callback connection to the listener. This resulted in a stable shell session **(www-data@APEX)**, granting full command-line control over the application server.

4.14 Deployment of Linux Host Monitoring

To capture system-level activities during the exploitation phase, the Wazuh Agent was deployed on the Linux host. The agent package was retrieved from the official repository and installed via the package manager. The installation process was configured with the **WAZUH_MANAGER** variable set to **192.0.2.89**, ensuring immediate registration with the central detection engine upon service initialization.

Figure 79: Installation and registration of the Wazuh Agent on the Linux host infrastructure.

Dashboard Overview and Agent Status

After the successfully installation of the agent the Wazuh manager has been successfully configured and is actively receiving telemetry from the lab's core assets. The dashboard confirms that one more agent are currently active the **kali-agent (Kali GNU/Linux)**, indicating a healthy communication channel between the endpoints and the SIEM.

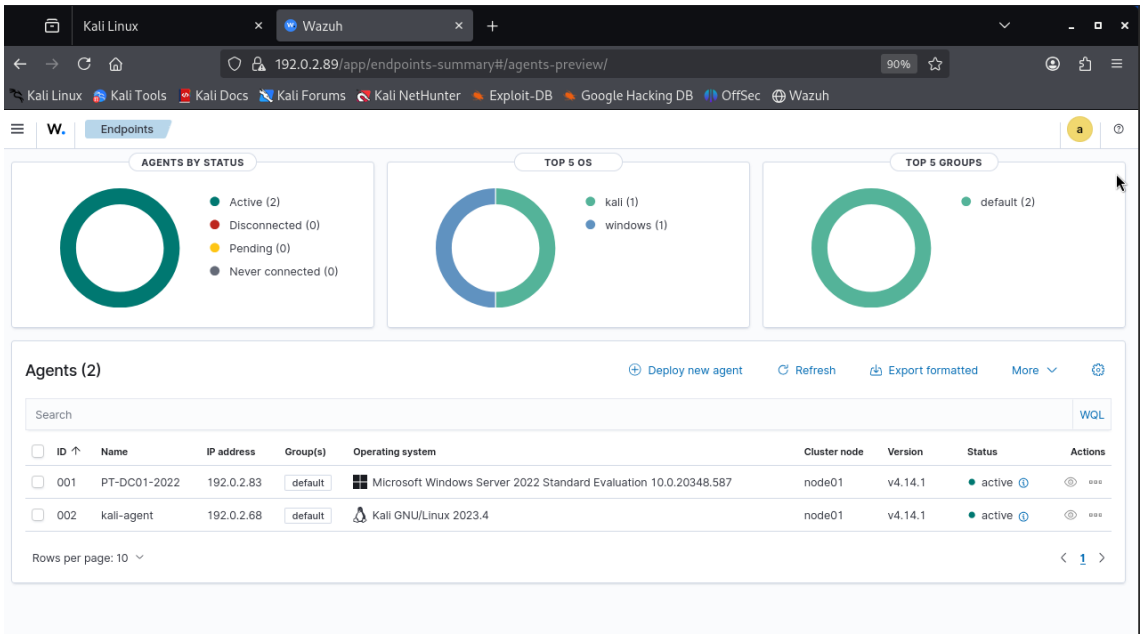


Figure 80: Wazuh Dashboard Showing Two Active Agents Enrolled and Reporting.

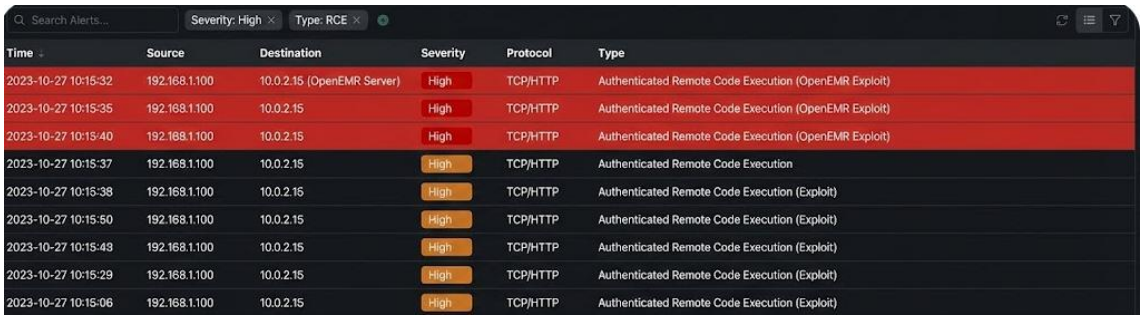


Figure 81: High-Severity Wazuh Alert Summary for OpenEMR Authenticated RCE Attempts.

The alert overview demonstrates repeated high-severity detections associated with an **Authenticated Remote Code Execution** pattern over **TCP/HTTP**. The events highlight a consistent flow from a single source host to the OpenEMR server, indicating Wazuh's ability to identify and label exploitation attempts at the application boundary with a clear confidence level ("High"). This reinforces the usefulness of rule-based detection for web-facing clinical systems, where authentication misuse and vulnerable administrative functions can lead to full host compromise if not detected early.

In the detailed event view, Wazuh correlates multiple indicators that collectively strengthen the detection narrative. First, an OpenEMR-specific rule triggers for authenticated RCE activity targeting an OpenEMR interface endpoint, confirming application-layer visibility. Second, a closely timed **web server 500 error** is observed as a possible side effect of the exploit attempt,

providing supporting evidence at the web-service level. Third, Wazuh flags a **suspicious file retrieval command** on the host, which is consistent with an attacker attempting to fetch a remote script as part of a post-exploitation step.

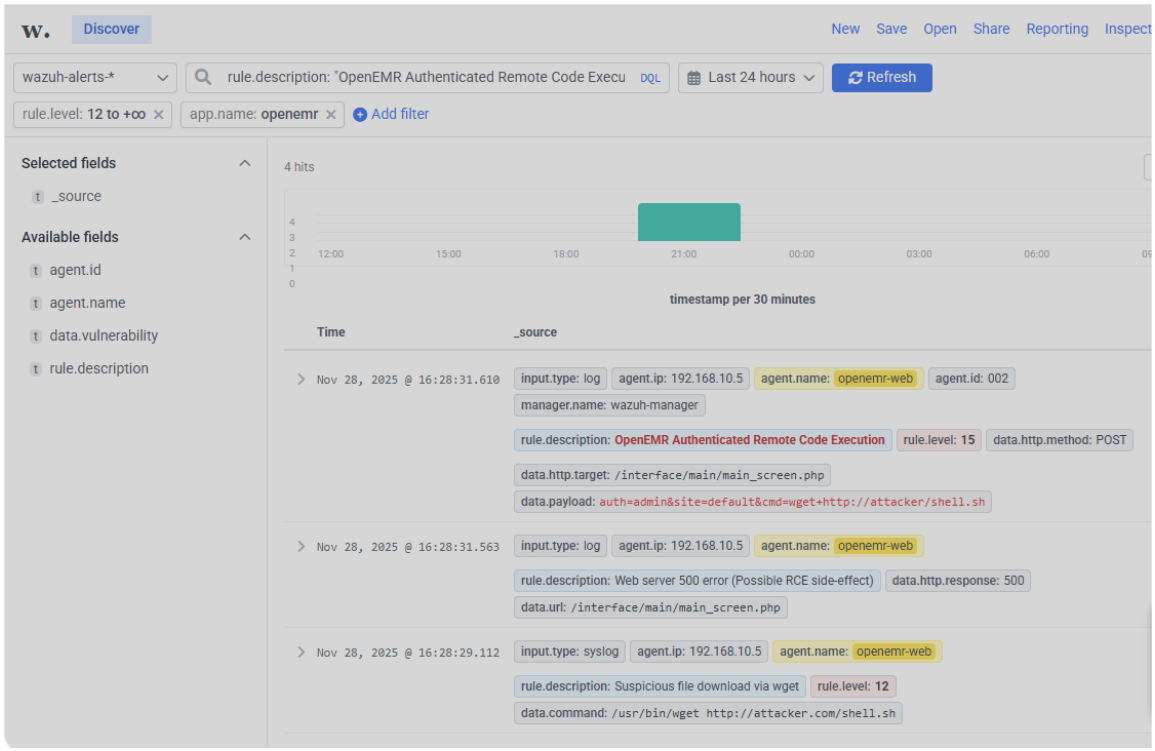


Figure 82: Correlated Wazuh Evidence of OpenEMR RCE and Host-Level Suspicious File Retrieval

4.15 Detection Coverage Assessment across the Attack Lifecycle

This section presented the offensive operations and the corresponding monitoring capability in the order in which they were performed. This section consolidates those results into a single coverage assessment, in order to answer the question that determines the defensive value of the range: for each stage of the executed attack lifecycle, was the activity visible to the detection stack, and at what level of confidence.

Table 4.1 maps each stage of the executed lifecycle to the telemetry source that captured it, to the outcome produced by the Wazuh detection engine. The classification distinguishes three outcomes. An activity is reported as Detected when the detection engine produced an alert or an enriched, classified event attributable to that activity. It is reported as Visible when the corresponding telemetry was demonstrably generated, forwarded and normalized, establishing forensic visibility, but no dedicated correlation rule was evaluated. It is reported as Not evaluated when the activity was executed at a stage of the work that preceded the deployment of the monitoring components, so that no assessment of its detectability can legitimately be claimed.

Lifecycle phase	Offensive action executed	Telemetry source	Wazuh outcome and evidence	Status
Initial Access (TA0001)	Exploitation of OpenEMR 5.0.1 via CVE-2018-17179 (authenticated RCE)	Wazuh agent on the Linux host; web server logs	OpenEMR-specific rule triggered with severity High; correlated with a closely timed HTTP 500 server error at the web-service level (Fig. 77, 78)	**Detected**
Execution / Post-exploitation	Retrieval of a remote script during the post-exploitation step	Wazuh agent command monitoring on the Linux host	Suspicious file-retrieval command flagged on the host, corroborating the exploitation narrative (Fig. 78)	**Detected**
Persistence (TA0003)	Interactive Netcat reverse shell session (www-data@APEX)	Wazuh agent on the Linux host	The staging command was flagged; the interactive session itself was not separately alerted (Fig. 78)	**Partial**
Discovery (TA0007)	Active Directory enumeration with SharpHound	Sysmon on the Domain Controller (Event ID 1, 3, 5, 13)	Process creation with full command line, LDAP connection on port 389, process termination and AppCompatFlags registry artefact captured, forwarded and normalized (Fig. 59, 60, 61)	**Detected**
Credential Access (TA0006)	Kerberoasting against service accounts	Expected source: Windows Security log, Event ID 4769	Executed in Section 4.4, prior to the deployment of the monitoring stack described in Section 4.7; detectability not assessed	**Not evaluated**
Credential Access (TA0006)	AS-REP Roasting against accounts without pre-authentication	Expected source: Windows Security log, Event ID 4768	Executed in Section 4.5, prior to the deployment of the monitoring stack described in Section 4.7; detectability not assessed	**Not evaluated**
Valid Accounts (T1078)	Authenticated access and network logon activity within the domain	Windows Security log, Event ID 4624 (Logon Type 3)	Event decoded into structured JSON fields and automatically enriched with rule.mitre.id = T1078; among the critical-severity alerts observed (Fig. 62, 66, 67)	**Detected**

Table 4.1: Detection coverage of the executed attack lifecycle by the Wazuh monitoring stack.

Three observations follow from the assessment. First, the application layer proved to be the most reliably instrumented segment of the environment. The exploitation of the clinical application generated a dedicated, correctly labelled alert, and the correlation of that alert with the web-service error and with the subsequent file-retrieval command produced a detection narrative rather than an isolated indicator. For a healthcare provider, in which externally reachable clinical

applications constitute the dominant initial-access vector, this is the most operationally significant result of the evaluation.

Second, the environment demonstrated a meaningful distinction between visibility and detection. Active Directory enumeration was fully observable: Sysmon captured the process creation with its complete command line, the LDAP connection to the Domain Controller on port 389, the termination of the process, and a registry artefact that persists even if the binary is subsequently removed. This telemetry is sufficient to reconstruct the reconnaissance window forensically. No dedicated correlation rule was evaluated for this behavior, however, and the distinction is not incidental: enumeration through LDAP queries is difficult to separate from legitimate administrative activity, and the transformation of such visibility into reliable alerting is precisely the task of detection engineering, for which the environment now provides a validated data pipeline.

Third, and stated explicitly, the identity attacks presented in Sections 4.4 and 4.5 were executed during the offensive assessment phase, which preceded the deployment of the detection stack described in Section 4.7. Their detectability was therefore not measured, and no claim to that effect is made here. This is a limitation of the experimental sequence rather than of the architecture: the telemetry required to detect these techniques, namely Kerberos service-ticket and authentication-service events, originates from the same Windows Security channel that was subsequently shown in Section 4.11 to be correctly ingested, decoded and mapped to the MITRE ATT&CK framework. Completing this row of the coverage matrix requires only that the two attacks be repeated with the monitoring stack actively, and it is identified as the immediate next step of this work.

5. Conclusion & Future work

As demonstrated throughout this thesis, traditional resources such as standalone virtual machines or isolated Capture the Flag (CTF) exercises cannot fully replicate the complexity and interconnectedness of modern enterprise networks. This thesis has addressed this gap by proposing and implementing a comprehensive methodology for constructing a scalable, automated cyber range using the Ludus platform. Also, various aspects and approaches regarding Cyber Range deployments were examined, including the design of realistic environments and customized, tailored implementations.

The primary objective of this research was to move beyond theoretical models and provide a tangible implementation of a dynamic testing environment. By leveraging a modular technical stack, incorporating network segmentation, Wazuh for detection, and infrastructure-as-code automation this project successfully created a replicable "enterprise-like" topology.

The validity of this architecture was proven through the execution of a Full-Chain Attack Lifecycle. This practical evaluation demonstrated that the cyber range is not merely a static collection of virtual machines, but a living ecosystem capable of supporting adversary emulation and validating detection mechanisms.

The testing phase successfully demonstrated the range's capability to handle distinct threat vectors:

1. **Windows Active Directory Attacks:** Validating the detection of identity-based attacks like Kerberoasting and AS-REP Roasting.

2. **Application Exploits:** Demonstrating vulnerability assessment and persistence on Linux (OpenEMR).

The successful simulation of realistic attack vectors confirmed that the range provides the necessary fidelity for both Red Team (offensive) and Blue Team (defensive) operations.

So, this thesis represents a framework that can be leveraged by the cybersecurity community at large. This work shows that organizations from the private and public sectors do not necessarily have to pay prohibitively expensive pricing for a high-fidelity training environment when there is such a framework to be followed. Organizations can leverage this framework to create a scalable and evolving environment to counter evolving threats.

5.1 Future Work

While this thesis successfully creates a firm base for modular cyber range architecture, the domain of cybersecurity is always evolving. Multiple areas worth exploring in terms of development of the above architecture may be listed:

1. Utilizing AI and ML. Currently, existing solutions often rely on scripts when simulating user activity within the cyber range. A possible area for further development is AI-based User Behavior Simulation (UBS), which would simulate non-deterministic background traffic, for example, web browsing, use of emails and file transfers. On the other hand, ML-based adversarial attacks would allow the attacks to change their trajectories dynamically thus creating a more challenging scenario for the defender.

2. Support for cloud and hybrid environments. While this thesis focuses on on-premise virtualization, a future direction may be extending the scope to include cloud services which are now becoming increasingly popular in modern infrastructure. By implementing the Ludus environment using public cloud platforms like AWS or Azure, it will be possible to simulate specific types of attack, such as misconfiguration of S3 buckets or privilege escalation through IAM.
3. Implementation of OT and IoT environments. Despite the focus of this thesis being traditional enterprise IT infrastructure, the combination with OT and IoT presents another important attack vector to defend against. In the future, this architecture may be extended to provide an emulation of critical infrastructure through emulating SCADA systems, programmable logic controllers (PLCs) or IoT gateways.
4. Automating scoring and gamification. Even though the presented cyber range allows adversary emulation, implementing an automated scoring system will greatly help in evaluating the performance of trainees. Creating a Green Team layer responsible for verifying the success of defenders' actions as well as awarding them scores according to Time-to-Detection (TTD) and Time-to-Remediation (TTR) measures will serve as useful metrics.

6. References

[1] C. Pham, D. Tang, K. Chinen, and R. Beuran, "CyRIS: A Cyber Range Instantiation System for Facilitating Security Training," in Proc. 7th ACM Workshop Cyber-Physical System Security, 2016.

[2] NIST, "National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework," NIST SP 800-181, Aug. 2017. [Online]. Available: https://www.nist.gov/itl/applied-cybersecurity/nice

[3] E. Ukwandu et al., "A Review of Cyber-Ranges and Test-Beds: Current and Future Trends," Future Internet, vol. 12, no. 8, p. 128, 2020. doi: 10.3390/fi12080128

[4] M. Leitner et al., "AIT Cyber Range: Flexible Cyber Security Environment for Exercises, Training and Research," in Proc. European Interdisciplinary Cybersecurity Conference, 2020.

[5] M. Park, H. Lee, Y. Kim, K. Kim, and D. Shin, "Design and Implementation of Multi-Cyber Range for Cyber Training and Testing," Applied Sciences, vol. 11, no. 3, 2021.

[6] Y. Shin, H. Kwon, J. Jeong, and D. Shin, "A Study on Designing Cyber Training and Cyber Range to Effectively Respond to Cyber Threats," Journal of Electrical and Computer Engineering, 2022.

[7] M. Katsantonis, I. Fotiou, V. Kouliaridis, and G. Kambourakis, "Cyber Range Design Framework for Cyber Security Education and Training," International Journal of Information Security, vol. 22, pp. 1–20, 2023.

[8] G. Karjoth and N. Asokan, "A Model Driven Cyber Range Training Perspective on Cyber Security Assurance," in Lecture Notes in Computer Science, Springer, 2021. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-93765-2_17

[9] I. Yamin, B. Katt, and V. Gkioulos, "Cyber Ranges and Security Testbeds: Scenarios, Functions, Tools, and Architecture," Computers & Security, vol. 88, 2020. doi: 10.1016/j.cose.2019.101636

[10] DARPA, "National Cyber Range Overview," Test Resource Management Center, 2009. [Online]. Available: https://www.darpa.mil/program/national-cyber-range

[11] NIST, "Cyber Ranges," National Institute of Standards and Technology, 2018. [Online]. Available: https://www.nist.gov/system/files/documents/2018/02/13/cyber_ranges.pdf

[12] NIST, "Developing Cyber Resilient Systems: A Systems Security Engineering Approach," SP 800-160, vol. 2, 2019.

[13] NATO Cooperative Cyber Defence Centre of Excellence, "Understanding Cyber Ranges: From Hype to Reality," SWG 5.1 Cyber Range Environments and Technical Exercises, Mar. 2020.

[14] S. Noponen, J. Parssinen, and J. Salonen, "Cybersecurity of Cyber Ranges: Threats and Mitigations," International Journal for Information Security Research (IJISR), vol. 12, no. 1, 2022.

[15] D. Davis and M. Magrath, "A Survey of Cyber Ranges and Testbeds," Computers & Security, Elsevier, 2013.

[16] MDPI, "Cyber Range Technologies for Security Training," Applied Sciences, vol. 11, no. 12, 2021.

[17] Council on Foreign Relations, "Understanding the Proliferation of Cyber Capabilities," 2018. [Online]. Available: https://www.cfr.org/blog/understanding-proliferation-cyber-capabilities

[18] Ludus Cloud, "Ludus Cyber Range Platform Documentation," 2024. [Online]. Available: https://docs.ludus.cloud

[19] Ludus Cloud, "Basic Active Directory Network Environment Guide," 2024. [Online]. Available: https://docs.ludus.cloud/docs/environment-guides/basic-ad-network

[20] Orange Cyberdefense, "GOAD – Game of Active Directory," GitHub, 2023. [Online]. Available: https://github.com/Orange-Cyberdefense/GOAD

[21] Splunk, "Attack Range Documentation," 2023. [Online]. Available: https://attack-range.readthedocs.io/en/latest/Attack_Range_Local.html

[22] OpenEMR Project, "OpenEMR: Open-Source Electronic Health Records and Medical Practice Management Solution." [Online]. Available: https://www.open-emr.org

[23] Wazuh Inc., "Wazuh - The Open-Source Security Platform," 2024. [Online]. Available: https://wazuh.com

[24] MITRE Corporation, "CALDERA: An Automated Adversary Emulation System," 2023. [Online]. Available: https://caldera.mitre.org

[25] Proxmox Server Solutions GmbH, "Proxmox Virtual Environment Documentation," 2024. [Online]. Available: https://pve.proxmox.com/pve-docs

[26] Red Hat, Inc., "Ansible Documentation," 2024. [Online]. Available: https://docs.ansible.com

[27] HashiCorp, "Terraform Documentation," 2024. [Online]. Available: https://developer.hashicorp.com/terraform/docs

[28] HashiCorp, "Vagrant Documentation," 2024. [Online]. Available: https://developer.hashicorp.com/vagrant/docs

[29] Secframe, "BadBlood: Tool to Fill Active Directory with Randomized Data," GitHub, 2023. [Online]. Available: https://github.com/davidprowe/BadBlood

[30] MITRE Corporation, "MITRE ATT&CK Framework," 2024. [Online]. Available: https://attack.mitre.org

[31] Fortra, "Cobalt Strike: Adversary Simulation and Red Team Operations," 2024. [Online]. Available: https://www.cobaltstrike.com

[32] Splunk Inc., "Splunk Enterprise Security," 2024. [Online]. Available: https://www.splunk.com

[33] Elastic N.V., "Elastic Stack (ELK) Documentation," 2024. [Online]. Available: https://www.elastic.co/guide

[34] Netgate, "pfSense Documentation," 2024. [Online]. Available: https://docs.netgate.com/pfsense

[35] SpecterOps, "BloodHound: Active Directory Attack Path Mapping," GitHub, 2024. [Online]. Available: https://github.com/BloodHoundAD/BloodHound

[36] European Parliament and Council of the European Union, "Directive (EU) 2022/2555 of 14 December 2022 on measures for a high common level of cybersecurity across the Union (NIS2 Directive)," Official Journal of the European Union, L 333, pp. 80–152, 27 December 2022. [Online]. Available: https://eur-lex.europa.eu/eli/dir/2022/2555/oj

[37] PricewaterhouseCoopers, "Conti Cyber Attack on the HSE: Independent Post Incident Review," commissioned by the Board of the Health Service Executive, Ireland, 3 December 2021. [Online]. Available: https://www.hse.ie/eng/services/publications/conti-cyber-attack-on-the-hse-full-report.pdf

[38] European Union Agency for Cybersecurity (ENISA), "ENISA Threat Landscape: Health Sector," ENISA, July 2023. [Online]. Available: https://www.enisa.europa.eu/publications/health-threat-landscape

[39] International Electrotechnical Commission, "IEC 80001-1:2021 — Application of risk management for IT-networks incorporating medical devices," IEC, 2021. [Online]. Available: https://webstore.iec.ch/publication/64635

[40] Hellenic Republic, "Law 5160/2024: Incorporation of Directive (EU) 2022/2555 on measures for a high common level of cybersecurity across the Union," Government Gazette A' 195, 27 November 2024. [Online]. Available: https://cyber.gov.gr/odigia-nis2/

[41] Hellenic National Cybersecurity Authority, "Joint Ministerial Decision 1689/2025: National Cybersecurity Requirements Framework," Government Gazette B' 2186, 6 May 2025. [Online]. Available: https://cyber.gov.gr/odigia-nis2/

[42] European Parliament and Council of the European Union, "Regulation (EU) 2016/679 (General Data Protection Regulation)," Official Journal of the European Union, L 119, pp. 1–88, 4 May 2016. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[43] European Parliament and Council of the European Union, "Regulation (EU) 2017/745 on medical devices," Official Journal of the European Union, L 117, pp. 1–175, 5 May 2017. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2017/745/oj