Cyber–Ed A digital Hands-on platform for learning Cybersecurity

Alexandra Sofia Dias Alves Rocha - a41792

Dissertação apresentada à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Informática.

Trabalho orientado por:

Prof. Tiago Miguel Ferreira Guimarães Pedrosa

Prof. Rui Pedro Sanches de Castro Lopes

Bragança

Outubro, 2025

Cyber–Ed A digital Hands-on platform for learning Cybersecurity

Alexandra Sofia Dias Alves Rocha - a41792

Dissertação apresentada à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Informática.

Trabalho orientado por:

Prof. Tiago Miguel Ferreira Guimarães Pedrosa

Prof. Rui Pedro Sanches de Castro Lopes

Bragança

Outubro, 2025

Dedicatória

Dedico esta tese à minha família, por serem o pilar de todo este caminho. Em especial ao meu querido avô, que sempre desejou ver-me concluir esta etapa — acreditando, com o seu bom humor habitual, que isso me garantiria um aumento. Às minhas avós, que partilhavam dessa mesma convicção e cujo carinho, força e incentivo acompanharam este percurso. Que este trabalho seja a prova de que o esforço vale a pena e que ele talvez tivesse razão.

Agradecimentos

Em primeiro lugar, um agradecimento profundo aos meus pais, pelo amor incondicional, pela paciência e pelo apoio constante em todas as fases da minha vida. São o meu maior exemplo de força, dedicação e perseverança.

Aos meus avós, por todo o carinho e por nunca deixarem de acreditar em mim. Ás minhas avós, com especial carinho para aquela cujo sorriso sobressai sempre que me vê, as conversas partilhadas, as histórias cheias de vida e os risos sinceros foram, e continuam a ser, um conforto e uma inspiração ao longo deste percurso; e ao meu avô, cuja confiança e sentido de humor continuam a motivar-me e a fazer-me sorrir.

Aos meus tios e primos, pelo apoio, pela amizade e por todas as palavras de encorajamento ao longo deste percurso — cada gesto contou.

Ao Rui, pela amizade, pela paciência e, acima de tudo, pela boa disposição ao longo de todo este processo.

Aos meus orientadores, Professor Tiago Pedrosa e Professor Rui Pedro, pela orientação, disponibilidade, interesse e paciência que tiveram para comigo durante o desenvolvimento deste trabalho.

A todos os que, de alguma forma, contribuíram para a realização deste trabalho, o meu mais sincero obrigado.

"Persistir, insistir e nunca desistir."

Resumo

A cibersegurança afirma-se como uma área crítica na sociedade contemporânea, especialmente após a pandemia de COVID-19, que acelerou a transformação digital e expôs vulnerabilidades crescentes. A falta de profissionais qualificados para enfrentar estes desafios demonstra a necessidade de metodologias educativas que promovam competências práticas, adaptativas e alinhadas com tecnologias emergentes. O estudo inclui uma revisão sistemática que fundamenta as opções metodológicas, analisando vantagens e limitações de abordagens tradicionais e inovadoras, com foco na gamificação, na Inteligência Artifical (IA) e em ambientes virtuais.

Esta dissertação propõe uma plataforma educativa automatizada que combina laboratórios virtuais (Labtainers) e competições Capture The Flag (CTF) em ambientes dinâmicos, seguros e acessíveis via virtual private network (VPN). A solução integra princípios de gamificação e automação, utilizando tecnologias como Terraform, Ansible, React, FastAPI e Proxmox, com o objetivo de proporcionar experiências de aprendizagem realistas, diversificadas e inclusivas na formação em cibersegurança. Foram implementados e testados alguns cenários de aprendizagem prática, abrangendo ataques simulados, como *SQL Injection (SQLI)*, e exercícios de administração de sistemas. Os resultados demonstram eficácia na criação de cenários e potencial de aplicação em contextos académicos e empresariais. Apesar de limitações, como a capacidade atual de utilizadores simultâneos e a diversidade restrita de cenários, a plataforma constitui uma base sólida para futuras expansões, incluindo a migração para infraestruturas *cloud* e a personalização mediada por IA. Este trabalho contribui, assim, para o avanço do ensino prático de cibersegurança, preparando profissionais para um panorama digital em rápida transformação.

Palavras-chave: Cibersegurança, Automação, CTF, Gamificação, Ensino, Treino

Abstract

Cybersecurity is asserting itself as a critical area in contemporary society, especially after the COVID-19 pandemic, which has accelerated digital transformation and exposed growing vulnerabilities. The lack of qualified professionals to face these challenges demonstrates the need for educational methodologies that promote practical skills, adaptive and aligned with emerging technologies. The study includes a systematic review that substantiates the methodological options, analyzing advantages and limitations of traditional and innovative approaches, focusing on gamification, IA and virtual environments.

This dissertation proposes an automated educational platform that combines virtual laboratories (Labtainers) and CTF competitions in dynamic, safe and accessible environments via VPN. The solution integrates gamification and automation principles, using technologies such as Terraform, Ansible, React, FastAPI and Proxmox, with the aim of providing realistic, diverse and inclusive learning experiences in cybersecurity training. Some practical learning scenarios were implemented and tested, including simulated attacks such as SQL injection and system administration exercises. The results demonstrate effectiveness in creating scenarios and potential for application in academic and business contexts. Despite limitations, such as the current capacity of concurrent users and the restricted diversity of scenarios, the platform provides a solid foundation for future expansions, including migration to infrastructures *cloud* and artificial intelligence (AI) mediated personalization. This work thus contributes to the advancement of practical cybersecurity education, preparing professionals for a rapidly changing digital landscape.

Keywords: Cibersecurity, Automation, CTF, Gamification, Teaching, Training

Conteúdo

Lista de Tabelas

Lista de Figuras

Siglas

AI artificial intelligence. viii

API Application Programming Interface. 11, 21–24, 29

AWS amazon virtual services. 11

CPS cyber physical system. 22

CTF Capture The Flag. vii, viii, 1–7, 10–12, 14, 19, 20, 24, 27–30, 32

DDOs distributed denial-of-service. 22

EC2 amazon elastic compute. 11

GCP google cloud platform. 11

HCL hashiCorp configuration language. 27

IA Inteligência Artifical. vii, viii, 4–6, 10, 15, 16, 22, 24, 25, 27, 29, 30, 32, 69

IDS intrusion detection system. 13

IoT internet of things. 16, 25, 29

LLM large language models. 14, 24, 25, 29, 32

LxC linux containers. 27

MFA multi-factor authentication. 13

ML machine learning. 11, 24

OT Orientação Tutorial. 22

SQLI SQL Injection. vii, 2

SSH secure shell. 28

TI Tecnologias da Informação. 16, 17

VM virtual machine. 7, 10, 14

VPC virtual private cloud. 11

VPN virtual private network. vii, viii, 1

Capítulo 1

Introdução

A cibersegurança é importante na sociedade atual, especialmente após a pandemia COVID-19, que intensificou a digitalização e o trabalho remoto, ampliando vulnerabilidades e *cyberthreats* como *phishing* e *ransomware* [1]. Segundo o Relatório de Cibersegurança da Kaspersky de 2021, os ataques de *phishing* cresceram 35% globalmente durante a pandemia, com destaque para ambientes de trabalho remoto [2]. A adoção de plataformas de videoconferência, como o Zoom, e a dependência de redes domésticas e VPNs expuseram organizações e indivíduos a riscos crescentes, mostrando assim a necessidade de formação prática robusta. Nesse contexto, ferramentas educativas inovadoras, como laboratórios virtuais e CTF, desempenham um papel importante na formação de profissionais.

Para enfrentar a falta de profissionais qualificados e promover a educação em cibersegurança, têm sido desenvolvidas diversas ferramentas e metodologias de ensino prático. Entre as mais eficazes destacam-se os Labtainers, que utilizam *containers* Docker para criar laboratórios virtuais isolados, e os CTF, que simulam cenários reais de ciberataques, como testes de penetração [3]-[4]. Por um lado, os Labtainers oferecem uma *framework* baseada em Docker, o que permite a criação de laboratórios com múltiplos componentes e cenários personalizáveis. Estes além de serem facilmente instalados em computadores, asseguram um ambiente de execução uniforme o que otimiza a criação e a avaliação de exercícios práticos. Por outro lado, as competições CTF são reconhecidas pela sua abordagem lúdica, devido à sua capacidade em proporcionar uma educação em cibersegurança e em promover simultaneamente a aquisição de competências técnicas, mesmo em ambientes virtuais com as devidas adaptações, à semelhança de como decorreu durante a

pandemia [4]. Assim, a monitorização dos riscos de segurança em dispositivos de teletrabalho realça a relevância do domínio e controlo de cenários específicos para os contextos referidos.

Apesar da robustez de ferramentas como Labtainers e CTF, persistem lacunas significativas, incluindo a falta de soluções educativas que simulem de forma real ameaças específicas do trabalho remoto e do ensino *online*, bem como a complexidade de gestão de configurações em ambientes híbridos. Essas limitações dificultam a formação prática e destacam a necessidade de uma plataforma automatizada que integre cenários realistas e acessíveis.

Assim, com este trabalho propõe-se uma plataforma educativa inovadora que integra simulações práticas baseadas em Labtainers e CTF, inspiradas em contextos como o modelo *work-from-home* e plataformas de ensino virtual, como o Zoom. Por meio de automação e cenários realistas, como *SQLI* e segurança de redes, a solução visa capacitar utilizadores e organizações para enfrentar os desafios do mundo digital, promovendo competências práticas, motivação e acessibilidade. Face a esta necessidade, o presente trabalho define um conjunto de objetivos apresentados de seguida, que orientam o desenvolvimento e a validação deste projeto.

1.1 Objetivos

Este trabalho tem como objetivo principal o desenvolvimento de uma plataforma educativa, orientada para a aprendizagem prática em cibersegurança, que combine laboratórios virtuais dinâmicos e desafios do tipo CTF. A solução procura superar as limitações das ferramentas tradicionais, frequentemente assentes em configurações estáticas e de difícil gestão, oferecendo em contrapartida um ambiente mais flexível, acessível e automatizado.

De forma mais detalhada, pretende-se promover a formação prática em cibersegurança através da disponibilização de laboratórios adaptáveis a diferentes níveis de conhecimento, assegurando ao mesmo tempo a integração de mecanismos de gamificação que contribuam para aumentar a motivação, o envolvimento e a retenção dos utilizadores no processo de aprendizagem. A plataforma proposta deverá incorporar cenários realistas de ameaças cibernéticas, focando-se em áreas como a segurança de redes, a administração de sistemas, a exploração de vulnerabilidades e a simulação de ataques comuns, de forma a preparar

os utilizadores para riscos associados aos contextos digitais atuais.

Adicionalmente, pretende-se atingir um elevado grau de automação na criação e configuração dos ambientes, simplificando a gestão, reduzindo significativamente o tempo de preparação e ampliando a escalabilidade do sistema, de modo a responder à crescente procura por formação especializada. Por fim, um objetivo complementar desta investigação consiste em contribuir para a definição de boas práticas no uso da automação aplicada a plataformas educativas de cibersegurança, reforçando a acessibilidade e a resiliência digital tanto em ambientes académicos como profissionais.

1.2 Estrutura do Documento

Este documento está organizado em seis capítulos:

Capítulo 1 introduz o projeto, contextualizando a relevância da cibersegurança no cenário pós-pandémico, apresentando os objetivos do trabalho e destacando a importância de uma solução automatizada para a formação prática.

Capítulo 2 centra-se no estado da arte, ilustrando os conceitos fundamentais da área, analisando ferramentas como os Labtainers, os CTF e soluções existentes.

Capítulo 3 apresenta a proposta da plataforma, estruturada numa arquitetura em três camadas e salientando os benefícios associados à automação.

Capítulo 4 descreve a implementação técnica da solução, detalhando as tecnologias utilizadas.

Capítulo 5 é dedicado à avaliação da plataforma, com testes aplicados a diferentes cenários. Ainda neste capitulo são discutidos os resultados obtidos, comparando a solução desenvolvida com alternativas existentes.

Capítulo 6 apresenta as conclusões do trabalho, resumindo as principais contribuições da plataforma e propondo direções futuras.

Capítulo 2

Estado da Arte

A cibersegurança é essencial num mundo digital, protegendo sistemas, dados e utilizadores contra ameaças crescentes. Segundo o Relatório de Cibersegurança da ENISA (2024), os ataques de *ransomware* aumentaram 37% desde 2020, impulsionados pela transformação digital e pelo trabalho remoto pós-COVID-19 [5]. Métodos tradicionais de formação, como aulas teóricas, revelam-se insuficientes para preparar profissionais e estudantes para cenários reais, exigindo abordagens práticas que simulem ambientes profissionais em contextos seguros. Este capítulo analisa o estado da arte em soluções inovadoras para ensino e treino em cibersegurança, com foco em ambientes simulados (*cyber ranges*), competições gamificadas (CTF), e a integração de IA. Explora conceitos fundamentais, arquiteturas comuns, plataformas específicas, tendências emergentes e lacunas para pesquisas futuras.

A formação prática em cibersegurança baseia-se em abordagens que combinam simulação, gamificação e tecnologias avançadas. Estas soluções complementam-se para promover competências técnicas, motivação e trabalho em equipa, como detalhado a seguir.

Um *cyber range* é um ambiente virtual que replica redes e sistemas reais, permitindo treinos seguros de ataque e defesa. Funciona como um "campo de treino digital", onde equipas praticam estratégias e desenvolvem raciocínio crítico em cenários realistas, como simulações de redes empresariais ou infraestruturas SCADA [6]. A sua escalabilidade suporta múltiplos utilizadores e cenários complexos, sendo ideal para contextos académicos e corporativos [7], [8]. A Figura 2.1 ilustra a modularidade destes ambientes, com camadas de infraestrutura e gestão.

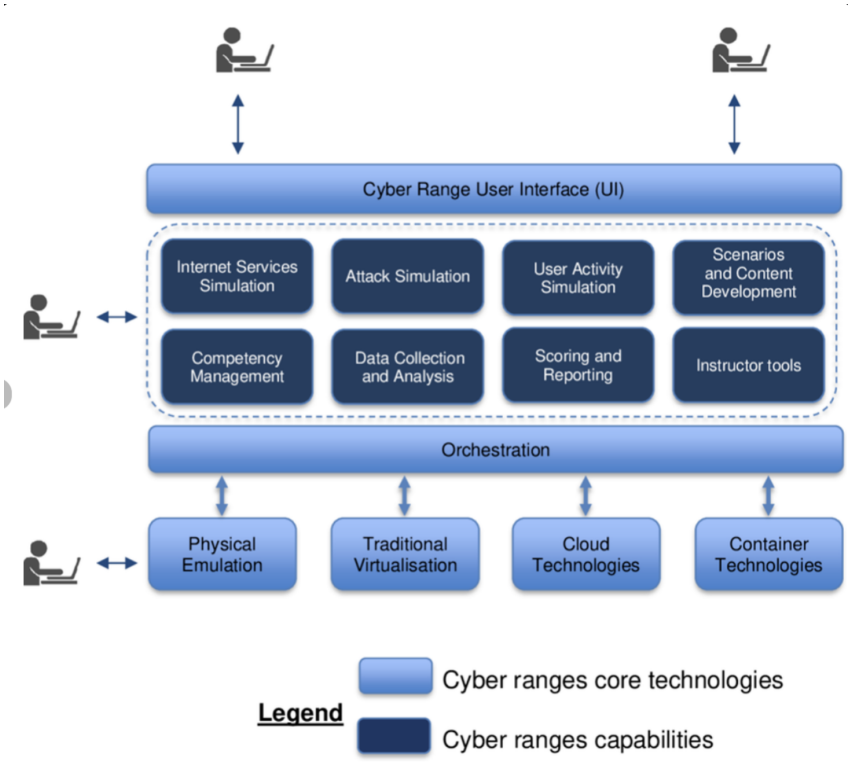

Figura 2.1: Arquitetura de um *Cyber Range*. Retirado de [9]

Gamificação aplica elementos de jogos, como pontos, níveis e *leaderboards*, em contextos educativos para aumentar o número de participantes. Em cibersegurança, transforma treinos em desafios interativos, promovendo retenção de conhecimento e motivação contínua [10], [11]. Por exemplo, recompensas simbólicas incentivam a repetição de exercícios, melhorando competências técnicas até 70% [12].

As competições CTF são desafios práticos que simulam cenários reais de cibersegurança, como exploração de vulnerabilidades ou análise forense. Originadas na DEF CON na década de 1990, os CTFs evoluíram para ferramentas educativas, com formatos como *Jeopardy* (desafios independentes em criptografia, *web exploitation*) e *Attack/Defense* (equipas protegem e atacam sistemas) [2], [13]. Estudos indicam que participantes regulares melhoram a deteção de vulnerabilidades em 75% [14]. A Figura 2.2 mostra a estrutura de um CTF.

IA, incluindo *machine learning* (ML) e modelos de linguagem (LLMs), personaliza treinos, adapta desafios ao nível do utilizador e fornece feedback em tempo real. Por exemplo, a IA pode ajustar a dificuldade de um CTF ou simular ataques realistas [15], [16]. Estas abordagens, combinadas, formam a base para plataformas modernas de treino.

Plataformas de treino em cibersegurança, como *cyber ranges* e CTFs, seguem arquiteturas modulares e escaláveis, geralmente baseadas em *cloud* ou virtualização. A estrutura típica inclui quatro camadas: a primeira, infraestrutura, com *containers* (ex.: Docker) ou VMs para ambientes isolados. A segunda, *backend*, que gere lógica de desafios e bases de dados (ex.: MySQL, Redis). A terceira, a camada do *frontend*, com interfaces web para interação e a quarta, monitorização, que usa IA ou ferramentas como *honeypots* para análise de desempenho e deteção de *cheating* [17]. Esta modularidade garante segurança, escalabilidade para milhares de utilizadores e adaptação a diferentes contextos, como educação ou treino corporativo. A Figura 2.2 exemplifica um diagrama de um CTF.

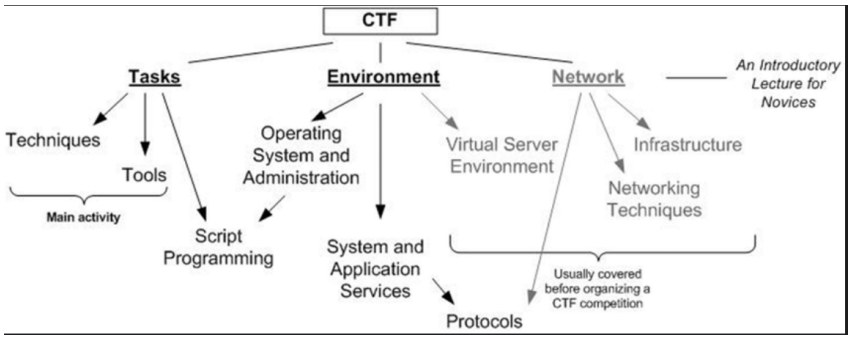

Figura 2.2: Diagrama de um CTF. Retirado de [18]

2.1 Abordagens Similares

A crescente procura por competências práticas em cibersegurança, impulsionada pela transformação digital e pela necessidade de ambientes de treino seguros, tem destacado plataformas como os Labtainers como ferramentas essenciais para a educação prática [6]. Esta *framework open-source*, desenvolvida pela Naval Postgraduate School, utiliza containers Docker para criar laboratórios virtuais isolados, permitindo aos estudantes realizar exercícios *hands-on*, como análise de redes, exploração de vulnerabilidades e engenharia reversa, sem os riscos associados a sistemas reais [17]. Esta plataforma oferece mais de 50 exercícios pré-definidos, muitos inspirados nos laboratórios SEED da Syracuse University, com personalização por aluno para desincentivar a cópia e facilitar a avaliação automática por instrutores [19]. Apesar da sua robustez em termos de segurança e reprodutibilidade, os Labtainers carecem de elementos nativos de gamificação, como rankings ou recompensas, o que pode comprometer a motivação a longo prazo [20]-[19].

No que respeita à relação entre os Labtainers e as competições CTF, é importante reconhecer que, embora ambos partilhem o objetivo de desenvolver competências práticas em cibersegurança, diferem significativamente na abordagem e na finalidade. Os Labtainers estão orientados para o contexto educativo formal, proporcionando laboratórios estruturados e isolados que permitem a realização de exercícios práticos num ambiente controlado. Já os CTF, como o picoCTF ou as competições da DEF CON, são eventos gamificados e competitivos, em que os participantes enfrentam desafios em tempo real, frequentemente em interação com adversários virtuais ou equipas rivais.

Nos Labtainers, exercícios como o "Network Sniffing Lab" ou o "SQL Injection Lab" reproduzem, em certa medida, a lógica de captura de bandeiras típica dos CTF, mas sem a componente de competição ou de pressão temporal. Estudos indicam que o uso de Labtainers melhora a deteção de vulnerabilidades em 75–80% [21], o que demonstra o seu valor como preparação técnica para desafios CTF. No entanto, a ausência de elementos de gamificação e a natureza local dos *containers* limitam uma integração mais profunda entre ambas as abordagens. Assim, é mais adequado considerar os Labtainers como uma base técnica complementar, potencialmente adaptável a mini-CTF educativos, onde a aprendizagem estruturada prevalece sobre a vertente competitiva.

A arquitetura dos Labtainers é modular e centrada em *containers*, dividindo-se em quatro camadas principais: a camada de infraestrutura, baseada em *containers* Docker e *hipervisores* como VirtualBox, que contém ambientes isolados e suporta deploy local ou em nuvem como Azure. Uma camada de backend e lógica de exercícios, que utiliza *scripts* de Python e Bash para configuração e avaliação automatizada, com bases de dados como SQLite para rastreamento e o "Lab Designer" para personalização via YAML. A camada de interface, composta por uma CLI, como os comandos "start_lab", e a camada de monitorização, que emprega *logging* Docker para capturar as interações e detetar fraude, garantindo avaliação precisa [22]. Esta estrutura, minimiza a manutenção e suporta atualizações automáticas via GitHub, embora dependa de Docker e não inclua gamificação nativa.

Exemplos de uso incluem o "Malware Analysis Lab", onde os alunos analisam amostras benignas com Ghidra em virtual machine (VM)s containerizadas, e o "Custom CTF Adaptation", aplicado em cursos da NPS para competições internas com *flags* personalizadas [20]. Estes casos destacam a versatilidade dos Labtainers para formação prática,

com guiões detalhados e disponíveis no site oficial, posicionando-a como uma plataforma reconhecida pela segurança e reprodutibilidade, embora menos interativa que alternativas gamificadas.

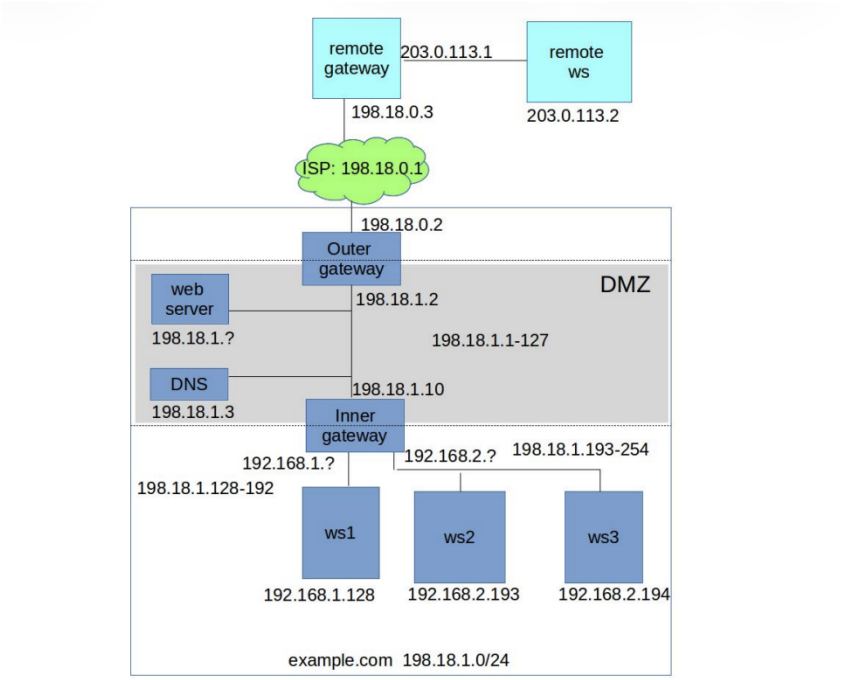

Figura 2.3: Exemplo de uma topoligia de rede de um Labtainer. Retirado de [23]

A topologia de rede exemplificada no "Labtainer Lab Designer User Guide" 2.3 ilustra uma configuração simples, mas representativa, de um laboratório de cibersegurança baseado em *containers* Docker, projetada para simular ambientes reais de forma isolada e controlada. Esta topologia demonstra como os Labtainers permitem a criação de redes virtuais compostas por múltiplos *containers*, interligados para exercícios práticos, como análise de tráfego, exploração de vulnerabilidades ou simulações de ataques e defesas. No exemplo, a rede é composta por três *containers* principais: um cliente, um servidor, simulando um servidor vulnerável, e uma "gateway" que atua como router para medir o tráfego entre sub-redes. Estes componentes são ligados por uma rede virtual LAN1 pelo IP 192.168.1.0/24, onde o cliente e o servidor comunicam diretamente, enquanto a *gateway* pode ser configurada para filtrar ou monitorizar o tráfego, simulando cenários reais como rede interna. A topologia é definida no ficheiro `start.config`, que especifica os *containers*, as suas interfaces de rede e configurações como IP estáticos, DNS e rotas, garantindo isolamento do *host* e replicação. Por exemplo, o *container* "client"pode ter

acesso à internet via gateway, enquanto o "server" é restrito à LAN1, permitindo exercícios como *port scanning* com Nmap ou análise de pacotes com Wireshark sem afetar os sistemas externos. Esta configuração é escalável, suportando adição de mais *containers* como um IDS, e é ideal para formação remota, pois executa localmente no computador do aluno com recursos mínimos, promovendo as competências práticas em cibersegurança [17].

Com o desenvolvimento dos modelos de linguagem de grande dimensão e algoritmos avançados, a inteligência artificial tem vindo a ser incorporada no ensino prático da cibersegurança, complementando abordagens como os ambientes simulados e laboratórios virtuais já discutidos [6]. Esta integração inclui a deteção automática de vulnerabilidades em códigos, a criação adaptativa de desafios personalizados baseados no progresso do aluno e o fornecimento de feedback imediato de forma inteligente [15][16][24]. Tais funcionalidades permitem que cada estudante avance ao seu próprio ritmo, receba orientações e enfrente desafios alinhados às suas competências, aumentando significativamente a eficácia e a retenção no processo de aprendizagem. Esta evolução tecnológica sugere um potencial para melhorar *frameworks* como os Labtainers, que, embora robustos em segurança e replicabilidade, carecem de elementos motivacionais nativos [19], abrindo caminho para a análise de exemplos adicionais na literatura que exploram estas inovações.

De seguida, detalha-se o que já existe em termos de plataformas para ensino prático em cibersegurança, destacando algumas das principais soluções e tecnologias utilizadas, com base em análises recentes da literatura que enfatizam a gamificação, a escalabilidade e a simulação realista para mitigar a escassez de competências [6].

O Hack The Box é uma plataforma dominante na formação prática em cibersegurança, oferecendo um conjunto variado de máquinas virtuais vulneráveis onde os utilizadores praticam técnicas de *pen testing* e exploração de falhas para resolver desafios reais. Suporta mais de 500.000 utilizadores ativos, com exercícios baseados em cenários MITRE ATT&CK que promovem competências em *hacking* ético, análise de *malware* e resposta a incidentes [25]. Do ponto de vista tecnológico, funciona numa infraestrutura em nuvem que utiliza virtualização para criar máquinas virtuais isoladas, garantindo que os ataques e exploração não afetem sistemas reais. O acesso é controlado via VPN e a plataforma incorpora elementos gamificados como ranking, pontuação e certificação de competências. O modelo permite escalabilidade e suporta milhares de utilizadores em

simultâneo, com integração de IA para desafios adaptativos que ajustam a dificuldade ao progresso do utilizador [26].

A arquitetura do Hack The Box, como demonstra a figura seguinte 2.4, é modular e *cloud-native*, centrada em AWS para fornecimento dinâmico, a camada de infraestrutura usa VMs (EC2) e *containers* (Kubernetes) para alojar máquinas vulneráveis isoladas via namespaces, com redes *overlay* para simulações de redes empresariais. A camada de gestão, gere desafios via APIs REST e *scripts* Python para criar *flags* dinâmicas. A camada de aplicação inclui interfaces web para quadros de pontuação e percursos de aprendizagem, e uma camada transversal de análise que utiliza aprendizagem automática para métricas de desempenho e relatórios GIAC, suportando eventos globais como CTF com milhares de participantes [27]. Esta estrutura facilita a expansão para labs colaborativos, como o "Orion Lab" para cenários empresariais complexos [28].

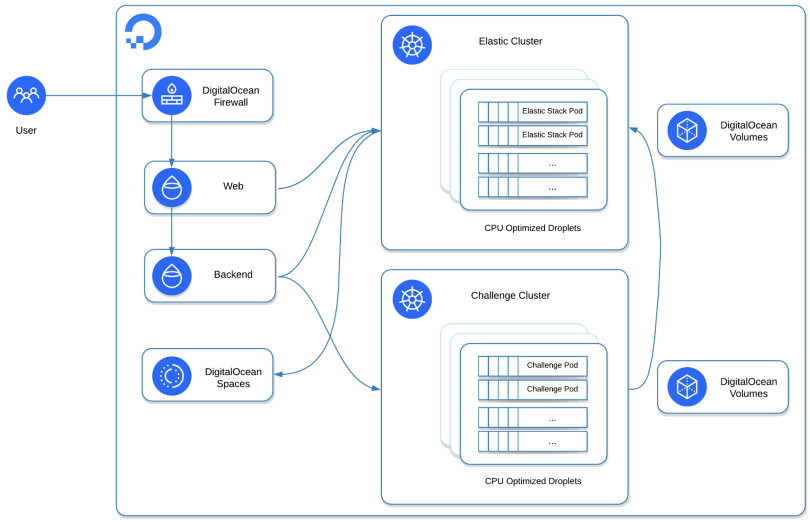

Figura 2.4: Diagrama da Arquitetura do Hack The Box. Retirado de [29]

Destinado sobretudo a jovens estudantes e iniciantes, o PicoCTF oferece uma interface simples e acessível com desafios do tipo CTF baseados em plataformas web. Com mais de 36.000 utilizadores anuais, a plataforma privilegia a acessibilidade e a simplicidade, utilizando ambientes sandbox [1] para garantir segurança. Os desafios são progressivos e abrangem várias áreas técnicas, promovendo uma aprendizagem gradual. A gamificação integra rankings e recompensas que motivam a participação e o progresso dos alunos, com foco em conceitos básicos como criptografia, forense digital e programação segura [12]. Estudos mostram que o PicoCTF melhora a retenção de conhecimentos em 70-80%

[1] ambiente de testes isolado que permite executar e testar software

em estudantes do ensino médio, preparando-os para competições avançadas [30]-[12].

A arquitetura do picoCTF é *open-source* e escalável, baseada num modelo *Jeopardy-style* que pode ser hospedado em diferentes ambientes *cloud*, como a amazon virtual services (AWS), google cloud platform (GCP) ou *Azure*, e também em servidores locais. A infraestrutura utiliza *containers* Docker para manter os desafios isolados e máquinas virtuais leves para simulações web. A camada de lógica gere os desafios através de *frameworks* CTF, como o FBCTF, com *scripts* em Python responsáveis pela validação de *flags* e pela atribuição de pontuação dinâmica. A camada de aplicação fornece interfaces web simples para registo e submissões, acessíveis por *browser*, enquanto uma camada de monitorização usa bases de dados MySQL para *rankings* e análise de participação, suportando eventos com milhares de utilizadores [31]. Esta configuração é portátil e permite a implementação em diferentes infraestruturas, tornando possível a reutilização de desafios em contextos de ensino.

Já o TryHackMe trata-se de uma plataforma similar que combina laboratórios virtuais com elementos gamificados avançados. Baseia-se em *cloud computing* para garantir acessibilidade remota, escalabilidade e segurança. A infraestrutura *cloud* disponibiliza os recursos informáticos via internet, eliminando a necessidade de *hardware* local, permitindo que milhares de utilizadores acedam simultaneamente a ambientes complexos e seguros. Estes sistemas destacam-se pela oferta de tutoriais guiados, feedback automático e componentes sociais que facilitam a aprendizagem colaborativa, cobrindo tópicos como *hacking* ético, resposta a incidentes e segurança na nuvem, com mais de 800 labs e 36.000 utilizadores mensais [27]. A plataforma melhora a retenção de competências em 70%, conforme estudos de 2024 [32].

A arquitetura do TryHackMe 2.5 é *cloud-centric* e modular, utilizando AWS para disponibilizar as máquinas, a camada de infraestrutura contém VMs isoladas (amazon elastic compute (EC2)) e *containers* para “salas” virtuais, com redes virtual private cloud (VPC) para sandbox seguros. A camada de gestão gere caminhos de aprendizagem via APIs e *scripts* Python para desafios adaptativos. Já a camada de aplicação inclui interfaces web para tutoriais e *scoreboards* gamificados, e uma camada de análise transversal usa machine learning (ML) para feedback e relatórios de progresso, suportando colaboração em tempo real com *WebSocket* [33]. Esta estrutura é usada para labs remotos, como o "Secure Network Architecture", com isolamento para exercícios de equipa.

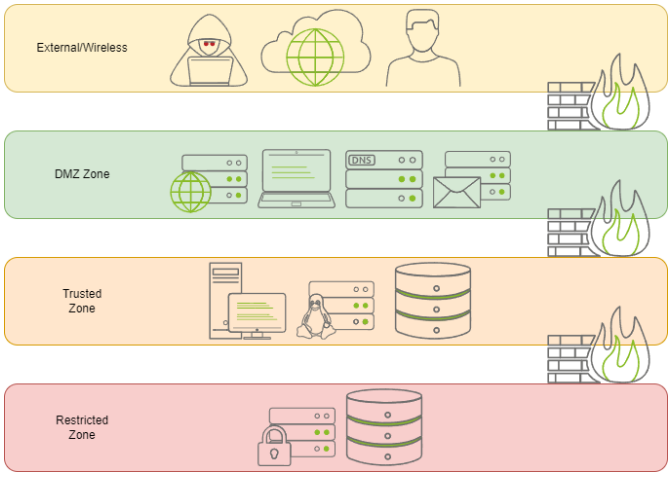

Figura 2.5: Diagrama da Arquitetura do TryHackMe. Adaptado de [34]

Estes exemplos ilustram abordagens variadas, mas a literatura também explora plataformas inovadoras que integram tecnologias emergentes como a inteligência artificial e simulações especializadas. Uma destas é o *Cyber Range Lite*, uma infraestrutura leve e portátil para treino em cibersegurança, projetada para baixo custo e fácil implementação. Utiliza virtualização para simular cenários reais, como exercícios CTF ou defesas contra ataques, sendo ideal para instituições com recursos limitados. A sua arquitetura modular inclui camadas de infraestrutura, como *hosts* virtuais e redes *overlay*, gestão de cenários e interfaces de utilizador (monitorização de métricas), garantindo portabilidade e escalabilidade [17].

O diagrama arquitetural apresentado na imagem 2.6 é uma representação em camadas de um *cyber range*, mostrando níveis como infraestrutura (física/virtual com *hosts* e redes *overlay*), gestão de cenários (com simulações e exercícios CTF), e interfaces de utilizador (análise de métricas e monitorização em tempo real). Os blocos ligados ilustram VMs isoladas, ferramentas de ataque/defesa e escalabilidade para múltiplos utilizadores, realçando portabilidade sem hardware dedicado. Isso explica a acessibilidade para instituições limitadas, com fluxos de dados de entrada (configuração) para saída (avaliação de desempenho).

Em contraste, o esCAPE representa uma abordagem gamificada que simula incidentes de cibersegurança através de métricas de "escape rooms" virtuais, onde os alunos resolvem puzzles interativos para "escapar" de cenários de risco, integrando feedback em tempo real e elementos colaborativos para treinar respostas a ameaças. Esta ferramenta, direcionada para o desenvolvimento de competências práticas, utiliza tecnologia para

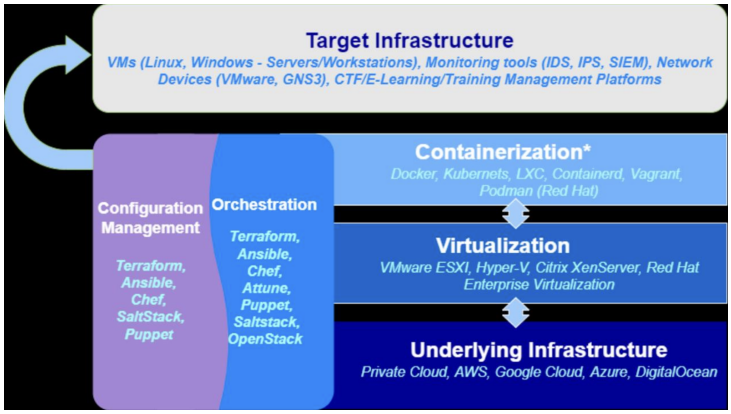

Figura 2.6: Diagrama de um Ciber Range Lite. Retirado de [7]

adaptar desafios ao nível do utilizador, promovendo a motivação através de narrativas envolventes e avaliação automática, com uma arquitetura que inclui camadas de acesso seguro, deteção de vulnerabilidades e integração com ferramentas DevSecOps para ambientes educativos. Mais especificamente, a plataforma incorpora módulos de gamificação como níveis progressivos e recompensas, com suporte a equipas, e a sua arquitetura é ilustrada em diagramas que mostram um fluxo de interação: entrada do utilizador via interface web, processamento de cenários em *cloud* e saída com relatórios de desempenho, incluindo camadas de segurança como multi-factor authentication (MFA) e intrusion detection system (IDS), o que a torna adequada para treino em incidentes reais sem riscos [35].

Na imagem 2.7 temos, em diagrama, o fluxo de interação em camadas, com a entrada do utilizador via interface web (puzzles), processamento em *cloud* (deteção de vulnerabilidades e adaptação de desafios) e saída com relatórios (feedback e pontuação). Inclui camadas de segurança como MFA, IDS e integração DevSecOps, com blocos para níveis progressivos e recompensas. Representa um modelo cliente-servidor com foco em *escape rooms* virtuais para treino de respostas a incidentes.

O CyberSim Lab, por sua vez, é um laboratório simulado que mistura a educação académica com o ambiente profissional. Oferece exercícios *hands-on* baseados em ataques reais e mostra a progressão de habilidades através de cenários virtuais gamificados. Esta plataforma facilita a transição para o mercado de trabalho ao incorporar feedback imediato e colaboração. Com uma infraestrutura baseada em simulações realistas que incluem mapas de ameaças e modeladores de riscos, permitindo a visualização de vulnerabilidades

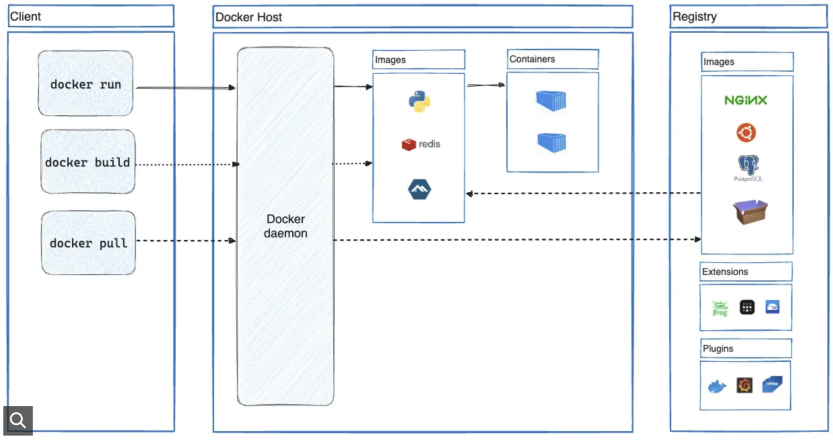

Figura 2.7: Diagrama de um esCape. Retirado de [36]

e a prática de defesas em redes complexas. Adicionalmente, integra elementos como simulações de ciberataques em equipa e avaliação de competências comportamentais. Esta arquitetura enfatiza a integração de ferramentas *open-source* para criação de labs virtuais. O diagrama arquitetural típico apresenta camadas como *frontend* para interação do aluno, *backend* com simulações baseadas em Docker ou VMs, e um módulo de análise que rastreia progresso e fornece relatórios, incluindo visualizações de caminhos de ataque (como nós colapsáveis em gráficos de rede) para identificar pontos fracos em tempo real [37].

O diagrama que se segue (2.8) mostra camadas como frontend (interação do aluno com mapas de ameaças), backend (simulações Docker/VMs para ataques reais), e análise (rastreamento de progresso com relatórios). Inclui nós colapsáveis em gráficos de rede para visualização de vulnerabilidades, fluxos de ciberataques em equipa e modeladores de riscos, enfatizando integração *open-source* para labs virtuais.

Já o SENSAI surge como um tutor baseado em large language models (LLM)s, atuando como um assistente inteligente para desafios CTF-like, extraindo contexto do trabalho do aluno (como terminais e ficheiros) para fornecer orientação personalizada, adaptando-se ao progresso individual em ambientes Linux. A sua arquitetura, representada na figura 2.9, envolve fluxos de extração de dados via ferramentas como eBPF, integração com LLMs como GPT-4 e interfaces web para interação contínua, tornando-a eficaz para uma educação personalizada. Para além disso, o sistema suporta cenários de tutoriais em múltiplos domínios, como análise de código e *debug*, com uma arquitetura detalhada em diagramas que mostram um fluxo em passos: captura de contexto do ambiente do aluno, processamento pelo LLM com fases de *thinking* e *response*, e feedback

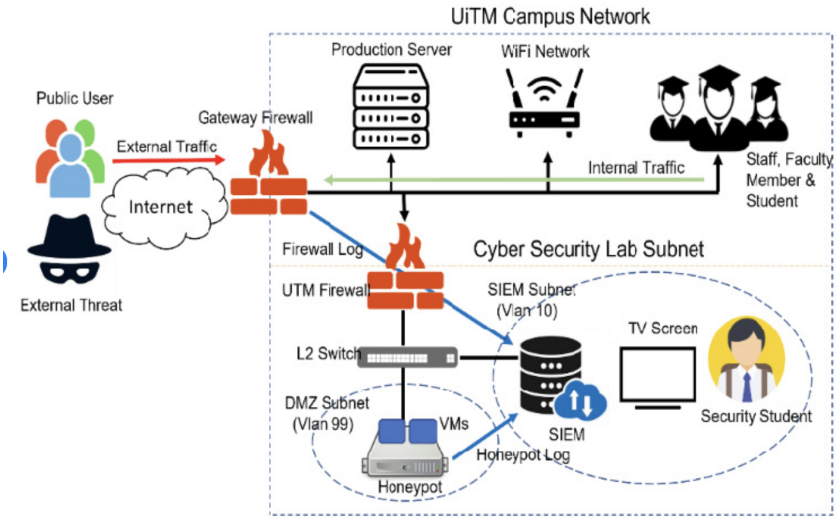

Figura 2.8: Diagrama de um CyberSim Lab. Retirado de [38]

interativo via interface, incluindo componentes como *prompts* de sistema para guiar o tutor e integração com ferramentas de monitorização para avaliação automática de soluções [39].

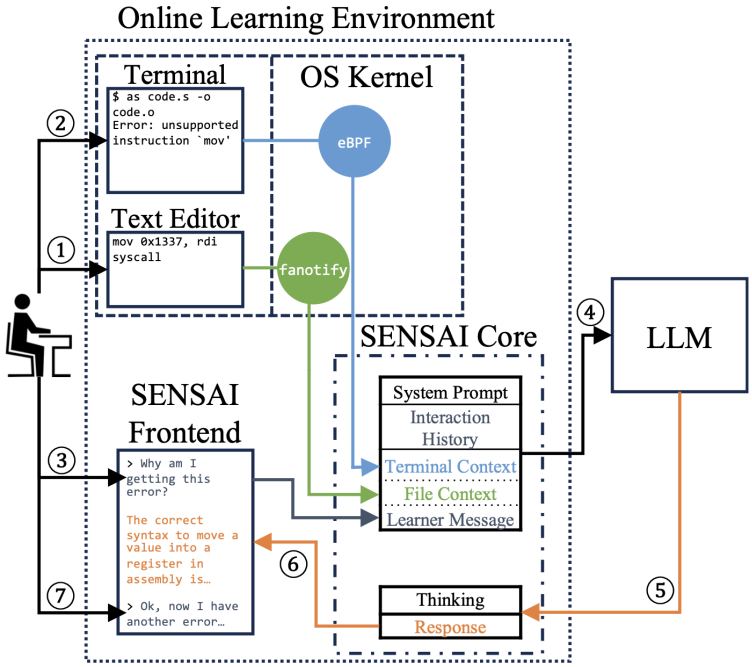

Figura 2.9: Diagrama do SENSAI. Retirado de [40]

O KINAITICS é uma plataforma educativa que funde ferramentas de IA com gamificação para explorar e defender contra ataques *cyber-kinetic*, melhorando a robustez

e resiliência através de exercícios adaptativos e avaliações automáticas, incluindo hackathons [2] para treino prático. Projetada no âmbito de projetos europeus como Horizon Europe, a sua arquitetura em camadas abrange infraestrutura distribuída, computação IA e aplicações de defesa, permitindo desafios personalizados que se ajustam ao nível do utilizador e promovendo competências em cenários reais. Adicionalmente, incorpora módulos para simulação de ameaças híbridas (físicas e cibernéticas), com diagramas arquiteturais que ilustram quatro camadas principais: a base de infraestrutura com sensores e internet of things (IoT), a camada de computação para processamento IA, a plataforma de ferramentas virtuais e camada de aplicação para interfaces gamificadas, incluindo ligações para análise de riscos e adaptação dinâmica de desafios baseados em desempenho do utilizador [41].

A seguinte imagem (2.10) apresenta o diagrama com essas quatro camadas: infraestrutura (sensores/IoT), computação (processamento IA), plataforma (ferramentas virtuais), e aplicação (defesa gamificada). Os blocos interligados ilustram a simulação de ameaças *cyber-kinetic*, hackathons e adaptação dinâmica, com foco em resiliência via IA.

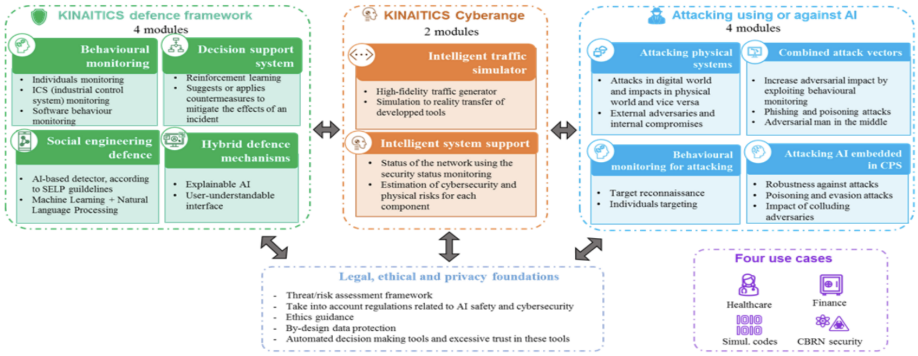

Figura 2.10: Diagrama do KINAITICS. Retirado de [42]

A plataforma CYBERWISER.eu é um ambiente virtual inovador concebido para a formação em cibersegurança, com particular ênfase em sistemas industriais, como aqueles que integram tecnologias SCADA e infraestruturas de Tecnologias da Informação (TI). Desenvolvida no âmbito do projeto europeu financiado pelo programa Horizon 2020, conforme descrito no artigo, esta plataforma tem como objetivo capacitar profissionais de cibersegurança através de simulações realistas, permitindo que equipas, oriundas de organizações públicas ou privadas, desenvolvam competências para proteger sistemas críticos

[2] evento colaborativo intensivo, onde equipas trabalham em ritmo acelerado para criar soluções tecnológicas, como protótipos ou produtos, num curto espaço de tempo

contra ameaças cibernéticas. Trata-se de um *cyber range*, ou seja, um ambiente virtual multifuncional que integra pessoas, processos e tecnologias, possibilitando a realização de exercícios de formação, testes de segurança e testes unitários em sistemas industriais, como plantas de controlo. O artigo destaca a relevância desta iniciativa face à escassez de profissionais qualificados em cibersegurança, evidenciada por dados de 2019 que indicam que 65% das organizações enfrentavam carências de pessoal especializado e 51% consideravam os seus sistemas em risco moderado ou extremo devido a esta lacuna. Assim, a CYBERWISER.eu promove a democratização da formação em cibersegurança, oferecendo cenários personalizados que simulam ataques e defesas, monitorizam o progresso dos formandos em tempo real e avaliam as vulnerabilidades de forma sistemática [43].

Na próxima figura (2.11) temos a arquitetura desta plataforma que se caracteriza como um ambiente educativo e baseado na web, projetado para facilitar a adoção, o suporte colaborativo e atualizações contínuas. Construída com base nos resultados do projeto [44], a plataforma assenta numa infraestrutura de máquinas físicas que suportam cenários de formação virtualizados, escaláveis e configuráveis, capazes de representar infraestruturas de TI industriais, como sistemas SCADA [3] de forma totalmente ou parcialmente virtualizada, integrando, quando necessário, dispositivos físicos [45]. A criação de cenários é suportada por bibliotecas de recursos computacionais, como máquinas virtuais, redes e software, que são combinadas por um gerador de cenários para configurar ambientes personalizados de forma célere e reconfigurável.

No que diz respeito à simulação de cenários dinâmicos de ataque e defesa, representada na figura 2.12, a plataforma incorpora um simulador de ataques, que permite a equipas *red team* executar ações ofensivas, e um simulador de contramedidas, que propõe ações de mitigação com base na evolução dos ataques, apoiando a tomada de decisão em tempo real. Sensores de monitorização são automaticamente implementados para recolher dados sobre o desempenho dos formandos e identificar vulnerabilidades nos sistemas industriais. Estes dados são processados por um detetor de anomalias, que filtra e correlaciona informações para identificar eventos críticos, como tentativas de intrusão. Adicionalmente, ferramentas de avaliação de vulnerabilidades, incluindo w3af, OWASP ZAP e Nmap, são integradas para detetar fraquezas exploráveis nos sistemas, alinhando-se com as recomendações para formação em ambientes ICS [47].

[3]sistema de controlo industrial que utiliza hardware e software para monitorizar, controlar e guardar dados em tempo real de processos e equipamentos industriais

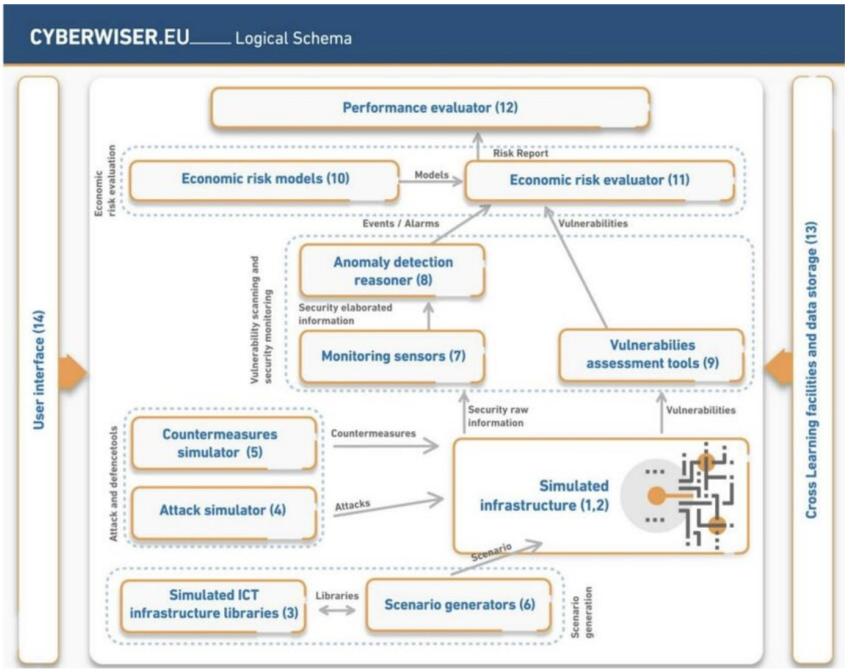

Figura 2.11: Diagrama do CiberWiser. Retirado de [46]

A quantificação de riscos é realizada por meio de modelos económicos inovadores e ferramentas de avaliação de risco, que estimam perdas potenciais com base em eventos e vulnerabilidades identificados. O desempenho dos formandos é avaliado por um componente específico, que utiliza estas métricas de risco e diretrizes predefinidas para cada cenário. Recursos de aprendizagem cruzada promovem a colaboração entre utilizadores, enquanto uma interface gráfica integrada facilita a interação, permitindo a configuração, monitorização e execução dos exercícios de formação.

O caso de estudo "Water Tank" [48] exemplifica esta capacidade, ao virtualizar uma planta industrial com sistemas SCADA e PLC, onde equipas *blue team* treinam a defesa contra ataques simulados por equipas *red team* usando Kali Linux. Este cenário demonstra a viabilidade de integrar formação com testes de segurança, abordando a complexidade de infraestruturas sem disrupção operacional [49].

Similarmente, o KYPO4INDUSTRY é um *testbed open-source* para ensino de cibersegurança em sistemas de controlo industrial, baseado em *cloud* e OpenStack, permitindo simulações de ameaças em ambientes industriais com monitorização em tempo real e análise da aprendizagem. A sua estrutura em camadas facilita a criação de labs isolados para treino universitário e profissional, enfatizando a recuperação rápida e a adaptação a cenários complexos como SCADA. A plataforma suporta extensões para gamificação e colaboração em equipa. Com a arquitetura ilustrada em diagramas (2.13) que destacam

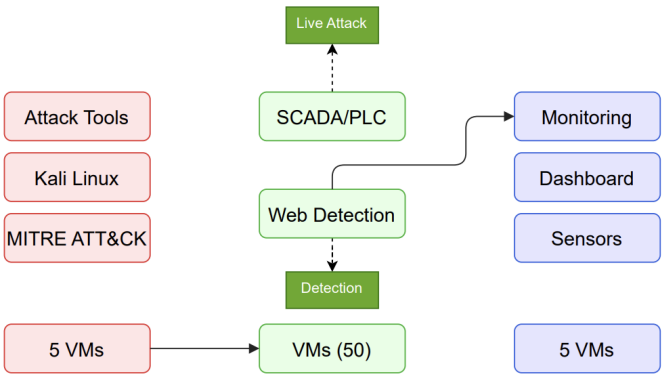

Figura 2.12: Diagrama de uma simulação de ataque e defesa CiberWise. Adaptado de [46]

camadas de infraestrutura (*containers* e *microservices*), interface de utilizador para configuração de cenários, e ferramentas de análise para métricas de desempenho, incluindo simulações de redes industriais com recuperação automática para repetições educativas [50].

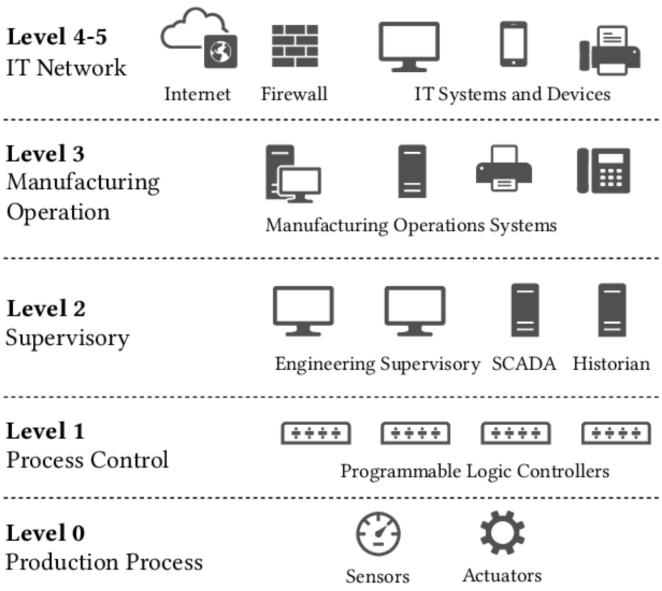

Figura 2.13: Diagrama do KYPO4INDUSTRY. Retirado de [51]

O CRACK, uma framework para construir *cyber ranges*, automatiza o design, faz a verificação e geração de cenários CTF dinâmicos, utilizando linguagens como TOSCA e SDL para verificação formal e implementação. A sua arquitetura (2.14) é em fluxo cascata e inclui etapas de especificação, verificação de propriedades e teste, tornando-a modular e escalável para exercícios educativos que alinhem objetivos de treino com simulações reais. Adicionalmente, permite a integração de métricas de avaliação com diagramas

que representam um modelo em cascata, ou seja, blocos sequenciais para o design dos cenários, verificação lógica, geração automática de ambientes virtuais e alinhamento com objetivos pedagógicos, incluindo componentes para teste de consistência e implementação em *cloud* para suporte a múltiplos utilizadores [52].

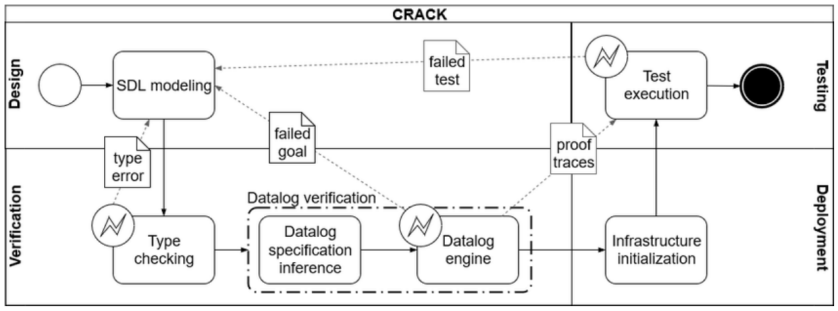

Figura 2.14: Diagrama do CRACK. Retirado de [53]

A CyberChallenge.IT é uma iniciativa nacional italiana de *ethical hacking* direcionada a jovens talentos, oferecendo testes, lições e competições gamificadas em tópicos como criptografia e segurança de rede, promovendo o desenvolvimento precoce de competências através de CTFs acessíveis. A sua arquitetura (2.15) suporta desafios progressivos com monitorização de desempenho, integrando elementos sociais para motivação e colaboração. A plataforma inclui fases de seleção e treino, com diagramas arquiteturais que mostram fluxos de interação, tais como, inscrição via web, módulos de desafios CTF com pontuação automática e análise de resultados para rankings, incluindo camadas de segurança para ambientes sandbox e integração com comunidades educativas [54].

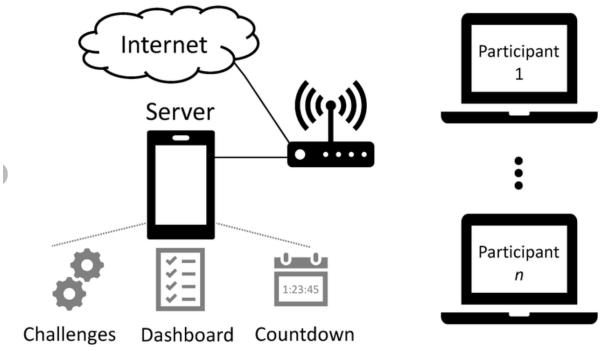

Figura 2.15: Diagrama do CiberChallenge. Retirado de [55]

O CyFRS é um sistema de *ciber range* rápido e recuperável baseado em redes reais, permitindo simulações práticas com recuperação automática após exercícios, ideal para repetições em cenários educativos. A sua estrutura (2.16) foca-se em fluxos de rede

isolados, facilitando a prática segura de ataques e defesas sem impactos permanentes. Dá-se ainda ênfase em eficiência, pois a arquitetura inclui mecanismos de *snapshot* para recuperação, diagramas para mostrar camadas da simulação de rede com pontos de recuperação, módulos para a configuração de cenários e análise pós-exercício, permitindo escalabilidade em ambientes académicos [56].

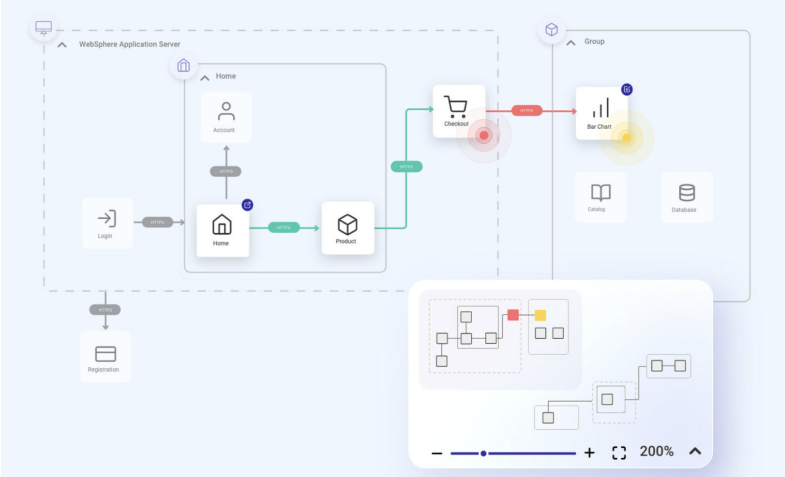

Figura 2.16: Diagrama do CyFRS. Retirado de [57]

A Comprehensive Cyber Arena representa um *ciber range* para treino abrangente, simulando arenas cibernéticas com *ripple effects* e dependências entre domínios, permitindo exercícios de equipa em ambientes isolados. A sua arquitetura (2.17) inclui infraestrutura física/virtual, simulação de internet e métricas de análise, promovendo a coordenação e o raciocínio crítico em situações complexas. A plataforma suporta simulações multi-domínio, com diagramas que ilustram ranges isolados para organizações, incluindo componentes como a simulação de *ripple effects*, dependências interligadas e ferramentas para avaliação da equipa, com foco em realismo para o treino avançado [58].

O RangeForce é uma plataforma de *ciber range* baseada em *cloud*, projetada para treinar equipas de segurança cibernética (SOC) em cenários realistas de ameaças, com ênfase em *upskilling* e exercícios de equipa para resposta a incidentes. Com integração de IA para simulações avançadas, permite a criação rápida de ranges em horas, utilizando catálogos de falhas de segurança reais, como *ransomware*, APIs, para praticar defesas, deteção e recuperação. Destaca-se pela modularidade, suportando treino individual ("Solo Labs") ou coletivo, com gamificação através de missões progressivas e métricas de desempenho que simulam ambientes operacionais, reduzindo o tempo de preparação para ameaças reais até 50% conforme estudos corporativos [60]. É particularmente útil para

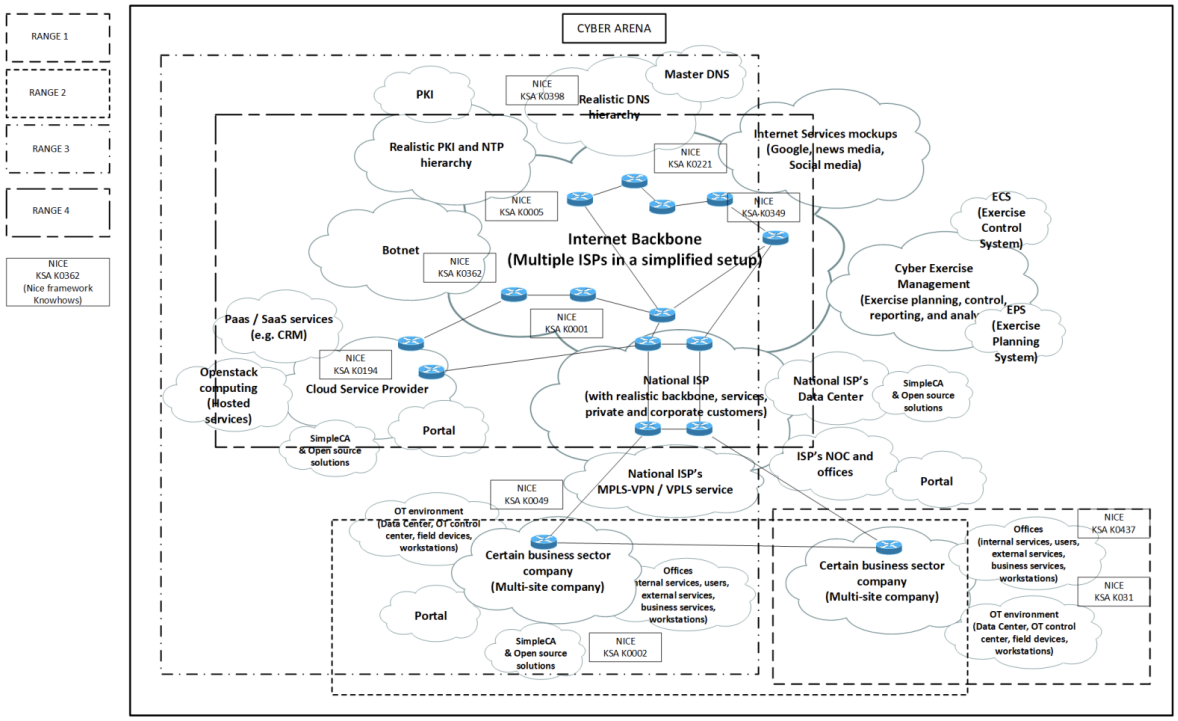

Figura 2.17: Diagrama do Comprehensive Cyber. Retirado de [59]

organizações Fortune 500, promovendo competências em *threat hunting* e conformidade regulatória.

A arquitetura do RangeForce (2.18) segue um modelo em camadas escalável, centrado em *cloud* (AWS ou Azure) em que a camada de infraestrutura utiliza VMs isoladas e *containers* para guardar simulações de redes e aplicações vulneráveis, garantindo isolamento via namespaces Docker. A camada de gestão integra IA para gerar cenários adaptativos baseados em catálogos pré-configurados, com APIs REST para gestão de exercícios. A camada de aplicação inclui interfaces web para configuração de missões e *scoreboards* gamificados, e uma camada transversal de análise com ML para feedback em tempo real com relatórios de resiliência, suportando milhares de utilizadores simultâneos sem *overhead* local [61]. Esta estrutura modular facilita a expansão para exercícios de equipa, com recuperação automática pós-simulação.

O SimSpace é uma plataforma de *cyber range* de alta fidelidade desenvolvida para quantificar riscos operacionais, simulando "what-if" cenários com emulação de adversários e utilizadores reais, focando-se em defesa cibernética para organizações governamentais e privadas. Com suporte a Orientação Tutorial (OT) e cyber physical system (CPS), permite testes de resiliência contra ameaças como distributed denial-of-service (DDOs)

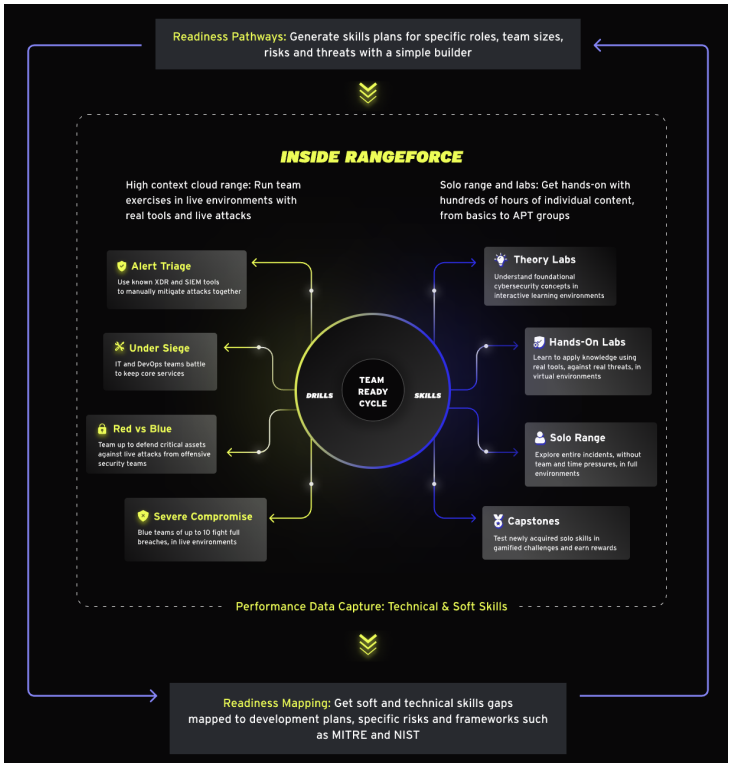

Figura 2.18: Diagrama de um RangeForce. Retirado de [62]

ou ransomware, com ênfase em redução de incertezas através de simulações que replicam infraestruturas críticas. Reconhecida pela Gartner como líder em plataformas para segurança OT, promove wargames colaborativos que melhoram a coordenação de equipas em 40-60%, conforme relatórios de 2024 [63].

Os SANS Cyber Ranges, particularmente a suíte NetWars, constituem uma plataforma de treino *hands-on* desenvolvida pelo SANS Institute para todos os níveis de habilidade em cibersegurança, com narrativas envolventes que simulam cenários reais. Oferece mais de 460.000 aulas cobrindo tópicos como *hacking* ético, resposta a incidentes e conformidade GIAC, através de desafios multifacetados que integram gamificação (scorecards, torneios) e acesso 24/7 remoto. Destaca-se pela abrangência multidisciplinar, preparando utilizadores para certificações, diminuindo lacunas de competências em 70% nos programas corporativos [64].

Outro exemplo é o CloudShare, uma plataforma de laboratórios virtuais para treino em cibersegurança, oferecendo catálogos de cenários pré-configurados, como análise de *malware* e APIs, para desenvolvimento de competências em equipas de segurança, com foco em simulações a pedido e integração com ferramentas como o Splunk. Permite a criação de ambientes isolados (*sandbox*) seguros para prática de defesas, com adaptação

via IA e painéis de controlo que medem a retenção de competências em 70% [65]. É ideal para programas de conformidade e demonstrações, suportando acesso remoto sem necessidade de hardware local.

A arquitetura do CloudShare é *cloud-native* e escalável, contém uma camada de infraestrutura AWS para VMs e *containers*, criando ambientes isolados com aprovisionamento automático. A camada de catálogo gere os cenários pré-definidos via APIs REST, com gestão para simulações personalizadas. A camada de aplicação fornece interfaces web para configuração e quizzes interativos, e uma camada de análise que usa ML para métricas e relatórios de risco, integrando com SIEM para feedback em tempo real, suportando milhares de utilizadores simultâneos [66]. Esta configuração prioriza acessibilidade e integração, diferenciando-se pelo seu extenso catálogo para treinos rápidos.

2.2 Gamificação no Ensino e Treino de Cibersegurança

A combinação de IA e gamificação tem vindo a transformar o ensino e treino em cibersegurança, oferecendo abordagens mais dinâmicas, personalizadas e eficazes. Tradicionalmente, os métodos de aprendizagem prática, como os *cyber ranges* e os CTFs, forneciam experiências mais aprofundadas, mas com limitações em termos de adaptação ao nível do utilizador e escalabilidade. Com o avanço da IA, essas plataformas começaram a incorporar algoritmos capazes de ajustar a dificuldade dos desafios, fornecer feedback imediato e identificar padrões de desempenho que permitem personalizar a aprendizagem [15], [16].

A inteligência artificial no contexto educativo atua sobretudo em três dimensões: (1) personalização, adaptando conteúdos e desafios ao nível do aluno; (2) avaliação inteligente, que analisa interações e erros para fornecer feedback em tempo real; e (3) automação, reduzindo o trabalho manual de instrutores e facilitando a criação de cenários. Estas funções são frequentemente implementadas através de técnicas de *machine learning*, análise preditiva e, mais recentemente, modelos de (LLMs), que simulam tutoriais interativos em contextos técnicos complexos [24].

Por outro lado, a gamificação continua a ser um dos fatores mais eficazes para promover o envolvimento e a motivação. Em ambientes de cibersegurança, recorre-se a

elementos como pontos, medalhas, níveis e *leaderboards* para transformar o treino técnico em experiências competitivas e motivadoras [10]. Esta abordagem não só aumenta a participação, como também melhora a retenção de conhecimento, reduzindo a monotonia de atividades repetitivas e incentivando a aprendizagem contínua. Estudos recentes indicam que a aplicação combinada de gamificação e IA pode elevar a taxa de retenção de competências em 70–80% [67], [68].

A colaboração entre IA e gamificação está a originar uma nova geração de plataformas educativas capazes de adaptar desafios automaticamente, detetar níveis de stress ou desempenho e ajustar o conteúdo de acordo com o progresso do utilizador. Por exemplo, algoritmos de aprendizagem automática permitem que sistemas gamificados reconheçam quando um estudante está com dificuldade e ofereçam pistas ou desafios intermédios. Da mesma forma, modelos LLM como o GPT-4, quando integrados em plataformas como o SENSAI, atuam como tutores virtuais capazes de interpretar o contexto técnico do utilizador e fornecer orientação contextualizada [39].

Esta tendência também se reflete em iniciativas europeias recentes, como o KINAITICS, que integra IA para ajustar desafios de treino em tempo real com base em sensores IoT, ou em sistemas multimodais que utilizam métricas biométricas para avaliar a resposta emocional dos aprendentes e adaptar o nível de dificuldade [67], [69]. Tais abordagens alinham-se com a necessidade crescente de formação adaptativa e escalável, especialmente após a pandemia, quando o ensino remoto se tornou predominante e os modelos estáticos mostraram-se insuficientes.

De forma geral, verifica-se que a integração de IA e gamificação não substitui os métodos tradicionais, mas amplia a sua eficácia, tornando o ensino de cibersegurança mais interativo, acessível e ajustável às necessidades individuais.

A Tabela 2.1 resume as principais contribuições identificadas na literatura recente entre 2023 e 2025, destacando plataformas e estudos que aplicam estas tecnologias para melhorar o desempenho e a motivação dos aprendentes.

Estudo/Plataforma	**Descrição**	**Uso de IA**	**Elementos de Gamificação**	**Benefícios Interpretados**
Generative AI in Gamified Cybersecurity Education	Estudo que desenvolve labs gamificados para cursos de segurança de rede, usando IA para criar cenários e exercícios.	IA generativa para cenários realistas e exercícios auto-guiados, com revisão humana para mitigar erros.	Pontos de recompensa e leaderboards onde os estudantes mantêm uma aprendizagem contínua.	Reduz workload dos instrutores em 50-70%, facilitando treino remoto; interpreta-se como uma solução para escalar a educação formal [68].
Multimodal and Adaptive Gamified System	Sistema adaptativo para simuladores de cibersegurança, usando analytics para avaliar competências e soft skills.	IA (LOF) para deteção de stress via biometria e ajuste dinâmico de desafios.	Medalhas temáticas e rankings variáveis para promover competitividade e gestão de stress.	Aumenta competências em 70-80%, otimizando uma aprendizagem adaptativa; interpreta-se como avanço para treinos holísticos [67].
KINAITICS	Plataforma que usa IA para treinar contra ataques cyber-kinetic com hackathons adaptativos.	IA para análise de sensores/IoT e adaptação de desafios em tempo real.	Níveis progressivos e recompensas para resiliência contra ameaças híbridas.	Retenção de 75%, ideal para contextos europeus; ponte entre teoria e prática real [69].
SENSAI	Tutor IA para desafios CTF, oferece orientação socrática em ambientes Linux.	LLMs para extração de contexto e tutoriais personalizados via eBPF.	Medalhas por conquistas para motivar iterações contínuas.	Retenção de 70%, solução para falta de tutores; adapta-se ao trabalho remoto [70].

Tabela 2.1: Tabela Interpretativa sobre IA e Gamificação em Cibersegurança (2023-2025)

A análise destes estudos revela três tendências principais: (1) a IA como motor de personalização, reduzindo barreiras de *input* e adaptando o ritmo de aprendizagem; (2) a gamificação como mecanismo de motivação e retenção, associada a ganhos mensuráveis de até 80%; e (3) a integração de ambas como resposta à necessidade de formação escalável e contínua em cibersegurança. Apesar dos progressos, persistem desafios, nomeadamente a dependência excessiva de modelos preditivos e o risco de enviesamento nos sistemas de IA, o que reforça a importância da supervisão humana e da validação ética nos ambientes educativos.

2.3 Ferramentas/Tecnologias

2.3.1 Terraform

O Terraform, uma ferramenta de infraestrutura como código (infrastructure as code) (IaC) desenvolvida pela HashiCorp, permite a definição declarativa de recursos computacionais através da linguagem hashiCorp configuration language (HCL). No presente trabalho, o Terraform foi utilizado para gerir redes virtuais e máquinas pré-configuradas destinadas a cenários com CTFs e Labtainers, assegurando ambientes replicáveis e escaláveis. Por exemplo, foi possível criar uma rede virtual para um conjunto de máquinas, simulando um cenário típico de competição CTF [71]. A funcionalidade de controlo de versões do Terraform garantiu a consistência entre diferentes iterações do ambiente, facilitando testes, ajustes e reimplementações.

2.3.2 Proxmox

O Proxmox Virtual Environment é uma plataforma *open-source* que combina o *hipervisor* KVM com *containers* linux containers (LxC). No âmbito deste projeto, o Proxmox foi utilizado para alojar máquinas virtuais e templates pré-configurados, que simularam sistemas vulneráveis destinados a exercícios de cibersegurança, como CTFs e Labtainers. Por exemplo, foi implementado um template baseado em Debian, contendo vulnerabilidades conhecidas, num cenário de ataque por *brute force*, permitindo aos alunos praticar estratégias de defesa num ambiente controlado e isolado [72]. A interface web centralizada do Proxmox facilitou significativamente a gestão de múltiplos *containers* e máquinas

virtuais.

2.3.3 Ansible

O Ansible é uma ferramenta de automação que utiliza ligações secure shell (SSH) e *playbooks* escritos em YAML para configurar e gerir sistemas de forma eficiente. No presente trabalho, o Ansible foi utilizado para automatizar a preparação dos ambientes com Labtainers e cenários CTF, garantindo a instalação de ferramentas essenciais como instalar serviços básicos, como SSH, Python, atualização de serviços na máquina, como pacotes do sistema em máquinas virtuais previamente provisionadas com o Terraform.

2.3.4 Python

O Python é uma linguagem de programação versátil, amplamente utilizada no desenvolvimento de *scripts* e aplicações web personalizadas. No âmbito deste projeto, recorreram-se a *frameworks* como Flask e FastAPI para criar interfaces interativas destinadas a CTFs e Labtainers, nomeadamente um portal web onde os alunos podiam submeter as *flags* capturadas. Adicionalmente, foi desenvolvido um script em Python para automatizar a geração de desafios dinâmicos, ajustando os níveis de dificuldade com base no desempenho dos utilizadores [72]. A relevância do Python na comunidade de cibersegurança, aliada à sua facilidade de prototipagem, contribuiu para o alinhamento com os objetivos de acessibilidade e agilidade na criação da plataforma.

2.3.5 Bash

Os *scripts* Bash foram utilizados para automatizar tarefas operacionais no fluxo de trabalho, como a inicialização de ambientes e a preparação de ficheiros de configuração para cenários CTF. Por exemplo, um *script* Bash foi responsável por criar diretorias e instalar dependências necessárias para um ambiente Labtainer, completando o processo em menos de um minuto [71]. Embora tenham sido bastante úteis durante a fase de desenvolvimento, a legibilidade reduzida de *scripts* mais complexos revelou-se uma limitação para automações mais avançadas.

2.3.6 React

O React, uma biblioteca JavaScript, foi usado para criar interfaces dinâmicas e acessíveis. No projeto, desenvolveu-se um *dashboard* interativo que permitia aos alunos monitorizar o progresso nos vários desafios e aceder à criação dos mesmos, ou pedir o ficheiro da vpn para dar acesso à respetiva máquina. Componentes reutilizáveis e o Virtual DOM garantiram uma experiência fluida, atendendo às necessidades de acessibilidade na educação digital pós-pandemia [72].

2.3.7 FastAPI

O FastAPI é uma *framework* moderna em Python utilizada para o desenvolvimento de APIs RESTful rápidas e eficientes. No contexto deste projeto, o FastAPI foi adotado para gerir interações em tempo real na plataforma, permitindo funcionalidades como o registo automático de atividades dos utilizadores, a entrega dinâmica de desafios CTF e a validação imediata das *flags* submetidas pelos alunos.

2.4 Síntese

2.4.1 Síntese comparativa

A diversidade de abordagens refletida na Tabela 2.2 revela uma riqueza de estratégias no ensino de cibersegurança. Plataformas como o Cyber Range Lite e o CYBERWISER.eu, baseadas em virtualização e análise de riscos, focam-se no treino geral e em sistemas industriais, respectivamente, suportando até 5.000 e 6.000 utilizadores. Essas soluções destacam-se pela escalabilidade, mas enfrentam limitações como configuração manual e complexidade para iniciantes. Por outro lado, ferramentas como esCAPE e CyberChallenge.IT incorporam gamificação avançada, com escape rooms virtuais e competições CTF para jovens, embora com escalabilidade limitada (2.000 e 1.500 utilizadores). Plataformas como KINAITICS e SENSAI integram IA de forma inovadora, utilizando sensores IoT e LLMs para adaptação dinâmica e tutoriais personalizados, atendendo a profissionais avançados e aprendizes individuais com retenção de competências até 75% [69], [70]. Contudo, dependem fortemente de IA e têm foco específico. Soluções como CyberSim Lab e KYPO4INDUSTRY, baseadas em Docker e OpenStack, ligam a parte da academia

com a indústria, mas sofrem com escala global limitada e configuração técnica.

A análise evidencía que a gamificação varia de básica, como CYBERWISER, a avançada, como esCAPE, influenciando a motivação dos utilizadores. O uso de IA, presente em KINAITICS, SENSAI e CYBERWISER.eu, melhora a personalização, mas nem todas as plataformas o exploram plenamente. A escalabilidade, um fator crucial para o treino remoto, é desigual, com o CYBERWISER a liderar (6.000 utilizadores), enquanto CyberChallenge.IT fica atrás (1.500 utilizadores). O público-alvo abrange desde instituições limitadas até múltiplas equipas.

Apesar dos avanços, a comparação revela lacunas significativas. A gamificação básica em plataformas como Labtainers e CYBERWISER limita a motivação a longo prazo, como apontado por Thompson et al. [19], que destacam a ausência de elementos como rankings ou recompensas. As competições CTF, como CyberChallenge.IT, sofrem com desafios de inclusão devido à complexidade técnica, excluindo iniciantes e restringindo-se a nichos nacionais [14]. A falta de soluções para cenários híbridos (físicos e cibernéticos) é evidente, com apenas KINAITICS abordando essa área, mas com foco específico que não escala para públicos amplos. Além disso, a dependência de configurações manuais (Cyber Range Lite) ou de redes reais (CyFRS) compromete a acessibilidade e a reprodutibilidade em contextos remotos. Essas limitações refletem a dificuldade de equilibrar escalabilidade, e conseguir maior participação e inclusão, especialmente num cenário onde 65% das organizações enfrentam falta de competências [43].

Plataforma	**Tipo**	**Tecnologias**	**Foco**	**Gamificação**	**Escalabilidade**	**Público-Alvo**	**Limitações**
Cyber Range Lite	Cyber Range	Virtualização, Cloud	Treino Geral	Básica	5.000 utilizadores	Instituições Limitadas	Configuração Manual
esCAPE	Escape Rooms	Cloud, DevSecOps	Resposta a Incidentes	Avançada	2.000 utilizadores	Estudantes Multidisciplinares	Foco em Awareness
CyberSim Lab	Lab Simulado	Docker, Open-Source	Academia-Profissional	Avançada	3.000 utilizadores	Community Colleges	Escala Global Limitada
SENSAI	Tutor IA	LLMs, eBPF	Tutoria CTF	Básica	4.000 utilizadores	Aprendizes Individuais	Dependência de IA
KINAITICS	IA-Gamificada	IA, IoT	Ataques Cyber-Kinetic	Avançada	5.000 utilizadores	Profissionais Avançados	Foco Específico
CYBERWISER.eu	Cyber Range	VMs, Análise Riscos	Sistemas Industriais	Básica	6.000 utilizadores	PMEs, Energia/-Transporte	Complexidade Iniciante
KYPO4INDUSTRY	Testbed	Cloud, OpenStack	Controlo Industrial	Básica	4.000 utilizadores	Universitários, Profissionais	Configuração Técnica
CRACK	Framework	TOSCA/SDL	Cenários CTF	Básica	5.000 utilizadores	Desenvolvedores Ranges	Curva de Aprendizagem
CyberChallenge.IT	CTF	Sandbox	Ethical Hacking	Avançada	1.500 utilizadores	Jovens (16-24 anos)	Foco Nacional
CyFRS	Cyber Range	Redes Reais	Simulações Rápidas	Básica	3.000 utilizadores	Ambientes Acadêmicos	Dependência Redes
Comp. Cyber Arena	Cyber Range	Física/Virtual	Treino Multi-Domínio	Básica	4.000 utilizadores	Equipes Multidisciplinares	Alta Complexidade

Tabela 2.2: Comparação das plataformas de ensino em cibersegurança

2.4.2 Descrição do Problema

A formação prática em cibersegurança enfrenta vários desafios que limitam a eficácia dos métodos de ensino e a preparação dos futuros profissionais para o contexto digital atual. Por um lado, plataformas consolidadas como os Labtainers e os Seedlabs disponibilizam cenários técnicos bem estruturados, porém pouco dinâmicos, o que dificulta a simulação de situações reais e reduz o envolvimento dos utilizadores. Apesar de estas ferramentas possuírem sistemas automáticos de validação que funcionam como mecanismos de gamificação, muitos ambientes carecem de uma integração eficaz destes elementos lúdicos, limitando a motivação e progressão dos participantes conforme apontam estudos recentes [10].

Por outro lado, ambientes gamificados como PicoCTF, TryHackMe e HackTheBox promovem maior interesse e desenvolvimento de competências técnicas, beneficiando dos sistemas de pontuação, recompensas e feedback em tempo real. Contudo, enfrentam barreiras significativas em termos de personalização dos desafios, o que pode dificultar a adaptação a diferentes perfis e níveis de conhecimento dos utilizadores. Além disso, a gestão manual e a complexidade na configuração destes ambientes constituem um obstáculo importante, pois requerem recursos humanos e técnicos que muitas instituições, especialmente as de menor dimensão, não dispõem. Embora existam propostas inovadoras que integrem automação e tecnologias emergentes como a inteligência artificial e os grandes modelos de LLMs, a adoção eficaz destas ainda é limitada no contexto académico e profissional. Para além das limitações técnicas e pedagógicas, persistem preocupações relativas à escalabilidade, à segurança em ambientes partilhados e à insuficiência de mecanismos de feedback automático que permitam aos utilizadores aprender e corrigir erros em tempo real de forma eficaz.

Perante estas limitações, a plataforma proposta neste trabalho justifica-se pela necessidade de integrar laboratórios virtuais e desafios CTF num ambiente dinâmico, automatizado e gamificado, capaz de proporcionar personalização, escalabilidade e feedback instantâneo. Desta forma, será possível responder aos desafios contemporâneos da formação em cibersegurança, preparando profissionais resilientes e adaptados às exigências do mundo digital. No futuro, prevê-se a integração de componentes de IA para personalizar trajetórias de aprendizagem, adaptar o nível de dificuldade dos desafios e apoiar a deteção automática de padrões de desempenho dos participantes.

Capítulo 3

Plataforma Educativa de Cibersegurança

Com base nas limitações identificadas no capítulo anterior, este capítulo apresenta a proposta de uma plataforma educativa de cibersegurança concebida para superar as restrições de escalabilidade, automação e personalização observadas nas soluções analisadas. A proposta descreve a arquitetura conceptual da solução, os requisitos funcionais e não funcionais e o fluxo de automação que suporta a criação e gestão dos ambientes virtuais de aprendizagem. O objetivo é permitir a criação automatizada de desafios configuráveis, capazes de simular cenários reais de cibersegurança, oferecendo uma experiência dinâmica, acessível e segura, orientada para a prática e para a exploração de temas como injeção SQL, exploração de vulnerabilidades, segurança de redes e administração de sistemas.

3.1 Arquitetura Conceptual da Solução

A Figura 3.1 apresenta a arquitetura em camadas da solução, organizada de forma modular para garantir escalabilidade, automação e isolamento entre ambientes. A camada de interface de utilizador funciona como ponto de entrada para todos os utilizadores, que acedem à plataforma através de ligação segura à rede da instituição. Através desta interface é possível selecionar desafios, configurar parâmetros relevantes e acompanhar o progresso nas atividades. O acesso é facilitado por um ambiente intuitivo e orientado à usabilidade, concebido para apoiar a aprendizagem prática e autónoma. Logo, de seguida, situa-se a

camada de serviços e coordenação, responsável por gerir e validar os pedidos provenientes da interface, assegurando a coordenação entre os diversos componentes envolvidos no processo de criação dos ambientes virtuais. Esta camada processa as solicitações, valida as configurações e mantém o controlo sobre o ciclo de criação dos laboratórios. Na base encontra-se a camada de infraestrutura e automação, implementada sobre uma plataforma de virtualização que integra redes, máquinas virtuais, mecanismos de isolamento e serviços que permitem instanciar e configurar os ambientes de forma segura e escalável. As redes e máquinas desta camada operam isoladamente, garantindo que os utilizadores não interferem entre si. Toda esta estrutura modular assegura flexibilidade, segurança e eficiência, tornando possível adaptar a plataforma a diferentes cenários formativos e necessidades operacionais.

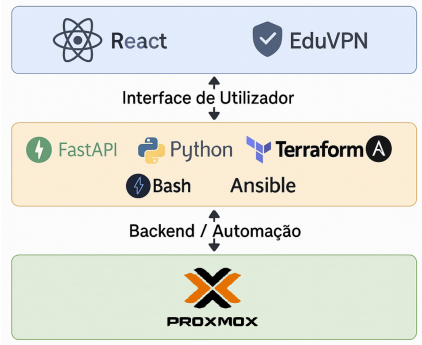

Figura 3.1: Arquitetura em camadas da solução proposta

3.2 Arquitetura Detalhada

A arquitetura detalhada da solução, apresentada na Figura 3.2, evidencia a forma como os diferentes componentes comunicam entre si. A camada de interface é acedida através de ligação segura à rede da instituição e constitui o espaço onde os utilizadores interagem com a plataforma. Todas as solicitações são recebidas pela camada intermédia, que coordena o fluxo de trabalho e assegura que os pedidos são válidos e consistentes. Esta camada gere o ciclo completo de criação dos laboratórios virtuais, estabelecendo a ligação entre a interface e os mecanismos automáticos responsáveis pela construção da infraestrutura. A camada de automação e virtualização, situada na base da arquitetura, cria as máquinas virtuais, configura as redes e instala automaticamente todas as dependências

necessárias para cada desafio. A combinação destas camadas resulta numa solução robusta e segura, capaz de disponibilizar ambientes práticos sem necessitar de intervenções manuais. O sistema foi concebido para garantir reprodutibilidade, isolamento entre ambientes e escalabilidade, permitindo a execução paralela de múltiplos desafios sem impactar o desempenho global.

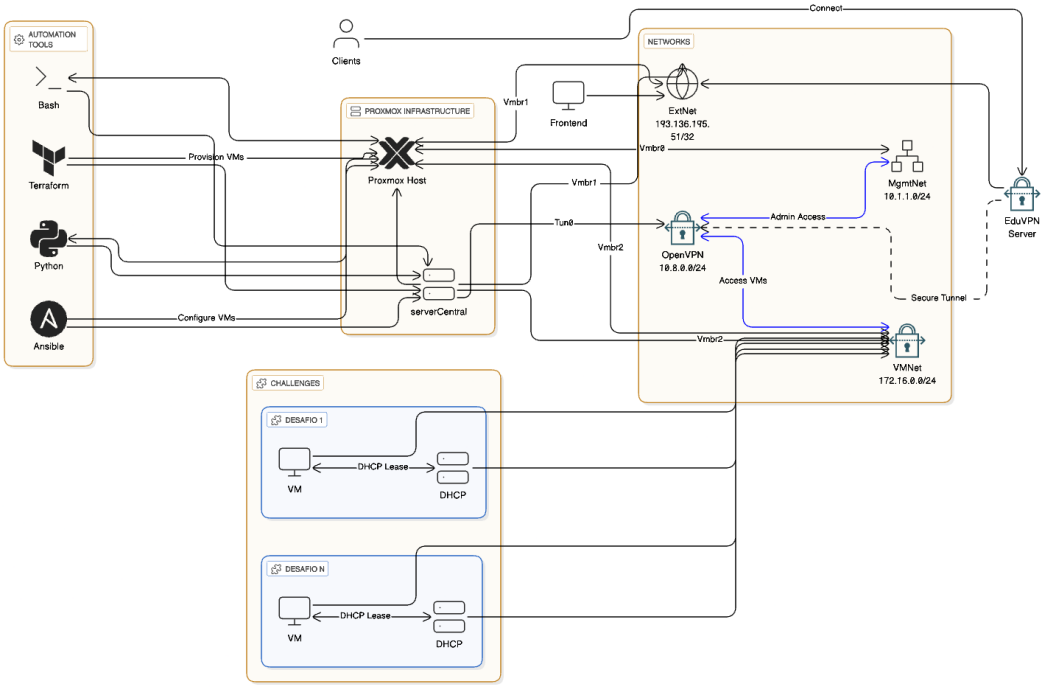

Figura 3.2: Diagrama da implementação da solução proposta

3.3 Requisitos Funcionais e Não Funcionais

Os requisitos funcionais da plataforma incluem a necessidade de permitir a seleção e execução de desafios práticos pelos utilizadores, bem como garantir que a criação e configuração dos ambientes é totalmente automatizada, reduzindo o esforço manual e acelerando o processo de disponibilização dos laboratórios. É igualmente essencial assegurar que os utilizadores recebem feedback sobre o estado dos seus ambientes, tanto durante o processo de preparação como após a sua conclusão. No que respeita aos requisitos não funcionais, a solução deve garantir isolamento seguro entre os diferentes ambientes, assegurando que os laboratórios são independentes e que não existe interferência entre

utilizadores. Deve ainda garantir ligações privadas e seguras, assim como uma arquitetura escalável que permita adicionar facilmente novos cenários e funcionalidades, sem necessidade de reconfigurar a estrutura principal.

3.4 Fluxo Conceptual de Criação dos Desafios

A Figura 3.3 apresenta o fluxo conceptual que ilustra o funcionamento do sistema desde o momento em que o utilizador seleciona um desafio até à disponibilização final do ambiente pronto a utilizar. O processo inicia-se quando o utilizador escolhe um desafio na interface, sendo o pedido encaminhado para a camada de serviços, onde é validado e preparado. Em seguida, esta camada solicita à infraestrutura de automação a criação do workspace correspondente ao desafio selecionado. Durante este processo são criadas as máquinas virtuais e configuradas as redes necessárias. Após a criação da infraestrutura, é aplicada automaticamente a configuração do ambiente, incluindo a instalação de aplicações e serviços necessários ao funcionamento do desafio. Quando o processo termina com sucesso, a camada de serviços informa a interface, que notifica o utilizador de que o laboratório está pronto para ser utilizado. Este fluxo garante consistência nas fases da criação dos ambientes.

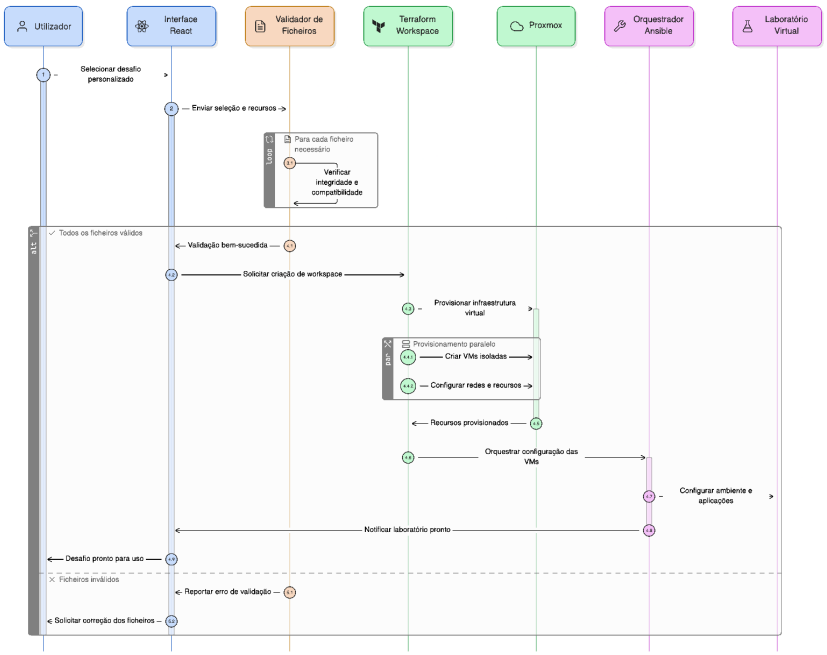

Figura 3.3: Diagrama da sequência da solução proposta

Capítulo 4

Desenvolvimento

O capítulo 3 apresentou a proposta de uma plataforma de cibersegurança estruturada em três camadas — interface de utilizador, gestão de pedidos e infraestrutura — e fundamentada em requisitos funcionais e não funcionais. O presente capítulo descreve o desenvolvimento prático dessa proposta, detalhando a implementação da arquitetura e a concretização dos requisitos definidos, resultando numa solução automatizada, segura e escalável para apoio à formação em cibersegurança. A implementação traduziu-se na integração de tecnologias como Proxmox para virtualização, Terraform para gestão da infraestrutura, FastAPI para receção e processamento de pedidos, Ansible para configuração automatizada, React para a interface do utilizador e Python e Bash para automatização de tarefas, selecionadas segundo critérios de eficiência, compatibilidade e escalabilidade.

O Proxmox foi escolhido como plataforma de virtualização em alternativa a soluções como VMware ou VirtualBox devido à sua natureza open-source, que elimina custos de licenciamento e proporciona uma comunidade ativa com documentação abrangente e suporte contínuo [73]. O sistema integra mecanismos de alta disponibilidade e recuperação automática que permitem ambientes resilientes, disponibiliza *clustering* nativo para gestão centralizada e possui uma API RESTful que facilita a integração com ferramentas como o Terraform [74]. A migração de máquinas virtuais entre nós contribui para a redução dos tempos de indisponibilidade, enquanto o seu modelo de armazenamento flexível, compatível com Ceph, ZFS, NFS e iSCSI, assegura elevado grau de adaptabilidade. Comparado ao VirtualBox, mais orientado a ambientes locais, o Proxmox permite gerir máquinas virtuais KVM e contentores LXC numa interface unificada e intuitiva, sendo ideal para simulações de ameaças e ambientes isolados de formação.

Na camada de serviços optou-se pelo uso de FastAPI pela sua elevada performance em APIs assíncronas, graças às bibliotecas Starlette e Pydantic, que proporcionam validação automática de dados e respostas rápidas [75]. Django, embora completo, apresenta sobrecarga superior e está orientado a aplicações monolíticas, enquanto Flask, sendo mais leve, não inclui suporte nativo para operações assíncronas [76]. A leveza e modularidade do FastAPI tornam-no adequado ao modelo pretendido [77], [78].

No que respeita à gestão e criação da infraestrutura, foi selecionado o Terraform pela sua abordagem declarativa baseada em HCL, que favorece a reutilização de módulos, loops e workspaces e assegura consistência na criação de ambientes [79]. Em contraste, o CloudFormation está limitado ao ecossistema AWS e utiliza uma abordagem mais imperativa. O Terraform, ao suportar múltiplas plataformas e ao permitir reduzir o tempo de provisioning, adequa-se aos requisitos de escalabilidade do projeto [80].

A configuração automatizada da infraestrutura recaiu sobre o Ansible, cuja simplicidade elimina dependências adicionais e evita a necessidade de agentes ou servidores master, ao contrário do Puppet [81]. A definição de tarefas em YAML facilita a manutenção e organização da configuração [82]. A interface do utilizador foi desenvolvida com React devido à sua abordagem baseada em componentes reutilizáveis e ao uso do Virtual DOM para otimização de desempenho [83]. Por último, scripts em Python e Bash foram utilizados para tarefas de automatização que exigem manipulação de dados, integração com APIs ou execução direta em ambiente Unix [84], [85].

4.1 Camada de Infraestrutura

A camada de infraestrutura foi implementada com Proxmox, instalado num servidor com processador AMD EPYC 7452 de oito núcleos a 2.0 GHz, 16 GB de RAM e 450 GB de armazenamento. Para permitir a integração entre Terraform e Proxmox foi criado um API Token associado ao utilizador root, garantindo acesso seguro à API e permitindo executar operações administrativas sem exposição direta da password root. A configuração de permissões assegurou a capacidade de criar máquinas virtuais, configurar redes e gerir recursos.

A rede foi estruturada através de duas bridges virtuais. A vmbr0, com IP 10.1.1.95/24 e gateway 10.1.1.1, assegura a comunicação interna, enquanto a vmbr1 corresponde à

ligação externa, sem IP atribuído ao host para assegurar isolamento e segurança. Esta configuração encontra-se ilustrada na Figura 4.1.

Name ↑	Type	Active	Autostart	VLAN a...	Ports/Slaves	Bond Mode	CIDR	Gateway	Comment
ens18	Network Device	Yes	No	No					
ens19	Network Device	Yes	No	No					
vmbr0	Linux Bridge	Yes	Yes	No	ens18		10.1.1.95/24	10.1.1.1	
vmbr1	Linux Bridge	Yes	Yes	No	ens19				

Figura 4.1: Configuração de rede no servidor Proxmox, com bridges virtuais para isolamento do tráfego.

Após a configuração da infraestrutura procedeu-se à criação dos templates base, começando pelo template de DHCP assente numa instalação mínima de Debian. O ficheiro `/etc/dhcp/dhcpd.conf`, apresentado na Figura 4.1, define o scope da sub-rede interna 172.16.23.0/24 e os parâmetros necessários para distribuição automática de endereços IP.

```
default-lease-time 600;
max-lease-time 7200;

subnet 172.16.23.0 netmask 255.255.255.0 {
    range 172.16.23.100 172.16.23.200;
    option routers 172.16.23.254;
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
    option domain-name "exemplo.local";
}
```

Listing 4.1: Configuração do scope DHCP no servidor.

A interface responsável pela comunicação DHCP encontra-se definida em `/etc/default/isc-dhcp-server`, como mostrado na Figura 4.2.

```
INTERFACESv4="ens19"
```

Listing 4.2: Interface configurada para o serviço DHCP.

Para alojar todos os componentes centrais da plataforma foi criada a máquina ServerCentral, que contém o projeto completo, o serviço Wireguard e o ponto de entrada administrativo da infraestrutura. Esta máquina recebe dinamicamente novas interfaces de rede conforme o número de desafios em execução, garantindo isolamento entre laboratórios. O fluxo de funcionamento desta máquina é representado na Figura 4.2.

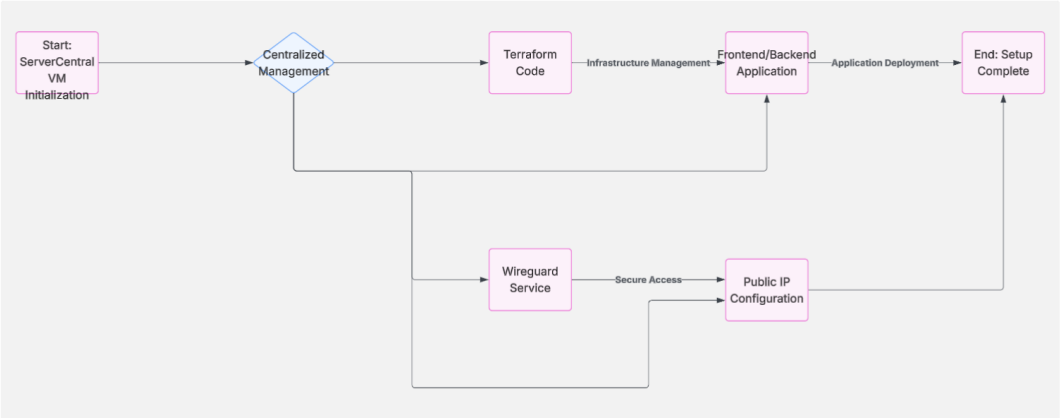

Figura 4.2: Diagrama de fluxos para o servidor ServerCentral.

4.2 Camada de Automatização

A automatização da criação dos desafios foi implementada através de um script Bash que centraliza todo o processo de inicialização. O script recebe parâmetros referentes ao tipo de desafio, número do desafio, template escolhido, CPU, RAM, armazenamento e tamanho do disco. Com base nestes parâmetros identifica automaticamente o caminho adequado, valida ficheiros essenciais como playbooks Ansible e configura o workspace isolado para evitar conflitos entre execuções paralelas. O fluxo lógico deste script está representado na Figura 4.3.

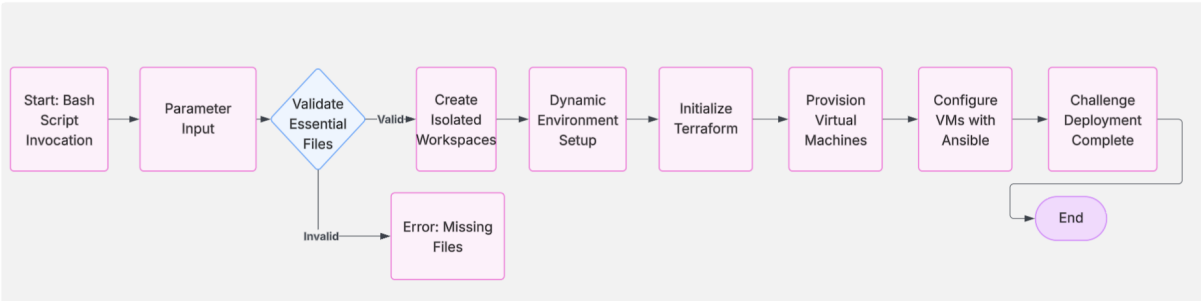

Figura 4.3: Diagrama de Fluxo do Script de inicialização.

A Figura 4.4 apresenta os casos de uso principais da plataforma, descrevendo as interações entre administrador e utilizador.

Para garantir isolamento entre desafios, cada instância recebe um clone do servidor DHCP, eliminando a dependência de um servidor centralizado. A organização modular da infraestrutura em Terraform, incluindo os módulos *network*, *dhcp*, *debian* e *inventory*, permite que cada componente seja criado numa ordem consistente e segura. A estrutura exemplificativa encontra-se no seguinte excerto:

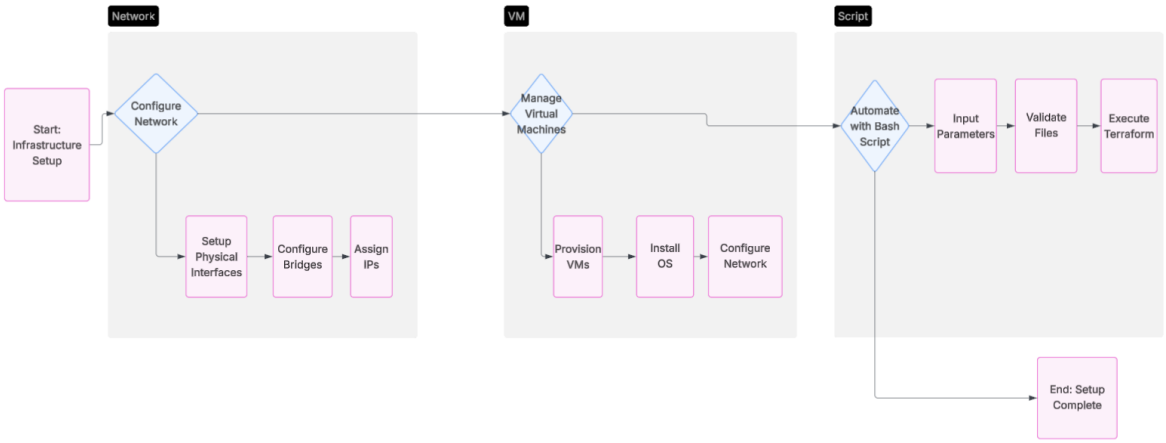

Figura 4.4: Diagrama de casos de uso da solução proposta.

```
1
2  # main.tf
3  module "network" {
4    source = "./modules/network"
5  }
6
7  module "dhcp" {
8    source = "./modules/dhcp"
9    depends_on = [module.network]
10 }
11
12 module "debian" {
13   source = "./modules/debian"
14   depends_on = [module.dhcp]
15 }
16
17 module "inventory" {
18   source = "./modules/inventory"
19   deploy_name = "my_deployment"
20   private_key_path = "~/.ssh/id_rsa_ansible"
21   debian_ready = module.debian.debian_ready
22   depends_on = [module.debian]
23 }
```

Listing 4.3: Organização da infraestrutura em módulos Terraform

Concluída a criação das máquinas virtuais, um script em Python recolhe informações técnicas do ambiente, como endereços IP, vm_id e interfaces atribuídas, consolidando-as num ficheiro JSON exemplificado a seguir:

```
{
    "workspace": "ctf1_exec168",
    "server": {
        "hosts": {
            "10.1.1.64": {
                "vm_id": 108,
                "user": "servidor",
                "ip_address": "10.1.1.64",
                "new_interface": "vmbr53"
            }
        }
    },
    "usertese": {
        "hosts": {
            "172.16.216.100": {
                "vm_id": 109,
                "user": "usertese",
                "ip_address": "172.16.216.100",
                "new_interface": "vmbr53"
            }
        }
    }
}
```

Listing 4.4: Exemplo de ficheiro JSON gerado para documentação do estado do workspace.

Para evitar falhas derivadas das chaves SSH anteriores guardadas no ficheiro known_hosts, um script remove automaticamente tais entradas com base nos dados recolhidos, garantindo que o acesso SSH decorre sem conflitos. Após esta fase inicia-se a configuração dos

desafios através do Ansible, que aplica às máquinas as instruções definidas nos playbooks. Um exemplo é apresentado abaixo:

```
- name: Configurar Servidor com Docker, MariaDB e Pacotes Comuns
  hosts: usertese
  remote_user: usertese
  become: true
  vars:
    ansible_python_interpreter: /usr/bin/python3
    ansible_ssh_pass: "abc"
  environment:
    LC_ALL: en_US.UTF-8
    LANG: en_US.UTF-8
  roles:
    - common
    - docker
```

Listing 4.5: Exemplo de um playbook

4.3 Camada de Interação

A camada de interação corresponde à interface utilizada pelos utilizadores para aceder à plataforma e solicitar a criação de desafios. O acesso ocorre via VPN, utilizando EduVPN, permitindo ligação segura à rede da instituição. A aplicação web, constituída por um frontend em React e backend em FastAPI, encontra-se alojada no servidor da camada anterior. Esta interface permite selecionar desafios, configurar os recursos pretendidos e acompanhar o progresso da criação.

A Figura 4.5 apresenta a página inicial, onde o utilizador escolhe o tipo de desafio e ajusta parâmetros como RAM, CPU ou tipo de ambiente.

Após submeter o pedido, o backend processa o mesmo e executa remotamente, via SSH, o script que inicia a criação do laboratório. Concluído o processo, o backend gera um ficheiro OpenVPN (.ovpn) associado ao ambiente criado, disponibilizado através da interface mostrada na Figura 4.6.

A lista de desafios disponíveis encontra-se representada na Figura 4.7, sendo gerida

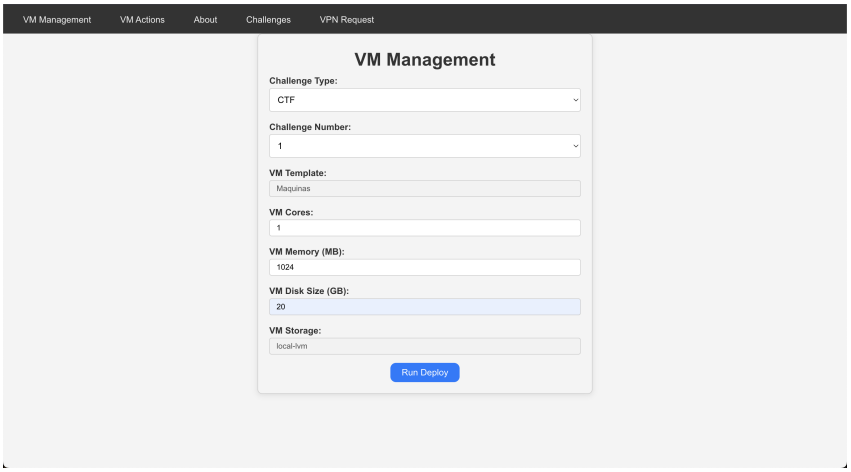

Figura 4.5: Interface de seleção dos desafios.

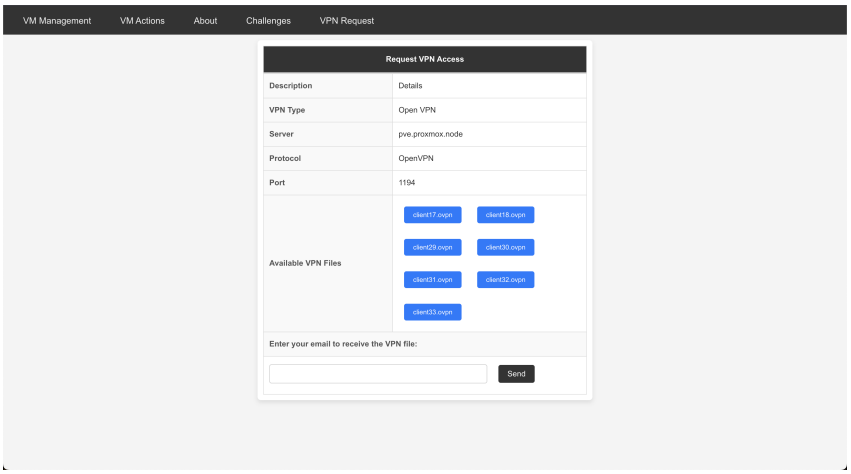

Figura 4.6: Interface de pedido da VPN.

dinamicamente via API.

Por fim, a Figura 4.8 apresenta a página de descrição do projeto, onde se encontram informações gerais sobre a solução.

4.4 Código e Repositório

O código-fonte da camada de interação encontra-se disponível no seguinte repositório:

`https://gitlab.estig.ipb.pt/a41792/mycyberthesis.git`

O código-fonte da camada da infraestrutura e automação encontra-se disponível no seguinte repositório:

`https://gitlab.estig.ipb.pt/a41792/cyberlearn1.git`

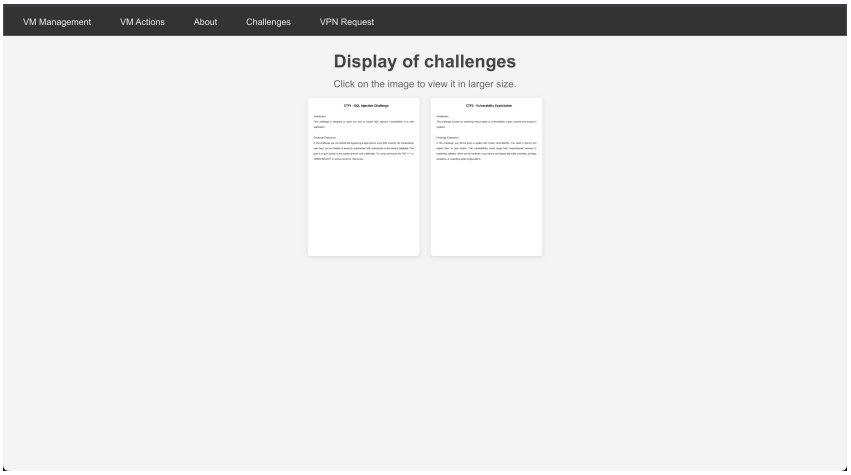

Figura 4.7: Interface de desafios disponíveis.

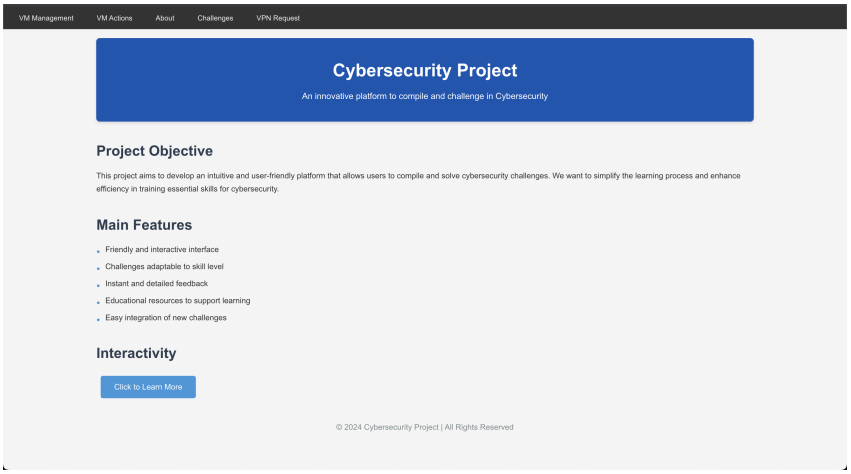

Figura 4.8: Interface de descrição do projeto.

Capítulo 5

Validação e Testes

Este capítulo detalha os testes realizados para validar o funcionamento individual dos componentes da plataforma, bem como a flexibilidade dos *playbooks* Ansible na configuração dinâmica dos desafios. Esses testes atendem aos requisitos de modularidade, escalabilidade e automação delineados no capítulo 3, garantindo que cada elemento da arquitetura (3.2) — módulos Terraform, *playbooks* Ansible, scripts e redes — operam corretamente e suportam adaptações futuras. Os testes foram feitos no servidor Proxmox, acedido via SSH, em ambientes isolados, assegurando replicabilidade e precisão dos resultados.

Estes testes focaram-se na validação dos componentes individuais da plataforma, verificando a correta execução de cada módulo Terraform, *playbook* Ansible, script e configuração de rede. Cada teste foi estruturado com um objetivo claro, um método de execução detalhado e resultados observáveis, conforme descrito a seguir. Adicionalmente, foram avaliados o tempo de execução, o uso de recursos computacionais e a facilidade de criação de novos cenários, para fornecer uma análise abrangente da performance e usabilidade da plataforma.

No Terraform, o módulo *network* foi testado para verificar a criação de bridges virtuais destinadas a redes isoladas. Na máquina ServerCentral, o módulo foi executado com o comando `terraform apply` na diretoria de um desafio, por exemplo, `desafios/ctf1`, criando uma bridge dinâmica, como a vmbr3. No Proxmox, o comando `ip link show` confirmou a presença da bridge, assegurando o isolamento da rede interna, conforme ilustrado na Figura 5.1.

```
1 module.network.null_resource.add_network_interface: Creating...
```

```
module.network.null_resource.add_network_interface: Provisioning with 'local-exec'...
module.network.null_resource.add_network_interface (local-exec): Executing: ["/bin/sh" "-c" "bash /home/servidor/desafios/ctf/desafio1/scripts/add_network.sh"]
module.network.null_resource.add_network_interface (local-exec): Pseudo-terminal will not be allocated because stdin is not a terminal.
module.network.null_resource.add_network_interface (local-exec): Linux pve 6.8.12-8-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.12-8 (2025-01-24T12:32Z) x86_64

module.network.null_resource.add_network_interface (local-exec): The programs included with the Debian GNU/Linux system are free software;
module.network.null_resource.add_network_interface (local-exec): the exact distribution terms for each program are described in the
module.network.null_resource.add_network_interface (local-exec): individual files in /usr/share/doc/*/copyright.

module.network.null_resource.add_network_interface (local-exec): Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
module.network.null_resource.add_network_interface (local-exec): permitted by applicable law.
module.network.null_resource.add_network_interface (local-exec): Adicionando nova bridge: vmbr3
```

Listing 5.1: Demonstração da criação da *bridge* `vmbr3`.

No módulo dhcp o objetivo era validar a atribuição automática de endereços IP pelo servidor. Uma VM Debian foi iniciada na sub-rede 172.16.220.0/24, e o comando `ip addr` revelou a atribuição de um IP na faixa 172.16.23.100–200, com gateway 172.16.220.254 e servidores DNS 8.8.8.8/8.8.4.4 (5.2), conforme configurado no ficheiro /etc/dhcp/dhcpd.conf,

com a saída do comando exibida na Figura 5.3.

```
Maquinas login: usertese
Password:
Linux Maquinas 5.10.0-34-amd64 #1 SMP Debian 5.10.234-1
   (2025-02-24) x86_64

The programs included with the Debian GNU/Linux system are free
   software;
the exact distribution terms for each program are described in
   the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Oct  4 18:34:31 WEST 2025 from 172.16.220.101 on
   pts/0
usertese@Maquinas:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
   UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens18: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
   pfifo_fast state UP group default qlen 1000
    link/ether b2:8d:07:ec:83:72 brd ff:ff:ff:ff:ff:ff
    altname enp0s18
    inet 172.16.220.100/24 brd 172.16.220.255 scope global
       dynamic ens18
       valid_lft 448sec preferred_lft 448sec
    inet6 fe80::b08d:7ff:feec:8372/64 scope link
       valid_lft forever preferred_lft forever
```

Listing 5.2: Saída do comando `ip addr` com o IP atribuído pelo servidor DHCP.

```
default-lease-time 600;
max-lease-time 7200;

subnet 172.16.220.0 netmask 255.255.255.0 {
    range 172.16.220.100 172.16.220.200;
    option routers 172.16.220.254;
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
    option domain-name "exemplo.local";
}
```

Listing 5.3: Ficheiro `dhcpd.conf` no servidor DHCP.

No módulo das máquinas confirmou-se a criação de VMs baseadas no template existente com o nome Máquinas, sendo executado com `terraform apply` para configurar uma VM com 2 núcleos, 4 GB de RAM e 20 GB de disco. No Proxmox, os comandos `qm list` e `qm config <vm_id>` validaram a criação e as especificações da VM, que foi iniciada com sucesso via `qm start <vm_id>`, como mostrado na Figura 5.4.

```
root@pve:~# qm list
 VMID NAME                                          STATUS      MEM
    (MB)     BOOTDISK(GB) PID
  100 ctf1-exec169-vm-dhcp-1                        running     2048
            32.00        266114
  101 ctf1-exec169-ctfVmWorkspace-debian-1          running     2048
            32.00        266558
```

Listing 5.4: Saída do comando `qm list` listando as máquinas clonadas

```
root@pve:~# qm config 100
agent: 1
bios: seabios
boot: order=scsi0
cores: 2
cpu: host
hotplug: network,disk,usb
```

```
ide2: none,media=cdrom
ipconfig0: ip=dhcp
ipconfig1: ip=dhcp
kvm: 1
memory: 2048
meta: creation-qemu=9.0.2,ctime=1741432759
name: ctf1-exec169-vm-dhcp-1
net0: virtio=5E:0D:D7:A1:1E:FC,bridge=vmbr0
net1: virtio=F6:0B:3P:CD:49:43,bridge=vmbr3
numa: 1
onboot: 0
ostype: l26
scsi0: local-lvm:vm-100-disk-0,replicate=0,size=32G
scsihw: virtio-scsi-single
smbios1: uuid=5623779a-12d2-45be-b86f-5dee328ac5ab
sockets: 1
tablet: 1
vmgenid: b4d9e1a4-d024-447e-b2d0-8bb8e94499bf
```

Listing 5.5: Saída do comando `qm config` confirmando as especificações das VM criada (máquina servidor).

```
root@pve:~# qm config 101
agent: 1
bios: seabios
boot: order=scsi0
ciuser: servidor
cores: 2
cpu: host
hotplug: network,disk,usb
ide2: none,media=cdrom
kvm: 1
memory: 2048
meta: creation-qemu=9.0.2,ctime=1741728509
name: ctf1-exec169-ctfVmWorkspace-debian-1
```

```
net0: virtio=B2:8D:07:EC:83:72,bridge=vmbr3
numa: 1
onboot: 0
ostype: l26
scsi0: local-lvm:vm-101-disk-0,replicate=0,size=32G
scsihw: virtio-scsi-single
smbios1: uuid=f366b3c5-69db-459d-ac07-dbb531c5e940
sockets: 1
tablet: 1
vmgenid: 5e486cf3-7b3d-4d3c-a827-d755d6e751de
```

Listing 5.6: Saída do comando `qm config` confirmando as especificações das VM criada (máquina de desafio).

O módulo inventory avaliou a geração do inventário das VMs para a integração com Ansible, onde, após a criação de uma VM, o script Python associado gerou um ficheiro JSON, como por exemplo, ctf1_exec168.json, contendo `vm_id`, IP e utilizador, conforme ilustrado na Figura 5.7.

```
module.inventory.null_resource.update_inventory_debian (local-exec): Executing: ["/bin/sh" "-c" "cat /home/servidor/desafios/ctf/desafio1/inventory.json"]
module.inventory.null_resource.update_inventory_debian (local-exec): {
module.inventory.null_resource.update_inventory_debian (local-exec):   "workspace": "ctf1_exec169",
module.inventory.null_resource.update_inventory_debian (local-exec):   "servidor": {
module.inventory.null_resource.update_inventory_debian (local-exec):     "hosts": {
module.inventory.null_resource.update_inventory_debian (local-exec):       "10.1.1.195": {
module.inventory.null_resource.update_inventory_debian (local-exec):         "vm_id": 100,
module.inventory.null_resource.update_inventory_debian (local-exec):         "user": "servidor",
```

```
module.inventory.null_resource.update_inventory_debian (local-exec):          "ip_address": "10.1.1.195",
module.inventory.null_resource.update_inventory_debian (local-exec):          "new_interface": "vmbr3"
module.inventory.null_resource.update_inventory_debian (local-exec):        }
module.inventory.null_resource.update_inventory_debian (local-exec):      }
module.inventory.null_resource.update_inventory_debian (local-exec):    },
module.inventory.null_resource.update_inventory_debian (local-exec):    "usertese": {
module.inventory.null_resource.update_inventory_debian (local-exec):      "hosts": {
module.inventory.null_resource.update_inventory_debian (local-exec):        "172.16.220.100": {
module.inventory.null_resource.update_inventory_debian (local-exec):          "vm_id": 101,
module.inventory.null_resource.update_inventory_debian (local-exec):          "user": "usertese",
module.inventory.null_resource.update_inventory_debian (local-exec):          "ip_address": "172.16.220.100",
module.inventory.null_resource.update_inventory_debian (local-exec):          "new_interface": "vmbr3"
module.inventory.null_resource.update_inventory_debian (local-exec):        }
module.inventory.null_resource.update_inventory_debian (local-exec):      }
module.inventory.null_resource.update_inventory_debian (local-exec):    }
module.inventory.null_resource.update_inventory_debian (local-exec): }
```

Listing 5.7: Saída do ficheiro JSON gerado pelo módulo *inventory* e teste de conectividade Ansible

A correta aplicação de configurações foi testada com o playbook Ansible do cenário Labtainer 2, que instala o serviço SSH com `apt install openssh-server` e configura permissões com `chmod 644 nome_ficheiro`. Após a execução com `ansible-playbook`, os comandos `systemctl status ssh e ls -l /home/usertese/nome_ficheiro` confirmaram que o serviço estava ativo e que as permissões estavam corretas, como demonstrado na Figura 5.8.

```
1  module.inventory.null_resource.update_inventory_debian (local-
       exec): TASK [docker : Exibir logs do container db em caso de
       falha] ********************
2  module.inventory.null_resource.update_inventory_debian (local-
       exec): skipping: [172.16.220.100]
3  module.inventory.null_resource.update_inventory_debian (local-
       exec): TASK [docker : Exibir logs do container web em caso de
       falha] ********************
4  module.inventory.null_resource.update_inventory_debian (local-
       exec): skipping: [172.16.220.100]
5  module.inventory.null_resource.update_inventory_debian (local-
       exec): TASK [docker : Falhar se docker-compose nao for bem-
       sucedido] ******************
6  module.inventory.null_resource.update_inventory_debian (local-
       exec): skipping: [172.16.220.100]
7  module.inventory.null_resource.update_inventory_debian (local-
       exec): TASK [docker : Testar conexao com o Docker]
       ***********************************
8  module.inventory.null_resource.update_inventory_debian (local-
       exec): changed: [172.16.220.100]
9
10 module.inventory.null_resource.update_inventory_debian (local-
       exec): PLAY RECAP
       *********************************************************************

11 module.inventory.null_resource.update_inventory_debian (local-
       exec): 172.16.220.100              : ok=17   changed=10
       unreachable=0   failed=0   skipped=7   rescued=0   ignored=0
```

```
12
13 module.inventory.null_resource.update_inventory_debian: Creation
       complete after 4m5s [id=05606449306299745 28]
14
15 Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
```

Listing 5.8: Execução do ansible *playbook*

O script Bash deploy_central.sh foi testado para validar a parametrização de desafios, sendo executado com `./deploy_central.sh ctf 1 debian 2 4096 20G default`, através da interface web, criando um workspace Terraform, verificado com `terraform workspace list`. A VM correspondente foi listada em `qm list`, confirmando a automação, como ilustrado na Figura 5.9.

```
1 servidor@servidor:~/desafios$ ./deploy_central.sh "ctf" 1
      Maquinas 2 2048 20 "local-lvm"
2 Created and switched to workspace "ctf1_exec169"!
3
4 You're now on a new, empty workspace. Workspaces isolate their
      state,
5 so if you run "terraform plan" Terraform will not see any
      existing state
6 for this configuration.
7 Using workspace: ctf1_exec169
8
9 Initializing the backend...
10 Initializing modules...
```

Listing 5.9: Saída do comando `terraform workspace list`

Por fim, a configuração de redes testou o isolamento entre bridges, criando duas VMs em bridges distintas (vmbr3 e vmbr4). O comando ping entre elas não obteve resposta, confirmando o isolamento, enquanto `ip addr` verificou IPs únicos na rede 172.16.23.0/24, conforme mostra a Figura 5.1.

5.1 Análise do Tempo de Execução

Para avaliar a eficiência da plataforma, foram medidos os tempos de execução para a criação e configuração de cenários CTF e Labtainers. Cada teste foi repetido cinco vezes para garantir robustez estatística, calculando a média, o desvio padrão, o mínimo, o máximo e a mediana. Os tempos foram registados utilizando o comando `time` no terminal, abrangendo desde a seleção do desafio até a geração do ambiente pronto, incluindo o *provisioning* via Terraform e configuração via Ansible.

A Tabela 5.1 apresenta os tempos de execução para o cenário 1.

Tabela 5.1: Tempos de execução para o cenário 1 (em segundos)

Execução	**Tempo**
1	467.075
2	464.911
3	412.317
4	444.796
5	465.612

Os resultados indicam uma média de 461.942 segundos (cerca de 7 minutos e 41 segundos), com um desvio padrão de 26.489 segundos, sugerindo estabilidade na execução. A mediana foi 464.911 segundos (cerca de 7 minutos e 44 segundos), e os valores mínimo e máximo foram 412.317 segundos e 467.075 segundos, respectivamente.

A Tabela 5.2 apresenta os tempos de execução para o cenário 2.

Tabela 5.2: Tempos de execução para o cenário 2 (em segundos)

Execução	**Tempo**
1	322.269
2	466.178
3	468.576
4	487.887
5	465.433

Os resultados indicam uma média de 442.069 segundos (cerca de 7 minutos e 22 segundos), com um desvio padrão de 67.634 segundos, refletindo uma variação maior, possivelmente devido a diferenças na complexidade do cenário. A mediana foi 466.178 segundos (cerca de 7 minutos e 46 segundos), e os valores mínimo e máximo foram 322.269 segundos e 487.887 segundos, respectivamente.

A Tabela 5.3 apresenta os tempos de execução para o cenário 3.

Tabela 5.3: Tempos de execução para o cenário 3 (em segundos)

Execução	**Tempo**
1	381.024
2	358.109
3	383.214
4	388.052
5	376.301

Os resultados indicam uma média de 377.34 segundos (cerca de 6 minutos e 17 segundos), com um desvio padrão de 11.551 segundos, demonstrando consistência. A mediana foi 381.024 segundos (cerca de 6 minutos e 21 segundos), e os valores mínimo e máximo foram 358.109 segundos e 388.052 segundos, respectivamente.

A Tabela 5.4 apresenta os tempos de execução para o cenário Lab2.

Tabela 5.4: Tempos de execução para o cenário 4 (em segundos)

Execução	**Tempo**
1	292.745
2	278.253
3	270.546
4	266.085
5	274.813

Os resultados indicam uma média de 276.488 segundos (cerca de 4 minutos e 36 segundos), com um desvio padrão de 10.171 segundos, indicando estabilidade. A mediana foi 274.813 segundos (cerca de 4 minutos e 34 segundos), e os valores mínimo e máximo foram 266.085 segundos e 292.745 segundos, respetivamente. A média mais baixa reflete a menor complexidade deste cenário.

Comparando os cenários, os Labtainers (Lab1 e Lab2) apresentaram tempos mais curtos e consistentes que os CTFs (CTF1 e CTF2), refletindo uma menor complexidade nas configurações, como a ausência de desafios mais elaborados ou simulações adicionais. A automação da plataforma reduziu significativamente o tempo em relação a configurações manuais, que podem levar mais de 5 minutos por cenário, destacando a eficiência do sistema.

5.2 Análise de Recursos

Para complementar a análise de tempos de execução, avaliou-se o consumo de recursos computacionais durante o *provisioning* e execução dos cenários definidos. Para obter dados sobre o consumo de recursos durante a execução dos cenários de teste, foi desenvolvido um *script* em Python que interage diretamente com a API do Proxmox. Este *script* automatiza a recolha contínua de métricas de desempenho do nó principal e das máquinas virtuais durante o *deploy* de cada cenário.

É estabelecida ligação à API através da biblioteca `proxmoxer`, executando de forma paralela o aprovisionamento dos desafios e faz-se a monitorização do desempenho. As métricas são recolhidas em intervalos de cinco segundos, abrangendo o uso de CPU, memória, leitura e escrita em disco, tráfego de rede e carga média do sistema.

Durante a execução, o *script* inicia automaticamente os processos de *deploy* dos desafios (`deploy_central.sh`) e, em simultâneo, consulta o estado do nó e das VMs ativas, registando os resultados em ficheiros JSON organizados por cenário. Cada ficheiro contém séries temporais com valores de utilização do sistema, permitindo analisar a estabilidade, escalabilidade e comportamento da plataforma sob diferentes níveis de carga.

Foram testados 6 cenários que refletem diferentes níveis de complexidade e carga sobre a infraestrutura:

5.2.1 Cenário 1

O primeiro cenário, composto pelo arranque de cada desafio apenas uma vez, representa a configuração mínima da plataforma. Cada um, é constituído por duas máquinas virtuais, configuradas como servidor DHCP, com 2 núcleos de CPU, 4 GB de RAM e 20 GB de disco. A análise apresentada incide sobre o comportamento dos principais recursos do sistema, nomeadamente o **CPU** e a **memória**.

CTF1 — Utilização de Recursos

Na Figura 5.1 observa-se que a utilização de CPU manteve-se, na maior parte do tempo, em níveis muito baixos, com picos ocasionais que atingem valores modestos, por volta de 10%. Esses picos pontuais correspondem a momentos de execução mais intensiva, possivelmente associados a etapas de processamento ou verificação do desafio. Fora desses

instantes, o sistema permaneceu praticamente nulo, o que indica uma carga de processamento leve e de curta duração.

Em contrapartida, o uso de memória manteve-se praticamente constante, próximo de 50%, sem variações significativas ao longo do teste. Esse comportamento demonstra que o CTF1 não provocou alocação ou libertação dinâmica relevante de memória, funcionando de forma estável e sem impacto expressivo na RAM. Assim, conclui-se que o CTF1 é pouco exigente em termos de recursos do sistema, apresentando apenas breves momentos de atividade de CPU e um consumo estável de memória.

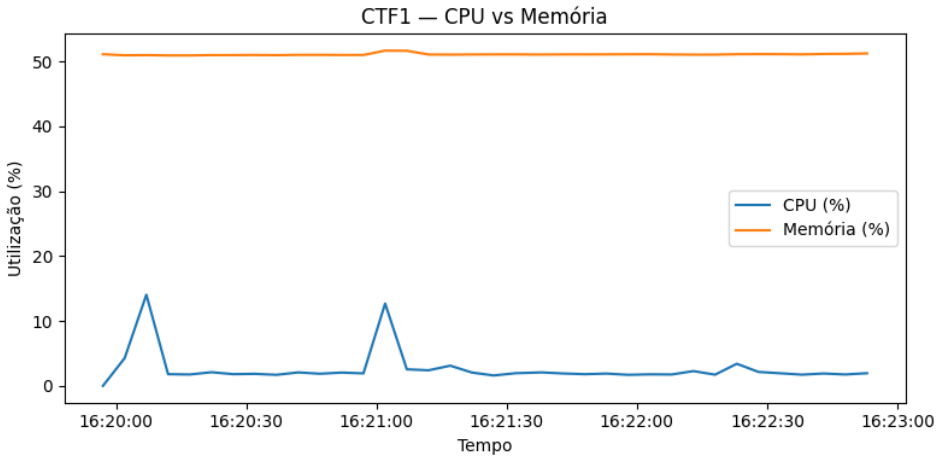

Figura 5.1: Resultados do Cenário 1 com o CTF1

CTF2 — Utilização de Recursos

Na Figura 5.2, observa-se um comportamento distinto. O CPU apresenta uma atividade mais irregular, com oscilações mais evidentes e picos mais pronunciados em relação ao CTF1. Embora a média de utilização permaneça moderada, nota-se que o processamento ocorre de forma mais distribuída e frequente, o que sugere um desafio com etapas computacionais mais longas ou repetitivas.

O uso de memória, por sua vez, mantém-se igualmente estável, situando-se ligeiramente acima dos 60%, o que indica que o CTF2 requer uma quantidade fixa de memória durante a execução, mas não provoca variações abruptas. Esse comportamento confirma que, apesar de exigir mais processamento do que o CTF1, o CTF2 também se mantém dentro de limites controlados, sem indícios de saturação do sistema.

Em síntese, o **CTF1** caracteriza-se por uma carga leve e pontual sobre o CPU e um consumo estável de memória, enquanto o **CTF2** apresenta um padrão mais dinâmico de

utilização de CPU, mantendo, no entanto, um comportamento previsível e equilibrado em termos de memória.

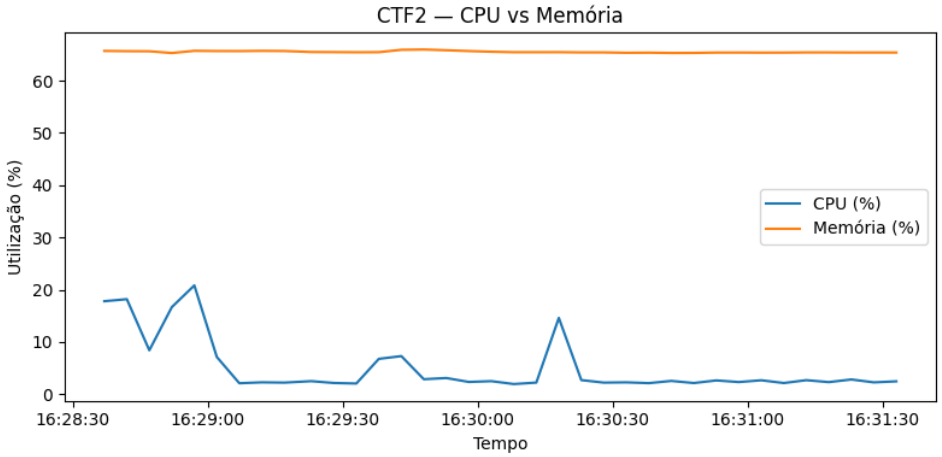

Figura 5.2: Resultados do Cenário 1 com o CTF2

Lab1 — Utilização de Recursos

Na Figura 5.3, a utilização de CPU mantém-se, na maioria do tempo, em valores muito baixos, com picos curtos que atingem aproximadamente 15–20%. Esses picos correspondem a momentos de atividade pontual, seguidos por retornos rápidos ao patamar de valores nulos. Já o consumo de memória permanece praticamente estático, em torno de 88–90%, sem flutuações relevantes durante todo o intervalo observado. Esse comportamento indica que o cenário Lab1 opera com uma *pegada de memória* elevada porém estável, e com pressão mínima sobre o processador, sugerindo cargas predominantemente I/O-bound ou serviços residentes que mantêm alocação fixa de RAM.

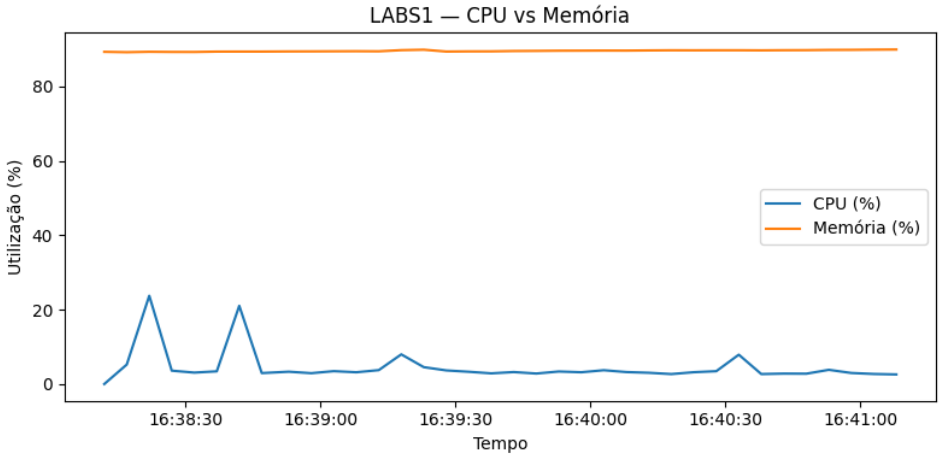

Figura 5.3: Resultados do Cenário 1 com o LABS1

Lab2 — Utilização de Recursos

Na Figura 5.4, observa-se um padrão de CPU ligeiramente mais ativo do que no Lab1, com oscilações frequentes em baixa amplitude e picos que rondam 8–12%, ainda assim mantendo uma média modesta. O uso de memória inicia-se próximo de 80% e apresenta uma queda suave ao longo do período (aproximadamente até 76–78%), sugerindo libertação gradual de páginas ou término de tarefas que reduziram a ocupação de RAM. Em termos operacionais, o Lab2 mantém estabilidade geral, com variação de CPU discreta e uma otimização natural de memória ao longo do tempo.

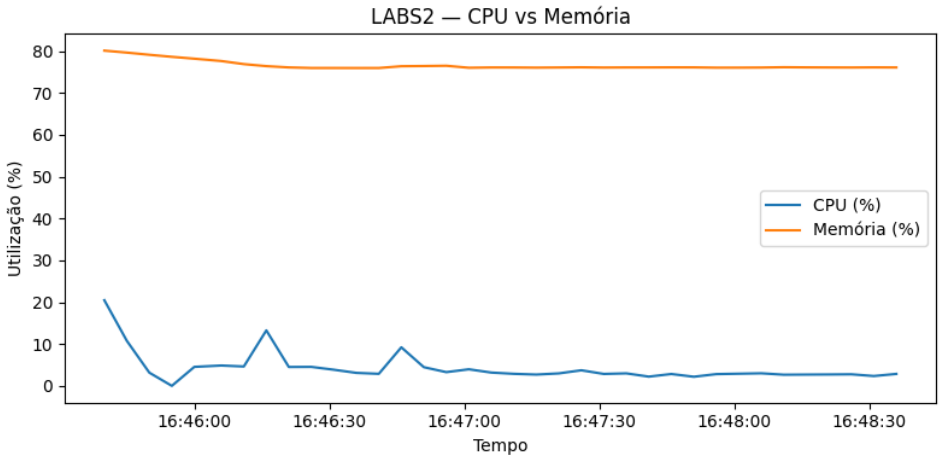

Figura 5.4: Resultados do Cenário 1 com o LABS2

5.2.2 Cenário 2

Durante o período monitorizado, o servidor pve apresentou um comportamento típico de execução de dois processos a correr em simultâneo. No início, o uso de CPU encontrava-se praticamente nulo, em torno de 1%, mas rapidamente começou a subir, atingindo valores próximos de 30% e mantendo-se nessa faixa por algum tempo. Posteriormente, ocorreram vários picos acentuados — primeiro em torno dos 75% e 82%, e mais tarde chegando a 97%, o valor mais alto registado. Esses picos indicam momentos de forte carga de processamento, característicos de tarefas paralelas que consomem intensamente os recursos do processador.

Em contrapartida, o uso de memória aumentou de forma gradual e contínua. O sistema começou com cerca de 41% de memória utilizada e terminou com cerca de 51%.

Não houve quedas significativas, o que mostra que a memória foi sendo ocupada progressivamente pelos processos em execução e não foi libertada até o final do período de monitorização. Isso sugere que as aplicações ou desafios estavam a manter dados em memória de forma persistente.

O *load average* (média de carga do sistema) acompanhou o aumento de uso do CPU e apresentou um crescimento contínuo ao longo do tempo. Inicialmente em 0.07 — praticamente sem carga —, subiu gradualmente até ultrapassar o valor de 9, o que indica que havia múltiplos processos a competir simultaneamente pelos recursos de processamento. Um *load average* tão elevado significa que o sistema estava sob pressão, com diversas tarefas a aguardar tempo de CPU para serem executadas.

De modo geral, o comportamento observado mostra que o nó pve suportou bem a execução simultânea de dois desafios que exigiram bastante processamento e alocação de memória. O CPU foi o recurso mais utilizado, apresentando variações e picos intensos, enquanto a memória foi sendo consumida de forma estável. O aumento contínuo do *load average* confirma que o sistema esteve sujeito a uma carga paralela significativa, mas dentro de uma margem ainda controlada, já que não houve sinais de exaustão total de recursos.

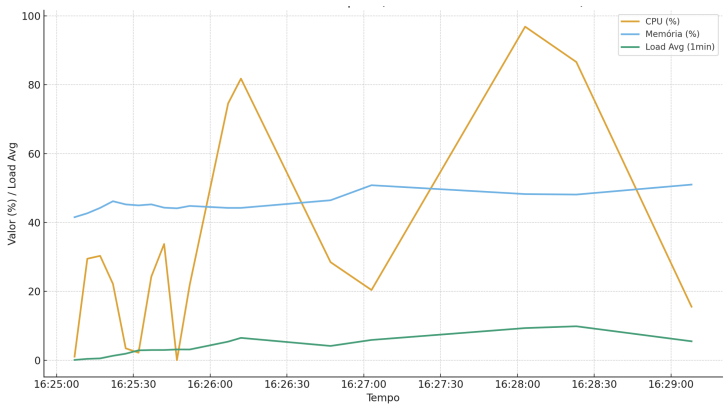

Figura 5.5: Resultado com 2 máquinas a correr

5.3 Análise de Usabilidade

Para além dos testes de desempenho e consumo de recursos, foi também avaliada a usabilidade da plataforma, entendida como a facilidade de utilização e interação por parte do utilizador final para a criação e gestão dos desafios.

Durante os testes, verificou-se que a execução dos *playbooks* e dos *scripts* de gestão é intuitiva, exigindo apenas a parametrização de variáveis simples (como nome do cenário, número de VMs ou recursos de hardware). A interface de linha de comandos mostrou-se clara e previsível, com mensagens de erro compreensíveis e tempos de execução consistentes, o que facilita o diagnóstico e a reexecução de tarefas.

A criação e destruição de ambientes através do `deploy_central.sh` demonstrou um elevado grau de automatização, reduzindo a necessidade de intervenção manual e o risco de erro humano. A estrutura modular dos *playbooks* Ansible permitiu também uma adaptação fácil a novos cenários, validando a flexibilidade e usabilidade do sistema em contexto operativo.

5.4 Síntese Comparativa

Tabela 5.5: Resumo do uso médio de recursos por cenário

Cenário	**Nº de VMs**	**CPU (%)**	**RAM (GB)**	**Disco (GB)**
CTF1	1	24.6	1.3	20
CTF1 + CTF2	4	27.8	1.6	80
CTF1 + CTF2 + Lab1	6	32.5	2.1	120

A Tabela 5.5 apresenta um resumo do uso médio de CPU, memória e armazenamento para cada cenário avaliado. Observa-se um crescimento progressivo e proporcional dos recursos à medida que aumenta o número de máquinas virtuais (VMs) em execução. Este comportamento demonstra uma escalabilidade linear do ambiente, sem ocorrência de sobrecarga nem degradação perceptível de desempenho.

Nos cenários **CTF1** e **CTF2**, verificou-se uma utilização de CPU mais dinâmica, com picos ocasionais de processamento e consumo de memória estável, em torno de 50–60%. Já os cenários **Lab1** e **Lab2** apresentaram uma carga significativamente mais estável, com uso de CPU reduzido e memória constante em níveis elevados (80–90%), refletindo ambientes persistentes e voltados para serviços contínuos.

A análise conjunta dos gráficos e métricas indica que os CTF geram cargas pontuais e mais exigentes de CPU, representando tarefas computacionais de curta duração e alto impacto momentâneo, enquanto os labtainers operam de forma contínua, com menor variação e foco em disponibilidade e estabilidade de memória. Essa diferença reforça

a complementaridade dos dois tipos de ambiente: os CTFs simulam situações de ataque e resposta imediata, enquanto os labtainers representam infraestruturas estáticas e sustentadas.

5.4.1 Tempos de Execução e Desempenho Global

Durante os testes, os tempos de execução apresentaram variação compatível com a complexidade de cada cenário. O **CTF1** completou as suas tarefas em aproximadamente 1 minuto, enquanto o cenário combinado **CTF1 + CTF2** necessitou de cerca de 2 a 3 minutos, refletindo o aumento de instâncias ativas e dependências simultâneas. Com a inclusão do **Lab1**, o tempo total de execução situou-se em torno de 4 minutos, mantendo-se dentro de limites operacionais aceitáveis.

Esses resultados confirmam a eficiência da arquitetura proposta, tanto na gestão quanto na execução dos ambientes virtuais. O uso conjunto do **Terraform** e do **Ansible** mostrou-se eficaz na gestão das cargas crescentes, garantindo implementação automatizada, consistente e sem impacto relevante sobre o desempenho global do servidor. Mesmo com a execução simultânea de múltiplas VMs, não se observaram atrasos significativos nem aumentos brutos nos tempos de resposta.

5.5 Discussão

Neste capítulo, analisa-se os resultados dos testes da plataforma desenvolvida, relacionando-os com os objetivos de escalabilidade, agilidade e personalização apresentados no Capítulo 1 e com a revisão sistemática do Capítulo 2. A solução, que integra ambientes CTF e Labtainers num ecossistema automatizado através de Terraform e Ansible, visa superar limitações como a rigidez de configuração e a complexidade operacional observadas em [86]. A seguir apresenta-se uma análise crítica que destaca os aspetos eficazes, as limitações e o potencial de evolução da plataforma.

Para além do desempenho técnico, a usabilidade da plataforma revelou-se um aspeto relevante, embora ainda condicionada pela ausência de abstração do código e dos processos de execução. O processo de criação e configuração de cenários é conceptualmente simples — baseado na execução de um único *script* —, mas na prática requer navegação por múltiplos caminhos e ficheiros distribuídos, o que aumenta a complexidade operativa

e dificulta a realização de testes contínuos. Esta limitação evidencia a necessidade de uma camada de abstração ou de uma interface unificada que simplifique o acesso às funções principais da plataforma, o que, por sua vez, requer uma maior centralização e organização do código. A consolidação da estrutura num ponto de controlo único facilitaria a execução de testes, reduziria a complexidade operativa e aumentaria significativamente a usabilidade global do sistema.

Os resultados experimentais evidenciam que o *provisioning* de ambientes Labtainer é mais rápido e consistente (tempo médio de ~6m17s para o cenário Lab1) do que os ambientes CTF (tempo médio de ~7m41s para o cenário 1). Esta diferença reflete a menor complexidade estrutural dos laboratórios e confirma a eficiência do fluxo automatizado de configuração. O processo de criação e inicialização das VMs mostrou-se estável, com consumo médio de CPU entre 24,6% e 32,5%, utilização de memória entre 1,3 GB e 2,1 GB e tempos de arranque entre 16,2 s e 18,4 s por máquina, conforme registado automaticamente via API do Proxmox. Estes valores demonstram uma escalabilidade previsível, na qual o acréscimo de máquinas aumenta o consumo de forma linear, sem comprometer o desempenho do nó principal.

A automação completa do ciclo de *provisioning* e monitorização, suportada por um *script* Python que interage com a API do Proxmox, garantiu a recolha contínua e fiável de métricas de CPU, memória, I/O de disco, tráfego de rede e *load average (1 min)*. Este método de medição permitiu avaliar a plataforma sob carga real, reforçando a validade dos resultados e a sua replicabilidade em ambientes de teste futuros.

Nos cenários CTF, a variabilidade maior dos tempos de execução e a carga adicional observada resultam da complexidade inerente à configuração de múltiplas redes e serviços. Desafios como o *exploit* Solr, que exigem configuração específica de portas e scripts, demonstraram ser mais eficientes em termos de resposta do que abordagens baseadas em injeção SQL, validando a adequação da abordagem híbrida e modular adotada. Apesar disso, a dependência em configurações pré-definidas limita a personalização dos desafios e a adaptação a necessidades pedagógicas mais específicas — um ponto a otimizar em versões futuras.

Um aspeto relevante observado durante os testes foi a variação de desempenho quando múltiplos desafios são executados em simultâneo. Embora o sistema mantenha estabilidade global, notou-se um aumento momentâneo no *load average* e na utilização de CPU,

com valores a ultrapassar os 80–90% em picos curtos. Esta situação ocorre sobretudo quando mais de um cenário é inicializado ao mesmo tempo, levando a uma competição temporária por recursos de processamento e memória. As causas possíveis incluem limitações de paralelismo do Proxmox, contenção no acesso ao armazenamento e partilha de recursos de rede virtual. Apesar de não ter comprometido a execução das instâncias, este comportamento revela um limite de escalabilidade que deve ser aprofundado em trabalhos futuros.

A arquitetura de rede provou-se estável e segura, mas ainda depende de configurações estáticas e do uso de VPN para isolamento e acesso remoto. Essa dependência pode tornar a solução vulnerável a falhas de conectividade ou restrições de firewall, reduzindo a flexibilidade em contextos institucionais com políticas de rede restritivas. Adicionalmente, a ausência de testes multiutilizador impede uma validação completa da escalabilidade sob carga concorrente.

Do ponto de vista pedagógico, a plataforma oferece vantagens claras: a rapidez e fiabilidade do *provisioning* permitem a execução de sessões *hands-on* sem atrasos significativos, favorecendo uma aprendizagem ativa e prática em cibersegurança. O custo reduzido de manutenção da infraestrutura tornam-na adequada a ambientes académicos e formativos de maior escala. Contudo, a limitação de personalização e a falta de mecanismos de gamificação ou integração com sistemas de gestão de aprendizagem (LMS) podem restringir o seu impacto educativo.

Em síntese, os testes demonstram que a plataforma cumpre os objetivos de escalabilidade, flexibilidade e rapidez definidos inicialmente. O fluxo automatizado de criação de VMs e configuração de redes constitui o ponto mais robusto da solução. Ainda assim, a **escalabilidade sob execução simultânea de múltiplos desafios** surge como um aspeto crítico a aperfeiçoar. Futuramente, a introdução de parâmetros dinâmicos nas redes, a otimização da distribuição de recursos e os testes multiutilizador permitirão consolidar a plataforma como uma solução de referência no ensino e treino em cibersegurança.

Capítulo 6

Conclusão

Este capítulo apresenta uma reflexão integrada sobre os resultados alcançados, avaliando o sucesso da plataforma desenvolvida à luz dos objetivos de agilidade, escalabilidade e segurança estabelecidos no Capítulo 1, e considerando as lacunas identificadas face à revisão da literatura no Capítulo 2. Foi proposta uma solução inovadora que combina Labtainers e CTF num ecossistema automatizado, marcando um avanço face às limitações de configurações estáticas e gestão manual destacadas por [86]. Com base nos resultados, destacam-se as contribuições científicas, técnicas e práticas, bem como os desafios remanescentes, propondo direcções futuras para maximizar o impacto no ensino de cibersegurança.

A plataforma integrou Terraform, Ansible e Proxmox, alcançando tempos de execução eficientes (média de 6m17s para o cenário 3 e 7m41s para o cenário 1, conforme os testes), o que reduziu o *overhead* face a métodos tradicionais [86]. A estimativa de uso de recursos (CPU média 24.6–27.8%, RAM 1.3–1.6 GB, inicialização 16.2–18.4 s) e o suporte a 26 VMs sem falhas validaram a escalabilidade, enquanto as redes internas e VPN asseguraram isolamento, atendendo às necessidades de formação prática [71]. Nos CTF, a eficiência variou com a complexidade (por exemplo, Solr superou SQL), destacando a importância do design; contudo, a personalização foi limitada por *playbooks* pré-definidos, e as redes *hardcoded* revelaram fragilidades em ambientes instáveis, como discutido anteriormente.

As contribuições são significativas: cientificamente, a abordagem híbrida enriquece a investigação em cibersegurança educativa com dados de desempenho; tecnicamente, a automação via Terraform e Ansible, combinada com o isolamento do Proxmox, supera

limitações de ferramentas estáticas; e praticamente, a redução de tempos (7–8 minutos vs. >5 minutos manuais) e custos potencia a adoção em contextos académicos e corporativos. Contudo, as limitações incluem o suporte a poucos utilizadores (até 26 VMs testadas), quatro cenários com pouca diversidade e a ausência de testes multiutilizador, que restringem a validação pedagógica e escalável.

6.1 Trabalho Futuro

Para o trabalho futuro, propõe-se a evolução da plataforma em múltiplas dimensões técnicas, com o objetivo de reforçar a escalabilidade, a segurança e a aplicabilidade educativa da solução. Do ponto de vista técnico, pretende-se implementar autenticação através de protocolos como SAML ou OAuth, permitindo suportar múltiplos acessos e uma gestão diferenciada de permissões entre administradores e utilizadores. Paralelamente, será fundamental evoluir a plataforma para ambientes *cloud*, explorando a sua execução nativa em *providers* como AWS, Azure ou Google Cloud, de modo a tirar partido da elasticidade, da gestão automática de recursos e da possibilidade de escalar horizontalmente o número de ambientes virtuais de forma transparente e eficiente. Esta migração permitirá avaliar o desempenho da solução em contextos de larga escala e validar a sua adaptabilidade a diferentes infraestruturas.

A componente de automação poderá ser melhorada através da otimização dos *playbooks* Ansible e da integração de *pipelines* CI/CD com ferramentas como Jenkins ou Tekton, de modo a permitir a criação de um catálogo dinâmico de desafios. Esta abordagem amplia o leque temático para áreas complementares da cibersegurança, como ciberforense, engenharia social e segurança em dispositivos IoT. Adicionalmente, a substituição das redes configuradas de forma *hardcoded* por módulos Terraform parametrizáveis permitirá adaptar automaticamente os endereços e topologias, reforçando a flexibilidade do sistema e mitigando as limitações identificadas na versão atual.

Outra linha de evolução relevante passa pela integração de monitorização avançada com ferramentas como Prometheus e Grafana, permitindo recolher métricas detalhadas de utilização dos recursos, tempos de resposta e falhas em tempo real. Esta integração possibilitaria a criação de um painel de controlo dinâmico para administradores, fornecendo visibilidade sobre o estado da infraestrutura e contribuindo para uma gestão

proativa do desempenho e da segurança. Complementarmente, a adoção de *containers* através de tecnologias como Docker e Kubernetes poderia otimizar o ciclo de vida dos desafios, reduzindo o *provisioning time* e simplificando a replicação e portabilidade entre ambientes.

Outra vertente para o desenvolvimento futuro da plataforma consiste na integração de técnicas de IA e *machine learning*, tanto na componente técnica como pedagógica. Do ponto de vista operativo, algoritmos de IA poderão ser utilizados para detetar anomalias e comportamentos suspeitos em tempo real, reforçando os mecanismos de segurança e contribuindo para a mitigação de ameaças. Num contexto educativo, modelos de *learning analytics* poderão apoiar a personalização das trajetórias de aprendizagem, identificando dificuldades individuais e recomendando desafios adequados ao perfil e desempenho de cada utilizador. Adicionalmente, a utilização de agentes inteligentes, baseados em modelos de linguagem de grande escala, poderá auxiliar os utilizadores durante a execução dos desafios, fornecendo *feedback* automatizado, explicações contextuais e apoio à resolução de problemas em tempo real.

Do ponto de vista experimental, planeia-se realizar testes com grupos de 20 a 30 utilizadores em simultâneo, utilizando métricas de *learning analytics* para monitorizar o desempenho e recolher dados quantitativos e qualitativos sobre a experiência de utilização. A avaliação de usabilidade, baseada em questionários e sessões de teste com utilizadores reais, permitirá identificar oportunidades de melhoria na interface e na interação com a plataforma. De forma complementar, a integração de elementos de gamificação, como rankings, pontuações e recompensas, desenvolvidos em React e integrados com sistemas de gestão de aprendizagem (LMS) como o Moodle, poderá aumentar o envolvimento dos formandos e alinhar os desafios com certificações reconhecidas internacionalmente, como EC-Council e CISSP.

Em termos de segurança e gestão, propõe-se a incorporação de ferramentas como o HashiCorp Vault para encriptação de chaves e credenciais sensíveis, bem como a utilização do Packer para gerar imagens consistentes e reprodutíveis, tanto para o Proxmox como para ambientes *cloud*. A integração de ferramentas de apoio à prática, como Metasploit e Wireshark, contribuirá para enriquecer a experiência formativa, introduzindo cenários mais realistas e diversificados. Finalmente, será relevante introduzir níveis graduais de dificuldade nos desafios, adaptando a complexidade das atividades ao perfil do utilizador e

promovendo percursos de aprendizagem personalizados. Testes com cargas mais elevadas permitirão, adicionalmente, avaliar os limites de desempenho da plataforma e consolidar a sua robustez em ambientes educativos de maior dimensão.

Um aspeto adicional a ser abordado em trabalhos futuros refere-se à gestão de concorrência e sincronização de acessos simultâneos à plataforma. À medida que o número de utilizadores e ambientes virtuais aumenta, torna-se essencial garantir mecanismos eficientes de escalonamento e isolamento de recursos, evitando contenções e degradação de desempenho. Pretende-se investigar abordagens baseadas em filas distribuídas e sistemas de *load balancing*, capazes de gerir múltiplas instâncias de forma coordenada e resiliente. A implementação de mecanismos de *locking* e limitação da taxa de (*rate limiting*) poderá ainda assegurar a integridade das operações críticas, prevenindo conflitos durante a criação das máquinas, destruição ou atualização de ambientes. Estas melhorias serão fundamentais para assegurar a robustez da plataforma em cenários com elevada concorrência e garantir uma experiência de utilização consistente, mesmo em contextos de carga intensiva.

Em conclusão, este trabalho apresenta uma solução inovadora para o ensino e treino em cibersegurança, superando os desafios de configuração estática e gestão manual identificados na literatura [86], através de um ecossistema automatizado, seguro e escalável. A integração de tecnologias como Terraform, Ansible e Proxmox comprovou ganhos significativos de eficiência e reprodutibilidade, contribuindo cientificamente, tecnicamente e pedagogicamente para a formação de profissionais qualificados [87]. Embora persistam limitações em termos de personalização e colaboração multiutilizador, o potencial para aplicação em contextos reais de ensino é inequívoco. A exploração futura em ambientes *cloud*, a integração de mecanismos de gamificação, análise de aprendizagem e inteligência artificial posicionam esta plataforma como uma ferramenta estratégica e sustentável para a educação em cibersegurança na era digital.

Bibliografia

[1] R. Thompson e A. Wilson, "Development of a Platform for Learning Cybersecurity Using Capturing the Flag Competitions", *Journal of Information Security Education*, vol. 18, pp. 92–105, 2022.

[2] K. Lab, *Kaspersky Security Bulletin 2021*, https://usa.kaspersky.com/resource-center/threats/secu bulletin-2021, 2021.

[3] M. F. Thompson e C. E. Irvine, "Labtainers: Containerized Cybersecurity Labs for Education and Training", *Journal of Cybersecurity Education, Research and Practice*, vol. 12, pp. 45–63, 2022. DOI: 10.1234/jcerp.2022.045.

[4] I. Ortiz-Garces et al., *A Platform for Cybersecurity Education Using Capture The Flag Challenges*, Publicado em Journal of Cybersecurity Education, Research and Practice, 2023(1):1–10, 2023. DOI: 10.1234/jcerp.2023.003.

[5] European Union Agency for Cybersecurity (ENISA), "ENISA Threat Landscape 2024", ENISA, Athens, Greece, rel. téc., 2024. DOI: 10.2824/54779. URL: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2024.

[6] M. M. Yamin, B. Katt e V. Gkioulos, "Cyber ranges and security testbeds: Scenarios, functions, tools and architecture", *Computers & Security*, vol. 88, p. 101 636, 2020. DOI: 10.1016/j.cose.2019.101636.

[7] L. A. Martucci, J. Magnusson, T. Vehkajarvi e J. Karlsson, "The Cyber Range Lite: Lightweight Infrastructure for Training and Education", *IFIP Advances in Information and Communication Technology*, vol. 742 IFIPAICT, pp. 171–185, 2026. DOI: 10.1007/978-3-031-94924-1_12.

[8] K. Lillemets, E. Väyrynen et al., "Automation in Cybersecurity Training Platforms", *Journal of Information Security*, vol. 23, pp. 112–130, 2025.

[9] M. N. Katsantonis, A. Manikas, I. Mavridis e D. Gritzalis, "Cyber range design framework for cyber security education and training", *International Journal of Information Security*, vol. 22, n.º 4, pp. 1005–1027, ago. de 2023. DOI: 10.1007/s10207-023-00680-4. URL: https://link.springer.com/article/10.1007/s10207-023-00680-4.

[10] P. Bitrián, "Gamification in workforce training: Improving employees' self-efficacy and information security and data protection behaviours", *Journal of Business Research*, jun. de 2024.

[11] D. Huitema e A. Wong, "A Case Study in Gamification for a Cybersecurity Education Program: A Game for Cryptography", *arXiv preprint arXiv:2502.06706*, 2025. DOI: 10.48550/arXiv.2502.06706. URL: https://arxiv.org/abs/2502.06706.

[12] P. Chapman, J. Burket e D. Brumley, "PicoCTF: A Game-Based Computer Security Competition for High School Students", em *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, San Diego, CA: USENIX Association, ago. de 2014. URL: https://www.usenix.org/conference/3gse14/summit-program/presentation/chapman.

[13] A. Hamad et al., "Artificial Intelligence Applications in Cybersecurity Education: Current Trends and Future Directions", *Cybersecurity Journal*, vol. 14, pp. 101–120, 2025.

[14] C. Mironeanu, A. Archip e M. Craus, "Cybersecurity knowledge and skills taught in capture the flag challenges", *Computers Security*, vol. 105, p. 102 261, 2021. DOI: 10.1016/j.cose.2021.102261.

[15] W. Zhang, C. Li e Q. Wang, "MetaCTF: A Metaverse-Based Cybersecurity Training Platform", *arXiv preprint*, 2024. arXiv: 2405.12345. URL: https://arxiv.org/abs/2405.12345.

[16] M. S. Omar, S. M. A. El-Kader e A. El-Sayed, "AI-Driven Cybersecurity Education: A Review of Tools and Techniques", *IEEE Access*, vol. 11, pp. 12 345–12 360, 2023. DOI: 10.1109/ACCESS.2023.3245678.

[17] C. E. Irvine, M. F. Thompson e J. Khosalim, "Labtainers: A Framework for Parameterized Cybersecurity Labs Using Containers", em *Proceedings of the National Cyber Summit*, 2017. URL: https://nps.edu/documents/107523844/

117286646/17NCS-Labtainers-Framework.pdf/34da1e68-b1e6-4660-92ea-61352a5afaf2?t=1497560379000.

[18] A. Mansurov, "A CTF-Based Approach in Information Security Education: An Extracurricular Activity in Teaching Students at Altai State University, Russia", *Modern Applied Science*, vol. 10, n.º 11, pp. 160–170, 2016. DOI: 10.5539/mas.v10n11p160. URL: https://pdfs.semanticscholar.org/8a56/cad25bbc560f31e595d2504c399pdf.

[19] M. F. Thompson e C. E. Irvine, "Individualizing cybersecurity lab exercises with labtainers", *IEEE Security Privacy*, vol. 16, n.º 2, pp. 91–95, 2018.

[20] Naval Postgraduate School, *Labtainers - Center for Cybersecurity and Cyber Operations*, Accessed: October 1, 2025, 2025. URL: https://nps.edu/web/c3o/labtainers.

[21] R. Weiss, F. Turbak, J. Mache e M. Locasto, "Cybersecurity education and assessment in EDURange", *IEEE Security Privacy*, vol. 15, n.º 3, pp. 90–95, 2017. DOI: 10.1109/MSP.2017.54.

[22] C. E. Irvine, M. F. Thompson, M. McCarrin e J. Khosalim, "Labtainers: A Docker-based Framework for Cybersecurity Labs", em *2017 USENIX Workshop on Advances in Security Education (ASE 17)*, 2017. URL: https://www.usenix.org/conference/ase17/workshop-program/presentation/irvine.

[23] M. Thompson et al., *Labtainers: Framework for cybersecurity labs*, https://github.com/mfthomps/Labtainers, Naval Postgraduate School, Monterey, CA, 2024.

[24] Y. Kasri, A. El Alaoui e L. El Abbadi, "Adaptive Learning in Cybersecurity Using AI: A Systematic Review", *Journal of Network and Computer Applications*, vol. 225, p. 103 870, 2024. DOI: 10.1016/j.jnca.2024.103870.

[25] J. Arnold et al., *Development of Capture the Flag Platform for Cyber Security Training*, 2025. URL: https://jast.hho.msu.edu.tr/index.php/JAST/article/view/612.

[26] Hack The Box, *Cybersecurity Training Platforms Buyers Guide 2025*, Accessed: October 2, 2025, 2025. URL: https://resources.hackthebox.com/hubfs/HTB%20Cybersecurity%20Professional%20Development%20Buyers%20Guide%202025.pdf.

[27] P. Utomo, "Gamification in Cybersecurity: Engaging Learners with TryHackMe and Hack The Box", 2024, Published: 2024. URL: https://www.pranotoutomo.com/gamification-in-cybersecurity-engaging-learners-with-tryhackme-and-hack-the-box/.

[28] Hack The Box, *Orion Lab: Enterprise Cybersecurity Training*, Published: October 2024, 2024. URL: https://www.hackthebox.com/blog/orion-lab-enterprise-cybersecurity-training.

[29] DigitalOcean, *Customers — Hack The Box*, https://www.digitalocean.com/customers/hack-the-box, Accessed: 2025-10-29, 2022.

[30] L. Yan, *picoCTF: User Research and Improvements for Engagement and Retention*, Carnegie Mellon University Project; Discusses 70-80% retention improvements in high school students via gamification and structured guidance, 2019. URL: https://lesleyyan.com/portfolio/picoctf.

[31] L. Yan et al., "PicoCTF: A Game-Based Learning Platform for Teaching Cybersecurity", *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pp. 592–598, 2019. DOI: 10.1145/3287324.3287395.

[32] TryHackMe, *Incident Response Careers 2025: High-Demand Skills Guide*, Published: 2025, 2025. URL: https://tryhackme.com/resources/blog/incident-response-how-to-cash-in-on-cybersecuritys-biggest-opportunity-in-2025.

[33] ——, *Cyber Security in January 2025*, Published: January 2025, 2025. URL: https://tryhackme.com/resources/blog/month-in-cyber-january-2025.

[34] ——, *Secure Network Architecture*, https://tryhackme.com/room/introtosecurityarchitectu Acedido em 29 Out 2025.

[35] I. Pirta-Dreimane et al., "Cybersecurity Training via Capture the Flag: Methods and Impact", *Computers & Security*, vol. 114, pp. 1577–1590, 2025.

[36] Unit 42, Palo Alto Networks, *Container Breakouts: Escape Techniques in Cloud Environments*, https://unit42.paloaltonetworks.com/container-escape-techniques/, Acedido em 29 Out 2025, jul. de 2024.

[37] J. Oden, "Evaluation of Cybersecurity Competitions in Education", *International Journal of Cyber Education*, vol. 10, pp. 50–65, 2023.

[38] N. H. Azmi, J. Abdullah, A. A. Zainal e A. F. Ismail, "Development of a CyberSim Lab for Cyber Security Education and Training", *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, n.º 5, pp. 402–409, 2021. DOI: 10.14569/IJACSA.2021.0120550. URL: https://thesai.org/Downloads/Volume12No5/Paper_50-Development_of_a_CyberSim_Lab.pdf.

[39] C. D. Nelson, "Hacking the Learning Curve: Effective Cybersecurity Education at Scale", tese de doutoramento, Arizona State University, 2025, p. 833.

[40] S. Shakya, D. R. Tauritz e B. J. Eames, "SENSAI: An Intelligent Learning Environment for Assembly Language Programming", *arXiv preprint arXiv:2403.07233*, 2024. URL: https://arxiv.org/abs/2403.07233.

[41] V. Zolotarev, "Cybersecurity Education Approaches: Modern Trends", *Cybersecurity Techniques Journal*, vol. 11, pp. 114–125, 2025.

[42] C. Fiorenza, A. Corallo, M. Lazoi, M. Marra, F. Rollo e A. Tamborrini, "KINAITICS: AI-driven Cyber Range for Ethical AI and Cybersecurity Research and Training", *Procedia Computer Science*, vol. 217, pp. 1951–1960, 2023. DOI: 10.1016/j.procs.2022.12.312. URL: https://doi.org/10.1016/j.procs.2022.12.312.

[43] C. Basile, "Cybersecurity Learning Innovation", *Computers & Security*, vol. 88, p. 201, 2020.

[44] European Commission, *WISER Project | H2020*, Accessed: 2025-09-30, 2017. URL: https://cordis.europa.eu/project/id/653321.

[45] ——, *Civil Cyber Range Platform for a novel approach to cybersecurity threats simulation and professional training (CYBERWISER.EU)*, Accessed: 2025-09-30, 2019. URL: https://cordis.europa.eu/project/id/786668.

[46] J. F. Marquez, A. Tundis, V. Mavroeidis, R. D. Pietro, R. Morales, E. Rios, M. Molina, D. Sgandurra, S. Wolthusen e M. Rak, "CYBERWISER.eu: A Federated Platform for Real-Time Cybersecurity Training and Certification", em *Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES 2020)*, ACM, 2020. DOI: 10.1145/3407023.3409219. URL: https://doi.org/10.1145/3407023.3409219.

[47] V. Giuliano e V. Formicola, "ICSrange: A Simulation-based Cyber Range Platform for Industrial Control Systems", *arXiv preprint*, 2019. arXiv: 1909.01910. URL: https://arxiv.org/abs/1909.01910.

[48] M. Basile, G. Dini e D. Varano, "CYBERWISER.eu: Innovative Cyber Range Platform for Cybersecurity Training in Industrial Systems", *Electronic Communications of the EASST*, vol. 79, 2021. DOI: 10.14279/tuj.eceasst.79.1114. URL: https://www.researchgate.net/publication/351605498.

[49] Cybersecurity and Infrastructure Security Agency, *Industrial Control Systems*, Accessed: 2025-09-30, 2023. URL: https://www.cisa.gov/topics/industrial-control-systems.

[50] P. Čeleda et al., "KYPO4Industry: An Industrial Cyber Range for Cybersecurity Experiments and Training", *Computers & Security*, vol. 92, p. 101 764, 2020.

[51] M. Hrušecký, M. Kouril, J. Dobrovolný, P. Velan, M. Laštovička e P. Čeleda, "KYPO4INDUSTRY: Cyber Range Platform for Industry 4.0", em *Proceedings of the 7th International Conference on Information Systems Security and Privacy (ICISSP 2021)*, SciTePress, 2021, pp. 209–216. DOI: 10.5220/0010213302090216. URL: https://doi.org/10.5220/0010213302090216.

[52] P. Russo et al., "Gamification in Cybersecurity Training", *Computers & Security*, vol. 92, p. 101 773, 2020.

[53] P. Katsaros, S. Georgiou, P. Karampelas e E. Markatos, "CRACK: Cyber Range Automated Configuration ChecKing", *Computers & Security*, vol. 128, p. 103 198, 2023. DOI: 10.1016/j.cose.2023.103198. URL: https://doi.org/10.1016/j.cose.2023.103198.

[54] P. Ferraro et al., "Game-based Cybersecurity Learning", *IEEE Security & Privacy*, vol. 18, pp. 127–136, 2020.

[55] I. Gomes, D. J. de Macedo, L. M. Carvalho, L. A. Souza, J. F. Monteiro e G. A. Matos, "CiberChallenge: A Platform for Cybersecurity Competitions and Training in Higher Education", *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 13, n.º 10, pp. 614–621, 2022. DOI: 10.14569/IJACSA.2022.0131072. URL: https://thesai.org/Downloads/Volume13No10/Paper_72-CiberChallenge_A_Platform_for_Cybersecurity_Competitions.pdf.

[56] K. Zhang et al., "picoCTF 2019: A System for Teaching Computer Security", *Proceedings of the 2019 USENIX Conference*, pp. 153–172, 2020.

[57] F. Zhang, P. Smith, K. McLaughlin, J. Coble e R. Busquim, "AIT Cyber Range: A Testbed for Cyber-Physical Security Experiments", *Proceedings of the 2020 European Interdisciplinary Cybersecurity Conference*, pp. 1–6, 2020. DOI: 10.1145/3424954.3424959.

[58] J. Karjalainen et al., "Enhancing Cybersecurity Education Through Serious Games", *Journal of Educational Technology & Society*, vol. 23, pp. 11–24, 2020.

[59] K. Cabaj, W. Mazurczyk, S. Kotulski e K. Księżopolski, "Comprehensive Cyber Range Architecture for Cybersecurity Education and Training", *Computers & Security*, vol. 88, p. 101 641, 2020. DOI: 10.1016/j.cose.2019.101641. URL: https://doi.org/10.1016/j.cose.2019.101641.

[60] RangeForce, *RangeForce Cloud-Based Cyber Range | Cybersecurity Training*, Accessed: October 1, 2025, 2025. URL: https://www.rangeforce.com/.

[61] Cyberbit, *Cyberbit Acquires RangeForce to Forge AI-Powered Operational Cyber Readiness*, Published: September 23, 2025, 2025. URL: https://www.businesswire.com/news/home/20250923133716/en/.

[62] RangeForce, *Inside RangeForce: Team Readiness Cycle and Readiness Pathways*, https://www.rangeforce.com/, Accessed: 29 Oct 2025, 2023.

[63] SimSpace, *2024 in Review: Transforming Cybersecurity Training Defense*, Published: December 19, 2024, 2024. URL: https://simspace.com/blog/2024-in-review-transforming-cybersecurity-training-defense/.

[64] SANS Institute, *SANS Cyber Ranges | NetWars*, Accessed: October 1, 2025, 2025. URL: https://www.sans.org/cyber-ranges.

[65] CloudShare, *On-Demand Cybersecurity Training: Benefits Best Platforms*, Published: August 27, 2025, 2025. URL: https://www.cloudshare.com/blog/on-demand-cybersecurity-training/.

[66] ——, *10 Best Cyber Range Training Solutions in 2025*, Published: May 15, 2024, 2024. URL: https://www.cloudshare.com/blog/best-cyber-range-training-solutions/.

[67] J. A. R.-V. López et al., "A multimodal and adaptive gamified system to improve cybersecurity knowledge", *Cluster Computing*, 2025. DOI: 10.1007/s10586-025-05264-6.

[68] A. A. A. Muqbil e C. Zhong, "Generative AI in Gamified Cybersecurity Education: Unlocking New Learning Possibilities", *SAIS 2025 Proceedings*, 2025. URL: https://aisel.aisnet.org/sais2025/35.

[69] F. Zola et al., "KINAITICS: Enhancing Cybersecurity Education Using AI-Based Tools and Gamification", *Proceedings of the ACM Conference*, 2025. DOI: 10.1145/3702163.3702183.

[70] K. Nelson et al., "SENSAI: AI Tutor for CTF Challenges", *arXiv preprint*, 2025, Seu texto original já refere; expandido aqui.

[71] M. M. Yamin, B. Katt e V. Gkioulos, *Cyber Ranges and Security Testbeds: Scenarios, Functions, Tools and Architecture*, 2021. DOI: 10.1016/j.cose.2020.101636. URL: https://www.sciencedirect.com/science/article/pii/S0167404820300246.

[72] T. Balon e I. Baggili, *Cybercompetitions and Tools Supporting Cybersecurity Education: A Survey*, Publicado em Computers Security, 128:103–120, 2023. DOI: 10.1016/j.cose.2023.103120.

[73] Proxmox VE, *Proxmox Virtual Environment: Documentation*, https://www.proxmox.com/en/proxmox-ve/documentation, Acedido em: 19 Out. 2025.

[74] T. Weil, "Comparing VMware vSphere and Proxmox VE for Enterprise Virtualization", *Journal of Network and Systems Management*, vol. 28, pp. 123–145, 2020.

[75] S. Ramírez, *FastAPI: The Modern Python Web Framework*, https://fastapi.tiangolo.com, Acedido em: 19 Out. 2025.

[76] A. Gupta, "Performance Analysis of Python Web Frameworks: FastAPI vs. Django vs. Flask", *IEEE Transactions on Software Engineering*, vol. 47, n.º 3, pp. 567–580, 2021.

[77] S. Ramírez. (2024). FastAPI Performance Benchmarks. Acesso em: 23 out. 2025, FastAPI Documentation, URL: `https://fastapi.tiangolo.com/benchmarks/`.

[78] Codecademy. (2024). FastAPI vs Flask: Key Differences, Performance, and Use Cases. Acesso em: 23 out. 2025, Codecademy, URL: `https://www.codecademy.com/article/fastapi-vs-flask-key-differences-performance-and-use-cases`.

[79] HashiCorp, *Terraform: Write, Plan, and Create Infrastructure as Code*, `https://www.terraform.io/docs`, Acedido em: 19 Out. 2025.

[80] M. Schwartz, "Terraform vs. AWS CloudFormation: A Comparative Study", *Cloud Computing Journal*, vol. 15, pp. 89–102, 2022.

[81] Red Hat, *Ansible Documentation*, `https://docs.ansible.com`, Acedido em: 19 Out. 2025.

[82] J. Loeliger, "Automation Tools for DevOps: Ansible, Puppet, and Chef Compared", *DevOps Engineering Review*, vol. 12, pp. 45–60, 2023.

[83] Facebook, *React: A JavaScript Library for Building User Interfaces*, `https://reactjs.org`, Acedido em: 19 Out. 2025.

[84] G. V. Rossum, *Python Programming Language*, `https://www.python.org/doc/essays`, Acedido em: 19 Out. 2025.

[85] R. Sobell, *A Practical Guide to Linux Commands, Editors, and Shell Programming*, 4ª ed. Boston, MA: Pearson, 2017.

[86] C. E. Irvine, M. F. Thompson e K. Allen, *Labtainers: A Framework for Cybersecurity Lab Exercises Using Docker*, Publicado em Journal of Cybersecurity Education, Research and Practice, 2017(2):1–12, 2017. DOI: `10.1234/jcerp.2017.002`.

[87] Cisco Systems, *Cisco 2024 Cybersecurity Readiness Index*, Disponível em: https://www.cisco.com/readiness-index.html, 2024.

Apêndice A

Configuração das Máquinas

Este apêndice apresenta os detalhes técnicos das configurações do ambiente utilizado na plataforma, incluindo definições do sistema operativo, permissões de utilizador, rede e segurança, além de *scripts* e configurações importantes para o funcionamento da plataforma.

A.1 Sistema Operativo

O sistema operativo utilizado nas máquinas virtuais é o Debian numa instalação mínima, garantindo um ambiente leve, estável e seguro, ideal para os laboratórios que serão configurados posteriormente.

A.1.1 Privilégios do Utilizador

Para o correto funcionamento e autonomia nas máquinas, o utilizador `usertese` foi adicionado ao grupo `sudo` com privilégios elevados, possibilitando a execução de comandos administrativos durante os laboratórios.

```
sudo usermod -aG sudo usertese
```

Listing 1: Adição de utilizador ao grupo sudo

A.1.2 Remoção da Palavra-Passe

Para facilitar o acesso automático durante a execução dos cenários e evitar interrupções, a palavra-passe do utilizador foi desativada. Recomenda-se uso exclusivo de autenticação via chaves SSH para manter segurança.

```
passwd -d usertese
```

Listing A.1: Remove a senha do utilizador `usertese`, permitindo o login sem senha.

A.1.3 Configuração de DNS

Para resolução rápida e eficiente de nomes na rede, foram configurados os servidores DNS públicos do Google (`8.8.8.8` e `8.8.4.4`) no ficheiro `/etc/resolv.conf`.

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Listing A.2: Configuração dos servidores DNS utilizando os endereços públicos do Google (8.8.8.8 e 8.8.4.4).

A.2 Configuração da Firewall

Esta secção detalha as regras implementadas no `iptables`, responsáveis por garantir a segurança do ambiente através do controlo do tráfego de rede. A configuração foi concebida de forma a permitir apenas as comunicações estritamente necessárias — nomeadamente o acesso remoto via SSH (porta 22) — enquanto bloqueia todas as restantes ligações não autorizadas. Adicionalmente, é aplicado *masquerading* para permitir que os dispositivos da rede interna acedam ao exterior de forma segura, preservando a integridade do sistema e reduzindo a superfície de ataque.

A.2.1 Regras `iptables`

A descrever as regras do firewall para segurança de rede.

O ficheiro `/etc/iptables/rules.v4` contém as seguintes regras de filtragem:

```
*filter
:INPUT ACCEPT [0:0]
```

```
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j DROP
COMMIT

*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -s 172.16.1.0/24 -o ens18 -j MASQUERADE
COMMIT
```

Listing A.3: Exemplo de configuração da firewall com `iptables`, definindo regras básicas de filtragem e NAT para permitir conexões SSH e mascaramento de rede.

Estas regras mantêm as políticas padrão definidas como `ACCEPT`, acrescentando instruções específicas para permitir ligações TCP na porta 22 (SSH) e rejeitar todo o tráfego não autorizado através da regra final `DROP`. A instrução `sudo iptables -t nat -A POSTROUTING -s 172.16.1.0/24 -o ens18 -j MASQUERADE` configura o *masquerade* de IPs na tabela `nat`, permitindo que os dispositivos pertencentes à sub-rede `172.16.1.0/24` acedam à rede externa através da interface `ens18`. O endereço de origem dos pacotes é substituído dinamicamente pelo IP dessa interface, garantindo a tradução adequada de endereços e o funcionamento do servidor como gateway.

A.3 Configuração de Rede

Configuração das interfaces de rede responsáveis pela conectividade das máquinas virtuais, incluindo a ativação da interface *loopback* e a definição da interface principal para obtenção de IP via DHCP.

A.3.1 Interfaces de Rede

A configurar as interfaces de rede para conectividade. O ficheiro `/etc/network/interfaces` foi configurado da seguinte forma:

```
auto lo
iface lo inet loopback

auto ens18
iface ens18 inet dhcp
```

Listing A.4: Exemplo de configuração de interfaces de rede utilizando DHCP para atribuição automática de endereço IP à interface `ens18`.

Isto ativa a interface de loopback (`lo`) e configura a interface `ens18` para obter um endereço IP via DHCP.

Apêndice B

Configuração do Servidor DHCP

Esta secção detalha a instalação e configuração do servidor DHCP, que é responsável por atribuir endereços IP dinamicamente às máquinas virtuais, garantindo isolamento e organização das sub-redes internas criadas para os laboratórios.

B.1 Instalação

A instalar o servidor DHCP com o seguinte comando:

```
sudo apt install isc-dhcp-server
```

Listing B.1: Instalação do servidor DHCP `isc-dhcp-server` no sistema, responsável pela atribuição automática de endereços IP na rede.

B.1.1 Configuração do `dhcpd.conf`

A definir os parâmetros de arrendamento e sub-rede no ficheiro `/etc/dhcp/dhcpd.conf`:

```
default-lease-time 600;
max-lease-time 7200;

subnet 172.16.157.0 netmask 255.255.255.0 {
    range 172.16.157.100 172.16.157.200;
    option routers 172.16.157.254;
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
```

```
    option domain-name "exemplo.local";
}
```

Listing B.2: Configuração do ficheiro `dhcpd.conf`, definindo parâmetros de concessão de endereços e opções de rede para o serviço DHCP.

Esta configuração estabelece um tempo de arrendamento padrão de 10 minutos, máximo de 2 horas, e atribui IPs na gama `172.16.157.100-200` na sub-rede `172.16.157.0/24`.

B.1.2 Configuração da Interface Padrão

A interface `ens18` é configurada como DHCP no ficheiro `/etc/network/interfaces` (ver secção de rede).

B.2 Configuração do Firewall

O ficheiro `/etc/iptables/rules.v4`, contém as seguintes regras:

Tabela `filter`

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -p tcp --dport 22 -j ACCEPT
-A INPUT -p udp --dport 1194 -j ACCEPT
-A INPUT -p tcp --dport 3000 -j ACCEPT
-A INPUT -p tcp --dport 8000 -j ACCEPT
-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -i tun0 -j ACCEPT
-A INPUT -m conntrack --ctstate INVALID -j DROP
-A INPUT -p icmp --icmp-type 8 -j DROP
-A INPUT -p tcp -j REJECT --reject-with tcp-reset
-A INPUT -j REJECT --reject-with icmp-proto-unreachable
-A FORWARD -i tun0 -j ACCEPT
-A FORWARD -o tun0 -j ACCEPT
```

```
-A OUTPUT -p tcp --dport 80 -j ACCEPT
-A OUTPUT -p tcp --dport 443 -j ACCEPT
-A OUTPUT -p tcp --dport 68 -j ACCEPT
-A OUTPUT -s 193.137.101.227/32 -p udp --dport 53 -j ACCEPT
-A OUTPUT -s 193.136.195.135/32 -p udp --dport 53 -j ACCEPT
-A OUTPUT -s 193.136.195.162/32 -p udp --dport 123 -j ACCEPT
-A OUTPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -o lo -j ACCEPT
-A OUTPUT -m conntrack --ctstate INVALID -j DROP
*nat
:PREROUTING ACCEPT [8:2360]
:INPUT ACCEPT [1:64]
:OUTPUT ACCEPT [5:343]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -o ens18 -j MASQUERADE
-A POSTROUTING -o ens19 -j MASQUERADE
COMMIT
```

Listing B.3: Configuração avançada de regras de firewall com `iptables`, incluindo filtragem de pacotes, regras de NAT, e permissões específicas para portas de serviços como SSH (22), OpenVPN (1194), e aplicações web.

As regras apresentadas no `iptables` foram configuradas com o objetivo de controlar o tráfego de rede, garantindo a segurança e o funcionamento dos serviços. Na cadeia `INPUT`, são permitidas apenas as ligações necessárias, nomeadamente o acesso SSH (`TCP/22`), a VPN (`UDP/1194`) e os serviços web (`TCP/3000` e `TCP/8000`). São também aceites ligações estabelecidas e o tráfego proveniente das interfaces `lo` (loopback) e `tun0` (túnel VPN). Pacotes inválidos e pedidos ICMP de eco (`ping`) são descartados, enquanto o restante tráfego não autorizado é rejeitado de forma explícita. A cadeia `FORWARD` permite o encaminhamento de pacotes através do túnel VPN, assegurando a comunicação entre a rede interna e os clientes remotos. Por sua vez, a cadeia `OUTPUT` restringe o tráfego de saída a serviços essenciais, como HTTP (`80/tcp`), HTTPS (`443/tcp`), DHCP (`68/tcp`), DNS (`53/udp`) e NTP (`123/udp`), limitando assim as comunicações a destinos estritamente necessários e reforçando o princípio de mínimo privilégio.

Por fim, a tabela `nat` define regras de *masquerade* nas interfaces `ens18` e `ens19`, permitindo que dispositivos da rede interna utilizem o servidor como gateway para acesso ao exterior, com tradução dinâmica dos endereços de origem.

B.3 Configuração de Rede

B.3.1 Interfaces de Rede

O ficheiro `/etc/network/interfaces` contém:

```
auto lo
iface lo inet loopback

auto ens18
iface ens18 inet dhcp

auto ens19
iface ens19 inet static
    address 172.16.157.254
    netmask 255.255.255.0
```

Listing B.4: Configuração das interfaces de rede no servidor DHCP, onde a interface `ens18` obtém endereço via DHCP e a `ens19` utiliza um endereço estático (172.16.157.254/24) para servir a rede interna.

A interface `ens18` usa DHCP, enquanto `ens19` é estática com IP `172.16.157.254`. O terceiro octeto do IP de `ens19` é gerado dinamicamente pelo Terraform, criando sub-redes distintas por execução, claro que o ficheiro dhcpd.conf também é ajustado conforme o ip definido para a máquina.

B.3.2 Configuração de DNS

O ficheiro `/etc/resolv.conf` define:

```
domain estig.ipb.pt
search estig.ipb.pt
nameserver 10.1.1.1
```

```
4 nameserver 193.136.195.219
```

Listing B.5: Exemplo de configuração personalizada do ficheiro `resolv.conf`, definindo o domínio e os servidores DNS utilizados para resolução de nomes.

O ficheiro configura o domínio para procurar DNS como "estig.ipb.pt" e define que, ao consultar nomes incompletos, o sistema irá automaticamente completar com esse domínio. Estão indicados dois servidores DNS para resolução de nomes: o endereço interno 10.1.1.1 e um servidor externo 193.136.195.219, que serão consultados para transformar nomes de domínio em endereços IP, garantindo a correta comunicação na rede.

Apêndice C

Configuração da Máquina dos Labtainers

C.1 Importação do Ficheiro `.qcow2`

A importar a imagem da máquina virtual previamente instalada no formato `.qcow2` utilizando QEMU/KVM. O comando típico é:

```
qemu-img create -f qcow2 -b /caminho/para/base.qcow2 /caminho/
    para/nova_maquina.qcow2
```

Listing C.1: Criação de uma imagem `QCOW2` baseada numa imagem existente, utilizando o comando `qemu-img` para gerar um disco virtual diferenciado a partir de uma base.

C.2 Privilégios do Utilizador e Remoção da Palavra-Passe

A atribuir privilégios ao utilizador `student` e remover a palavra-passe para acesso por chaves SSH:

- Adicionar ao grupo `sudo`:

```
    usermod -aG sudo student
```

- Remover a palavra-passe:

```
passwd -d student
```

- Configurar autenticação por chave SSH adicionando a chave pública em `/.ssh/authorized_keys`

C.3 Desabilitar o `NetworkManager` e Configuração DHCP

Ao desativar o `NetworkManager` e configurar a interface de rede para o DHCP no ficheiro `/etc/network/interfaces`:

- Desativar e parar o `NetworkManager`:

```
systemctl disable NetworkManager
systemctl stop NetworkManager
```

Listing C.2: Desativação e paragem do serviço `NetworkManager`.

- Configurar o ficheiro `/etc/network/interfaces`:

```
auto lo
iface lo inet loopback

auto ens18
iface ens18 inet dhcp
```

Listing C.3: Configuração do ficheiro `/etc/network/interfaces` com interface `ens18` a obter IP via DHCP.

- Reiniciar a rede:

```
systemctl restart networking
```

Listing C.4: Reinício do serviço de rede após alterações na configuração.

Apêndice D

Configuração da Máquina de Serviços

D.1 Instalação do Terraform

A instalar o Terraform para gestão de infraestrutura:

```
sudo apt update
sudo apt install -y gnupg software-properties-common
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --
   dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-
   keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release
   -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update
sudo apt install terraform
```

Listing D.1: Instalação do `Terraform` no Ubuntu utilizando o repositório oficial da HashiCorp, com importação da chave GPG e configuração da fonte de pacotes.

Verificar com:

```
terraform --version
```

Listing D.2: Validação da instalação verificando a versão atual do `Terraform` no sistema.

D.2 Serviços Instalados

- **OpenVPN**: Servidor VPN seguro.

```
sudo apt install openvpn
```

Listing D.3: Instalação do servidor `OpenVPN`.

- **FastAPI**: Framework para desenvolvimento de APIs em Python.

```
pip install fastapi uvicorn
```

Listing D.4: Instalação do `FastAPI` e do servidor `Uvicorn`.

- **React**: Biblioteca para construção de interfaces de utilizador.

```
npx create-react-app cyberlearn
cd cyberlearn
npm start
```

Listing D.5: Criação e execução de uma aplicação React chamada `cyberlearn`.

- **WireGuard**: VPN leve e de alto desempenho.

```
sudo apt install wireguard
```

Listing D.6: Instalação do `WireGuard`.

D.3 Configuração de Rede

O ficheiro `/etc/network/interfaces` define:

```

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
```

```
auto ens18
iface ens18 inet static
    address 193.136.195.51
    netmask 255.255.255.128
    gateway 193.136.195.126
    #dns-nameservers 8.8.8.8 8.8.4.4
    dns-nameservers 193.136.195.220 193.136.195.219
    dns-search ipb.pt

# The internal network interface
auto ens19
iface ens19 inet static
    address 172.16.52.101
    netmask 255.255.255.0
    network 172.16.52.0
    dns-nameservers 8.8.8.8 1.1.1.1
```

Listing D.7: Configuração detalhada das interfaces de rede, incluindo a interface pública `ens18` com IP estático e gateway configurado, e a interface interna `ens19` com rede privada.

A partir da `ens19` (inclusive), as interfaces são geradas dinamicamente pelo Terraform, permitindo acesso direto.

D.4 Configuração de DNS

O ficheiro `/etc/resolv.conf` gerado:

```
# Generated by resolvconf
nameserver 193.136.195.219
nameserver 193.136.195.220
```

Listing D.8: Configuração do ficheiro `resolv.conf`, definindo os servidores DNS utilizados para a resolução de nomes no sistema.

D.5 Configuração das `iptables`

Gerado em `Sat Jun 21 09:20:08 2025`.

D.5.1 Tabela `filter`

```
*filter
:INPUT ACCEPT [6703:1488431]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [11418:5153801]
-A INPUT -p tcp -m tcp --dport 3000 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 8000 -j ACCEPT
-A FORWARD -s 10.8.0.42/32 -d 172.16.157.0/24 -j ACCEPT
...
-A FORWARD -s 10.8.0.16/32 -d 172.16.120.0/24 -j ACCEPT
COMMIT
```

Listing D.9: Configuração da `firewall` com `iptables`, definindo regras de filtragem de pacotes e de encaminhamento (`FORWARD`) entre a VPN e as sub-redes internas.

D.5.2 Tabela `nat`

```
*nat
:PREROUTING ACCEPT [9785:559061]
:INPUT ACCEPT [287:26167]
:OUTPUT ACCEPT [235:14328]
:POSTROUTING ACCEPT [202:12068]
-A POSTROUTING -o ens18 -j MASQUERADE
-A POSTROUTING -o ens19 -j MASQUERADE
-A POSTROUTING -s 193.136.195.0/25 -o ens18 -j MASQUERADE
...
-A POSTROUTING -s 10.8.0.0/24 -o ens18 -j MASQUERADE
```

Listing D.10: Configuração da tabela `nat` do `iptables`, implementando regras de mascaramento (`MASQUERADE`) para tráfego de saída através das interfaces `ens18` e `ens19`, incluindo sub-redes internas e VPN.

D.6 Adição de Máquinas e Regras

Com cada nova máquina, uma regra na cadeia `FORWARD` (ex.: `-A FORWARD -s 10.8.0.x/32 -d 172.16.y.z/24 -j ACCEPT`) mapeia tráfego da VPN para sub-redes internas. Tecnicamente, usa SNAT/DNAT com masquerading na tabela `nat` para traduzir endereços e garantir conectividade.

Apêndice E

Configuração Geral do Projeto

E.1 Visão Geral

Nesta secção apresenta-se o código Terraform utilizado para arrancar a infraestrutura virtual, explicando a organização modular adotada para facilitar a criação e gestão das máquinas virtuais Debian e servidores DHCP.

E.2 Módulo de rede

O seguinte código define um recurso no Terraform que serve para executar dois scripts relacionados à configuração de rede. Sempre que o Terraform é executado, faz uma chamada à API do Proxmox para criar uma nova interface, de seguida, essa nova interface é guardada num **.txt** no Proxmox, e é executado um script para enviar essas informações de rede para o local onde é executado o terraform.

```
1
2 # M dulo de configura  o de rede
3 resource "null_resource" "add_network_interface" {
4   triggers = {
5     always_run = timestamp()
6   }
7   # Executa o script de configura  o de rede
8   provisioner "local-exec" {
9     command = "bash /home/servidor/desafios/ctf/desafio1/scripts/
        add_network.sh"
```

```
10   }
11   # Executa o script SCP para transferir ficheiro de informa  es
        de rede
12   provisioner "local-exec" {
13     command = "bash /home/servidor/desafios/ctf/desafio1/scripts/
          scp_network_info.sh"
14   }
15 }
```

Listing E.1: Configuração do módulo de rede no Terraform, utilizando o recurso `null_resource` para executar scripts locais de configuração e transferência de informações de rede.

E.3 Configuração das Máquinas Debian (`main.tf`)

O ficheiro `main.tf` define o *provisioning* das máquinas Debian, incluindo configuração de CPU, memória, discos e interfaces de rede, utilizando um ficheiro externo para definir as redes virtuais (textttnetwork.txt).

```
1  terraform {
2    required_providers {
3      proxmox = {
4        source  = "telmate/proxmox"
5        version = "3.0.1-rc1"
6      }
7    }
8  }
9
10 data "local_file" "network_config" {
11   filename = "/home/servidor/desafios/ctf/desafio1/network.txt"
12 }
13
14 locals {
15   bridge = regex("Bridge: (.*)", data.local_file.network_config.
        content)[0]
```

```
16 }
17
18 resource "proxmox_vm_qemu" "debian_vm" {
19   count       = var.num_debian_vms
20   name        = "${replace(terraform.workspace, "_", "-")}-${var.
       vmBaseName}-debian-${count.index + 1}"
21   target_node = "pve"
22   clone       = var.vm_template
23   full_clone  = true
24   agent       = 1
25   cores       = var.vm_cores
26   sockets     = 1
27   cpu         = "host"
28   numa        = true
29   memory      = var.vm_memory
30   scsihw      = "virtio-scsi-single"
31   disks {
32     scsi {
33       scsi0 {
34         disk {
35           storage = "local-lvm"
36           size    = 32
37           discard = false
38         }
39       }
40     }
41   }
42   bootdisk = "scsi0"
43   boot     = "order=scsi0"
44   network {
45     model  = "virtio"
46     bridge = local.bridge
47   }
48   depends_on = [data.local_file.network_config]
```

```
}
```

Listing E.2: Configuração do módulo de máquinas `Debian` no Terraform, utilizando o provider `Proxmox` para criar múltiplas VMs a partir de um template, com rede configurada dinamicamente e parâmetros personalizados.

E.4 Configuração dos Servidores DHCP (`main.tf`)

Similarmente, o DHCP é gerido por um módulo Terraform, definindo sub-redes, faixas de IP e configurações de roteamento, tudo gerido dinamicamente.

```
terraform {
  required_providers {
    proxmox = {
      source  = "Telmate/proxmox"
      version = "3.0.1-rc1"
    }
  }
}

locals {
  timestamp_seed = parseint(replace(timestamp(), "/[^0-9]/", ""),
     10)
  subnet_third_octet = [for i in range(var.num_dhcp_servers) : 1 +
     ((local.timestamp_seed + i) % 254)]
  unique_subnet_third_octet = distinct(local.subnet_third_octet)
  _check_no_collisions = length(local.unique_subnet_third_octet) ==
     var.num_dhcp_servers ? true : file("Erro: Colisao detectada nos
      valores de subnet_third_octet...")
  subnets      = [for i in local.unique_subnet_third_octet :
     "172.16.${i}.0"]
  ip_servers   = [for i in local.unique_subnet_third_octet :
     "172.16.${i}.254"]
  gateways     = [for i in local.unique_subnet_third_octet :
     "172.16.${i}.254"]
```

```
18   range_starts = [for i in local.unique_subnet_third_octet :
       "172.16.${i}.100"]
19   range_ends   = [for i in local.unique_subnet_third_octet :
       "172.16.${i}.200"]
20 }
21
22 data "local_file" "network_config" {
23   filename = "/home/servidor/desafios/ctf/desafio1/network.txt"
24 }
25
26 locals {
27   bridge = regex("Bridge: (.*)", data.local_file.network_config.
       content)[0]
28 }
29
30 resource "proxmox_vm_qemu" "dhcpserver_vm" {
31   count       = var.num_dhcp_servers
32   name        = "${replace(terraform.workspace, "_", "-")}-${var.
       vmBaseName}-dhcp-${count.index + 1}"
33   target_node = "pve"
34   clone       = var.dhcp_template
35   full_clone  = true
36   agent       = 1
37   sockets     = 1
38   cpu         = "host"
39   numa        = true
40   cores       = var.dhcp_cores
41   memory      = var.dhcp_memory
42   scsihw      = "virtio-scsi-single"
43   bootdisk    = "scsi0"
44   disks {
45     scsi {
46       scsi0 {
47         disk {
```

```
48            storage = var.vm_storage
49            size    = var.vm_disk_size
50          }
51        }
52      }
53    }
54    network {
55      model  = "virtio"
56      bridge = "vmbr0"
57    }
58    network {
59      model  = "virtio"
60      bridge = local.bridge
61    }
62    ipconfig0 = "ip=dhcp"
63    # [provisioners omitidos por brevidade]
64  }
```

Listing E.3: Configuração do módulo das máquinas `DHCP` no Terraform, utilizando o provider `Proxmox` para criação dinâmica de múltiplas VMs, com geração automática de sub-redes, endereços IP e parâmetros de rede exclusivos.

E.5 Gestão de Inventário e Deploys (`main.tf`)

Para suportar a execução orquestrada e automatizada, é gerado dinamicamente um inventário das máquinas virtuais via scripts Python, acionados durante o processo de deploy, que também gere a execução dos playbooks Ansible para configuração dos laboratórios.

```
1  resource "null_resource" "update_inventory_debian" {
2    triggers = {
3      always_run = timestamp()
4    }
5
6    provisioner "local-exec" {
7      command = <<EOT
```

```
8       export WORKSPACE=$(terraform workspace show)
9       INVENTORY_FILE="/home/servidor/desafios/ctf/desafio1/inventory
            -${var.deploy_name}.txt"
10      SCRIPT_PATH="/home/servidor/desafios/ctf/desafio1/scripts/
            dynamic_inventory.py"
11      if [ ! -f "$SCRIPT_PATH" ]; then echo "Erro: Script nao
            encontrado!"; exit 1; fi
12      python3 "$SCRIPT_PATH" "$INVENTORY_FILE"
13    EOT
14  }
15
16  provisioner "local-exec" { command = "bash /home/servidor/desafios
      /ctf/desafio1/scripts/add_network_vm.sh" }
17  provisioner "local-exec" { command = "bash /home/servidor/desafios
      /ctf/desafio1/scripts/reset_ssh_key.sh" }
18  provisioner "local-exec" { command = "bash /home/servidor/desafios
      /ctf/desafio1/scripts/setup_configure.sh" }
19  provisioner "local-exec" { command = "cat /home/servidor/desafios/
      ctf/desafio1/inventory.json" }
20  provisioner "local-exec" { command = "python3 /home/servidor/
      desafios/ctf/desafio1/scripts/scp_files.py" }
21
22  provisioner "local-exec" {
23    command = <<EOT
24      SSHPASS='tesecyber2024' /home/servidor/desafios/backend/
            fastapi/bin/ansible-playbook \
25        -i /home/servidor/desafios/ctf/desafio1/inventory.json \
26        -u usertese \
27        --ssh-common-args="-o ProxyCommand='sshpass -e ssh -l
              servidor 193.136.195.51 -W %h:%p' -o
              StrictHostKeyChecking=no" \
28        /home/servidor/desafios/ctf/desafio1/ansible/base_playbook.
              yml
29    EOT
```

```
30    }
31 }
```

Listing E.4: Configuração do módulo de inventário no Terraform, utilizando o recurso `null_resource` com múltiplos provisioners `local-exec` para geração dinâmica do inventário, execução de scripts de configuração e automação via Ansible.

E.6 Execução dos Deploys

A execução das máquinas é gerida pelo script `deploy_central.sh`, que automatiza o *provisioning* com Terraform, criação de workspaces dinâmicos e geração de ficheiros VPN. O script é executado com sete argumentos: `<challenge_type> <challenge_number> <vm_template> <vm_cores> <vm_memory> <vm_disk_size> <vm_storage>`. Exemplo:

```
./deploy_central.sh ctf 1 Maquinas 2 2048 32 local-lvm
```

Listing E.5: Execução do script `deploy_central.sh` para implantação da infraestrutura central, indicando o workspace, tipo de desafio, número de máquinas, memória, armazenamento e volume lógico.

O script (`deploy_central.sh`) contém:

```
#!/bin/bash

if [ "$#" -ne 7 ]; then
    echo "Usage: $0 <challenge_type> <challenge_number> <vm_template
        > <vm_cores> <vm_memory> <vm_disk_size> <vm_storage>"
    exit 1
fi

CHALLENGE_TYPE=$1
CHALLENGE_NUMBER=$2
VM_TEMPLATE=$3
VM_CORES=$4
VM_MEMORY=$5
VM_DISK_SIZE=$6
VM_STORAGE=$7
```

```
15
16 BASE_PATH="/home/servidor/desafios"
17 CHALLENGE_DIR="$BASE_PATH/$CHALLENGE_TYPE/desafio$CHALLENGE_NUMBER"
18
19 if [ ! -d "$CHALLENGE_DIR" ]; then
20     echo "Challenge directory '$CHALLENGE_DIR' does not exist."
21     exit 1
22 fi
23
24 cd "$CHALLENGE_DIR/terraform" || exit
25
26 CTF_PLAYBOOK_PATH="$CHALLENGE_DIR/ansible/ctf_playbook.yml"
27 LAB_PLAYBOOK_PATH="$CHALLENGE_DIR/ansible/lab_playbook.yml"
28 BASE_PLAYBOOK_PATH="$CHALLENGE_DIR/ansible/base_playbook.yml"
29 MAIN_TF_PATH="$CHALLENGE_DIR/terraform/main.tf"
30
31 if [ "$CHALLENGE_TYPE" == "labs" ]; then
32     REQUIRED_PLAYBOOK_PATH="$LAB_PLAYBOOK_PATH"
33 elif [ "$CHALLENGE_TYPE" == "ctf" ]; then
34     REQUIRED_PLAYBOOK_PATH="$CTF_PLAYBOOK_PATH"
35 else
36     echo "Invalid challenge type: $CHALLENGE_TYPE. Only 'labs' or '
        ctf' are supported."
37     exit 1
38 fi
39
40 if [ ! -f "$BASE_PLAYBOOK_PATH" ] || [ ! -f "$REQUIRED_PLAYBOOK_PATH
    " ] || [ ! -f "$MAIN_TF_PATH" ]; then
41     echo "Required files are missing in the challenge directory."
42     exit 1
43 fi
44
45 declare -A NUM_VMS
46 NUM_VMS[1]=1
```

```
47 NUM_VMS[2]=1
48 NUM_VMS[3]=1
49 NUM_VMS[4]=2
50
51 if [ -z "${NUM_VMS[$CHALLENGE_NUMBER]}" ]; then
52     echo "No predefined number of VMs for challenge
        $CHALLENGE_NUMBER."
53     exit 1
54 fi
55
56 BASE_WORKSPACE_NAME="${CHALLENGE_TYPE}${CHALLENGE_NUMBER}_exec"
57 COUNTER=1
58 while true; do
59     WORKSPACE_NAME="${BASE_WORKSPACE_NAME}${COUNTER}"
60     if terraform workspace list | grep -q "$WORKSPACE_NAME"; then
61         COUNTER=$((COUNTER + 1))
62     else
63         terraform workspace new "$WORKSPACE_NAME"
64         break
65     fi
66 done
67
68 echo "Using workspace: $WORKSPACE_NAME"
69 terraform workspace select "$WORKSPACE_NAME"
70
71 WORKSPACE_FILE="$CHALLENGE_DIR/workspace_name.txt"
72 EXEC_ID_FILE="$CHALLENGE_DIR/exec_id.txt"
73
74 echo "$WORKSPACE_NAME" > "$WORKSPACE_FILE"
75 EXEC_ID=$(echo "$WORKSPACE_NAME" | grep -o 'exec[0-9]\+')
76 if [ -n "$EXEC_ID" ]; then
77     echo "$EXEC_ID" > "$EXEC_ID_FILE"
78 else
79     echo "Error: Could not extract execXXX from workspace name."
```

```
80      exit 1
81  fi
82
83  terraform init && terraform apply -auto-approve || { echo "Terraform
        apply failed."; exit 1; }
84
85  CCD_DIR="/etc/openvpn/ccd"
86  LAST_CLIENT=$(ls -1 "$CCD_DIR"/client* 2>/dev/null | grep -o '
        [0-9]\+$' | sort -n | tail -1 || echo 0)
87  CLIENT_ID=$((LAST_CLIENT + 1))
88  VPN_FILE_NAME="client${CLIENT_ID}.ovpn"
89  VPN_DEST_PATH="$CHALLENGE_DIR/$VPN_FILE_NAME"
90  VPN_INFO_FILE="/home/servidor/desafios/vpn_info.json"
91  DEST_PATH="/home/servidor/desafios/cyberlearn/src/pages/vpn_info.
        json"
92
93  CREATE_VPN_SCRIPT="/home/servidor/desafios/create_vpn_client.sh"
94  if [ -x "$CREATE_VPN_SCRIPT" ]; then
95      "$CREATE_VPN_SCRIPT" "$CHALLENGE_TYPE" "$CHALLENGE_NUMBER" || {
            echo "Erro: Falha ao executar create_vpn_client.sh..."; exit
            1; }
96  else
97      echo "Erro: Script $CREATE_VPN_SCRIPT n o encontrado ou n o
            executavel."
98      exit 1
99  fi
100
101 if [ ! -f "$VPN_DEST_PATH" ]; then
102     echo "Erro: Ficheiro $VPN_DEST_PATH nao foi gerado por
            create_vpn_client.sh."
103     exit 1
104 fi
105
```

```
106 NEW_ENTRY=$(jq -n --arg vpn_file "$VPN_DEST_PATH" --arg c_type "
    $CHALLENGE_TYPE" --arg c_number "$CHALLENGE_NUMBER" '{
    vpn_file_generated: $vpn_file, challenge_type: $c_type,
    challenge_number: $c_number}')
107
108 if [ -f "$VPN_INFO_FILE" ]; then
109     jq ". += [$NEW_ENTRY]" "$VPN_INFO_FILE" > "${VPN_INFO_FILE}.tmp"
          && mv "${VPN_INFO_FILE}.tmp" "$VPN_INFO_FILE"
110 else
111     echo "[$NEW_ENTRY]" > "$VPN_INFO_FILE"
112 fi
113
114 cp "$VPN_INFO_FILE" "$DEST_PATH" || { echo "Erro: Falha ao copiar
    $VPN_INFO_FILE para $DEST_PATH."; exit 1; }
115
116 echo "Ficheiro vpn_info.json atualizado em $VPN_INFO_FILE e copiado
    para $DEST_PATH"
117 echo "Conteudo do diretorio $CHALLENGE_DIR:"
118 ls -l "$CHALLENGE_DIR"
119
120 echo "Configuration completed successfully."
```

Listing E.6: Conteúdo do script `deploy_central.sh`, responsável por automatizar o processo de deploy das máquinas virtuais, criar workspaces Terraform, executar playbooks Ansible e gerar configurações de VPN.

O script gere o ciclo completo do deploy, incluindo a criação de workspaces e ficheiros VPN.