UNIVERSIDADE
LUSÓFONA

Venom Autonomous Pentest Tool

Trabalho Final de curso

Relatório Intercalar 1º Semestre

João Pedro Santos Ramos: 22309710

Orientador: Daniel Silveira

Coorientador: Luís Miguel Campos

Trabalho Final de Curso | LEI | 29/06/2026

www.ulusofona.pt

Direitos de cópia

Resumo

A prática de Pentesting é uma componente crítica na validação da postura de segurança das organizações. No entanto, a execução destes testes em ambientes laboratoriais enfrenta, atualmente, um obstáculo significativo: a dependência excessiva da intervenção humana para a tomada de decisão a cada etapa do ataque. Esta abordagem manual resulta em processos ineficientes, difíceis de escalar e carentes de reprodutibilidade científica.

Este projeto propõe o desenvolvimento do Venom APT **(Autonomous Pentesting Tool)**, uma ferramenta de orquestração ofensiva desenhada exclusivamente para ambientes laboratoriais autorizados. O sistema visa automatizar o ciclo de vida completo de um teste de intrusão, desde o reconhecimento inicial até à geração de relatórios, eliminando a microgestão humana. A arquitetura do Venom APT baseia-se num motor de decisão autónomo que utiliza planeamento condicional e árvores de ataque para determinar logicamente a melhor ação seguinte, emulando o comportamento de um adversário real de forma consistente.

A solução integra mecanismos de controlo operacional (guardrails), responsáveis por limitar e validar as ações do sistema, bem como rastreabilidade total, assegurando que a automação opera estritamente dentro do perímetro autorizado e em conformidade com princípios éticos. Os objetivos centrais do projeto incluem a redução do tempo de execução do ciclo de ataque entre **50%- 60%** e o aumento da densidade de cobertura de táticas e técnicas (TTPs) em **2.5x** face aos testes manuais. Estes valores são definidos como **parâmetros-alvo realistas** para um protótipo que automatiza sobretudo as fases repetitivas do ciclo de ataque, mantendo a análise crítica e a validação final no operador humano; a respetiva justificação detalhada é apresentada no capítulo dedicado aos **KPIs de avaliação e à metodologia experimental**. O resultado final é um protótipo funcional que demonstra como a automação estruturada pode aumentar a eficiência, a consistência e a auditabilidade dos exercícios de cibersegurança, sem comprometer a segurança do ambiente de teste.

Palavras-chave:

Automação de Pentesting, Cibersegurança Ofensiva, Motor de Decisão, Reprodutibilidade, Ambientes Laboratoriais.

Abstract

Pentesting is a critical component in validating the security posture of organizations. However, performing these tests in laboratory environments currently faces a significant obstacle: excessive reliance on human intervention for decision-making at each stage of the attack. This manual approach results in inefficient processes that are difficult to scale and lack scientific reproducibility.

This project proposes the development of Venom APT **(Autonomous Pentesting Tool)**, an offensive orchestration tool designed exclusively for authorized laboratory environments. The system aims to automate the entire intrusion test lifecycle, from initial reconnaissance to report generation, eliminating human micromanagement. The architecture of Venom APT is based on an autonomous decision engine that uses conditional planning and attack trees to logically determine the best next action, consistently emulating the behavior of a real adversary.

The solution integrates strict security mechanisms (guardrails) and full traceability, ensuring that automation operates strictly within the authorized perimeter and in accordance with ethical principles. The core objectives of the project include reducing the attack cycle execution time by **50%-60%** and increasing the coverage density of tactics and techniques (TTPs) by **2.5x** compared to manual testing. These values are defined as **realistic target parameters** for a prototype that primarily automates the repetitive phases of the attack cycle, while maintaining critical analysis and final validation with the human operator; the detailed justification for this is presented in the chapter dedicated to evaluation **KPIs and experimental methodology**. The end result is a working prototype that demonstrates how structured automation can increase the efficiency, consistency, and auditability of cybersecurity exercises without compromising the security of the test environment.

Keywords:

Pentesting Automation, Offensive Cybersecurity, Decision Engine, Reproducibility, Laboratory Environments.

Índice

Lista de Figuras

Lista de Tabelas

1 Identificação do Problema

1.1 Enquadramento Prático e Contexto Real do Problema

No panorama atual da segurança ofensiva, a metodologia predominante para a realização de testes de intrusão (pentesting), mesmo em ambientes laboratoriais rigorosamente controlados, permanece intrinsecamente manual e dependente de decisão humana. Os processos de ataque simulado exigem intervenção crítica de operadores humanos especializados em todas as etapas do ciclo de vida — desde o reconhecimento inicial até à exploração de vulnerabilidades, pós-exploração e elaboração de relatórios técnicos finais.

Esta dependência excessiva da operação manual cria um ponto de estrangulamento operacional significativo: o caráter manual da tomada de decisão, que condiciona diretamente o tempo de execução, a cobertura de TTPs e a reprodutibilidade dos testes (ver Secções 3.1–3.3 e 4.2.6).

O tempo necessário para o operador analisar resultados intermédios e decidir o comando seguinte prolonga o ciclo de teste e dificulta a sua escalabilidade, uma vez que cada decisão exige atenção individualizada e sequencial (ver definição do KPI de Eficiência Temporal na Secção 4.2.6 e o modelo de recolha de dados descrito nas Secções 4.4–4.7).

Esta latência decisional exige um esforço humano intensivo por ciclo de teste e introduz uma variabilidade subjetiva significativa, dependente da perícia, experiência e até do estado de fadiga momentâneo do operador. Este aspeto está diretamente relacionado com o KPI de Reprodutibilidade definido na Secção 4.2.6 e com o enquadramento de escassez de talento em cibersegurança analisado na Secção 3.1.2.

O problema manifesta-se de forma particularmente crítica num contexto de crescimento da complexidade dos ambientes de TI e da sofisticação das ameaças cibernéticas, amplamente documentado em relatórios de threat intelligence e análises de tendências (por exemplo, ENISA Threat Landscape e Verizon Data Breach Investigations Report) ([ENISA, 2023]; [Verizon, 2024]). As organizações enfrentam uma lacuna crescente entre a necessidade de testes frequentes, escaláveis e auditáveis e a realidade dos métodos atuais que, pela sua natureza manual, comprometem a eficiência operacional, a profundidade da cobertura de cenários de ataque e a fiabilidade científica dos resultados (ver também Secções 3.1.1 e 3.3).

1.2 Evidências de Circunstâncias Reais: Dimensão Quantitativa do Problema

Crescimento do Mercado e Investimento Significativo em Soluções Manuais

O mercado global de penetration testing demonstra crescimento robusto, com projeções a indicar que excederá 5 biliões USD anualmente até 2031[1], com uma taxa de crescimento anual

composta superior a 24% até 2026. O mercado mais abrangente de security testing atingiu 13 biliões USD em 2024 e projeta-se alcançar 58.3 biliões USD até 2033, representando um CAGR de 18.1%[2].

Este investimento massivo reflete a criticidade crescente dos testes de segurança. Os dados do State of Pentesting 2024 revelam que as empresas investem, em média, 164,400 USD anualmente em assessments manuais de pentesting — correspondendo a 12.9% do orçamento total de IT Security[3]. Contudo, apesar deste investimento substancial, 60% das empresas conduzem pentesting apenas duas vezes por ano ou menos, resultando numa avaliação meramente snapshot da postura de segurança, com longos períodos de exposição a risco não validado entre testes.

Adicionalmente, o mercado global de pentesting apresentou um crescimento expressivo em 2023, tendo alcançado um valor estimado de 2.8 mil milhões USD, com projeções a indicar um crescimento anual superior a 17% entre 2024 e 2032 [4]. Este dinamismo é impulsionado por exigências regulatórias crescentes e pela expansão das superfícies de ataque digitais decorrentes da adoção massiva de *cloud computing* e da integração contínua de novas tecnologias. Em paralelo, vários relatórios de mercado salientam que as organizações enfrentam desafios na priorização e otimização de investimentos em IT Security, reforçando a pressão para maximizar a eficiência dos recursos disponíveis.

1.3 O Skills Gap Crítico em Pentesting

A indústria de cibersegurança enfrenta uma escassez crítica e crescente de profissionais qualificados. O estudo *ISC2 Cybersecurity Workforce Study 2024*, estima um déficit global de 4.8 milhões de profissionais de cibersegurança, o que representa um aumento de 19% face ao ano anterior [5]. Estes skills gap tem repercussões diretas e mensuráveis: organizações com lacunas de competências consideradas críticas ou significativas apresentam quase o dobro da probabilidade de sofrer breaches materiais quando comparadas com organizações sem *skills gaps*[6].

Em termos financeiros, o *IBM Cost of a Data Breach Report 2024* demonstra que equipas de segurança com carência severa de recursos especializados enfrentam, em média, um acréscimo de 1.76 milhões USD no custo de cada breach relativamente a organizações com equipas adequadamente dimensionadas[7].

No domínio específico de *penetration testing*, estudos nacionais indicam que a escassez de competências é particularmente acentuada. O relatório governamental *Cyber Security Skills in the UK Labour Market 2023* estima que cerca de 23% das empresas privadas no Reino Unido apresentem *skills gaps* em *penetration testing*, evidenciando a dificuldade em recrutar e reter especialistas nesta área[8].

Em paralelo, relatórios recentes sobre o estado do pentesting, como o *State of Pentesting 2024*, apontam a disponibilidade de pentesters qualificados como uma das principais barreiras à realização de testes com a frequência e abrangência desejadas, a par de preocupações com o risco para a continuidade do negócio e da falta de recursos internos para remediar as vulnerabilidades identificadas[9].

Esta escassez é particularmente problemática porque o pentesting é uma disciplina avançada que exige competências técnicas especializadas desenvolvidas ao longo de meses ou anos de prática intensiva. A rápida evolução tecnológica em áreas como *cloud security*, *AI security* e *containerization* agrava o problema, uma vez que os programas de formação tradicionais não conseguem produzir especialistas ao ritmo necessário para satisfazer a procura do mercado.

1.4 Desfasamento entre Mudança e Validação

Uma das evidências mais claras da inadequação dos métodos manuais é o desfasamento entre a frequência de mudança nos ambientes de TI e a frequência de validação de segurança (doravante designado por Frequency Gap). O *State of Pentesting Survey 2024* revela que 73% das empresas reportam mudanças nas suas infraestruturas de TI pelo menos trimestralmente, enquanto apenas 40% conduzem pentesting com a mesma frequência[10].

Este desfasamento de 33 pontos percentuais significa que a maioria das organizações opera com janelas prolongadas de risco não validado. Cada mudança relevante na infraestrutura — *deployments* de novos serviços, atualizações de configuração, adição de novos endpoints — pode introduzir vulnerabilidades que permanecem por testar durante semanas ou meses. Relatórios anteriores sobre o estado do pentesting indicam ainda que cerca de 85% das organizações previam aumentar os seus orçamentos de pentesting, na sequência do crescimento de ameaças como *ransomware* e do aumento da complexidade dos ambientes de TI [11]. Contudo, este reforço orçamental não se traduziu, na maioria dos casos, num aumento proporcional da frequência dos testes, devido às limitações estruturais do modelo manual.

1.5 Volume Crescente de Vulnerabilidades e Sobrecarga Operacional

Os dados demonstram um aumento sustentado no volume de vulnerabilidades identificadas. O Cobalt State of Pentesting Report 2024 documenta um crescimento de 21% no número médio de findings por pentest engagement em 2023, em comparação com 2022. Este aumento acompanha também a subida do número anual de CVEs publicados, refletindo um cenário em que novas vulnerabilidades são introduzidas e descobertas a um ritmo cada vez mais elevado[12].

Em paralelo, o State of Pentesting 2024 da Pentera indica que mais de 60% das organizações reportam um mínimo semanal de 500 eventos de segurança que requerem remediação, levando os autores a concluir que atingir um estado de correção total de vulnerabilidades se tornou, na prática, inviável num contexo operacional real[13]. Na prática, as equipas de segurança são forçadas a priorizar continuamente, tratando apenas uma fração das vulnerabilidades e

incidentes identificados dentro das janelas de tempo e recursos disponíveis, o que contribui para a manutenção de níveis persistentes de risco.

Esta sobrecarga é agravada por cortes orçamentais e reduções de pessoal. O relatório da Cobalt evidencia uma redução significativa na taxa global de correção (apenas 29.31% das vulnerabilidades identificadas se encontram no estado “corrigido”)e um aumento do tempo médio de reparação (Mean Time To Repair - MTTR) face a anos anteriores.

Esta degradação é atribuinda a um contexto generalizado de contenção de custos, caracterizado pela redução de equipas de segurança e limitação de recursos disponíveis[14]. Com menos especialistas para tratar vulnerabilidades e menor capacidade técnica para lidar com casos mais complexos, o tempo necessário para implementar correções aumenta significativamente, criando um ciclo vicioso de exposição prolongada a risco.

Este fenómeno reforça a necessidade de soluções automatizadas que permitam reduzir a dependência de intervenção manual e melhorar a capacidade de resposta das equipas de segurança.

1.6 Cinco Dificuldades Críticas Identificadas

A persistência de processos manuais resulta em cinco dificuldades críticas que limitam estruturalmente a eficácia dos testes de segurança ofensiva:

1.6.1 Ineficiência Temporal (Latência de Execução)

O caráter manual da tomada de decisão o “tempo de pensar” entre a obtenção do *output* de uma ferramenta e a escolha do próximo passo torna os testes morosos e ineficientes. Em ciclos de teste que envolvem múltiplos alvos ou vetores de ataque complexos, esta latência decisional impede a obtenção de resultados em tempo útil.
Relatórios recentes sobre o estado do pentesting indicam que as organizações investem, em média, cerca de 164,400 USD por ano em exercícios de pentesting manual, correspondendo a aproximadamente 12.9% do orçamento total de segurança, embora 60% das organizações realizem pentests apenas duas vezes por ano ou menos[15]. Isto significa que um investimento significativo é canalizado para avaliações pontuais (*snapshot*) da postura de segurança. Em paralelo, a literatura recente sobre *automated security validation* e *automated red teaming* aponta para reduções relevantes no esforço manual de investigação e na carga operacional, ao automatizar tarefas repetitivas de descoberta, correlação e priorização de eventos [16].

1.6.2 Cobertura de Cenários Limitada

Devido ao elevado esforço humano exigido por cada teste, os operadores tendem a privilegiar caminhos de ataque mais diretos ou previamente conhecidos. Esta tendência reduz o número de variantes e TTPs (Táticas, Técnicas e Procedimentos) explorados numa mesma janela temporal, deixando por identificar vulnerabilidades que permanecem ocultas por não serem abrangidas pelos testes mais comuns ou por exigirem exploração mais aprofundada.

Estudos comparativos entre testes manuais e automatizados destacam que as abordagens automatizadas oferecem maior velocidade e cobertura superficial, mas têm dificuldade em

identificar vulnerabilidades lógicas, falhas de desenho e problemas altamente contextuais. Por isso, diversas fontes industriais sublinham que o manual penetration testing permanece crítico para identificar vulnerabilidades complexas e de alto impacto, funcionando como complemento indispensável à automação [17]. Assim, o desafio não é eliminar o elemento humano, mas ampliar a cobertura sem aumentar linearmente o esforço manual.

1.6.3 Inconsistência e Subjetividade

A dependência de operadores humanos conduz inevitavelmente a variações nos resultados. Diferentes profissionais de pentesting, ou o mesmo profissional em momentos distintos, tomam decisões ad hoc diferentes, resultando numa falta de padronização que dificulta a análise comparativa e a validação longitudinal.

Esta inconsistência é inerente a processos manuais, onde fatores como perícia individual, fadiga, conhecimento tácito não documentado e preferências pessoais de ferramentas influenciam as escolhas táticas. Sem um modelo formalizado de decisão ou uma camada de orquestração que imponha uma lógica comum, torna-se difícil comparar resultados entre ciclos de teste, equipas ou fornecedores distintos.

1.6.4 Défice de Reprodutibilidade Científica

A reprodutibilidade constitui um dos principais desafios na área da cibersegurança ofensiva, em particular no contexto de testes de intrusão (pentesting). Embora muitos trabalhos académicos apresentem novas técnicas, ferramentas ou abordagens, a capacidade de reproduzir os resultados obtidos é frequentemente limitada ou inexistente.

Estudos empíricos sobre reprodutibilidade em sistemas computacionais demonstram que este problema é estrutural. Uma análise a 402 artigos na área de sistemas computacionais revelou que, apesar de 56,2% dos trabalhos disponibilizarem código, apenas 32,3% conseguiam ser executados com sucesso[18] num curto período, sendo que muitos exigiam esforço adicional ou intervenção direta dos autores. Este resultado evidencia que a simples disponibilização de código não garante a reprodutibilidade dos resultados.

Adicionalmente, estudos prévios na área indicam níveis ainda mais baixos de partilha de código. Por exemplo, apenas cerca de 9% dos artigos disponibilizam código publicamente[19], o que limita significativamente a validação independente.

Outro fator crítico identificado é a dependência do conhecimento implícito dos autores. Em muitos casos, a reprodução de resultados requer contacto direto com os autores dos artigos analisados, sendo que uma parte significativa do código analisado apenas foi obtida através de pedidos por email[20]. Esta dependência compromete a transparência e a escalabilidade da validação científica.

No contexto específico da segurança ofensiva, estes desafios são ainda mais acentuados. Ferramentas de pentesting operam frequentemente em ambientes altamente dinâmicos, dependentes de configurações específicas, versões de software, estado da rede e comportamento dos alvos. Pequenas variações no ambiente podem conduzir a resultados completamente diferentes, tornando difícil replicar ataques de forma consistente.

Face a estes desafios, torna-se evidente a necessidade de abordagens que garantam:

- Registo completo das execuções (logs, comandos, resultados)
- Persistência estruturada de artefactos
- Normalização dos resultados das ferramentas
- Definição clara do contexto e ambiente de execução

Neste contexto, o sistema proposto neste trabalho — Venom APT — é concebido para mitigar diretamente estes problemas. Através da automatização do processo de execução de ferramentas, da recolha sistemática de evidência e da normalização dos resultados, o sistema permite não só executar testes de intrusão de forma estruturada, como também garantir a sua rastreabilidade e reprodutibilidade.

Desta forma, o Venom APT não se limita a automatizar tarefas de pentesting, mas introduz uma abordagem orientada à engenharia de sistemas, onde cada ação é registada, cada resultado é estruturado e cada execução pode ser analisada e reproduzida posteriormente. Esta abordagem alinha-se com as necessidades identificadas na literatura e contribui para a evolução da prática de segurança ofensiva para um modelo mais rigoroso, transparente e científico.

1.6.5 Escalabilidade Reduzida

O modelo manual não é escalável. A necessidade de intervenção humana constante limita estruturalmente a frequência e a abrangência dos testes, um requisito crítico num contexto em que as superfícies de ataque e as exigências regulatórias se encontram em expansão contínua.

O desfasamento entre mudança e validação descrito anteriormente ilustra esta limitação: o *State of Pentesting* 2024 indica que 73% das organizações reportam alterações significativas na sua infraestrutura de TI pelo menos trimestralmente, enquanto apenas 40% realizam testes de intrusão com a mesma cadência[21] .

Em simultâneo, mais de 60% das organizações reportam um mínimo semanal de 500 eventos de segurança que requerem correção, levando os autores a concluir que atingir um estado de “orreção total de vulnerabilidades é, na prática, inviável[22].Nestes termos, aumentar a frequência e a cobertura dos testes exclusivamente através de mais trabalho manual torna-se operacionalmente inviável, criando janelas persistentes de risco não validado que podem ser exploradas por atacantes.

1.7 Problema Fundamental: Ausência de Automatização do Fluxo de Decisão

O problema central que este projeto procura resolver é a ausência de automatização estruturada do fluxo de decisão em testes de intrusão em ambiente laboratorial. Atualmente, existem ferramentas capazes de executar comandos de ataque de forma automatizada, mas

falta uma camada de orquestração que consiga estruturar o fluxo de ataque de forma lógica e autónoma, desde o reconhecimento inicial até à pós-exploração e geração de relatórios.

Esta lacuna distingue-se claramente da mera automatização da execução. Ferramentas convencionais de testes de intrusão automatizam ações individuais — como varrimento de portas, exploração de vulnerabilidades conhecidas ou quebra de palavras-passe — mas continuam a exigir intervenção humana para decidir qual o próximo passo com base nos resultados obtidos. O operador tem de interpretar os resultados, escolher a próxima técnica, ajustar parâmetros e decidir quando terminar o cenário. O que falta no ecossistema atual são sistemas com capacidade de planeamento condicional e decisão autónoma; isto é, sistemas que analisam os resultados de uma ação e, através de lógica formal ou modelos de planeamento, selecionam autonomamente a próxima técnica a aplicar, emulando o raciocínio de um atacante humano experiente.

A investigação académica e industrial recente demonstra que esta abordagem é tecnicamente viável. Trabalhos conduzidos por MITRE, nomeadamente por Applebaum, Miller, Strom e coautores, demonstram que a emulação automatizada de adversários deve ser tratada como um problema conjunto de planeamento e atuação em ambientes com incerteza (planning and acting with unknowns), e não apenas como um problema de execução de comandos isolados [23].

Estes autores descrevem sistemas em que um planeador automático, baseado numa representação lógica do ambiente e dos perfis de adversário, decide que ações executar em cenários de elevada incerteza, navegando de forma autónoma por redes alvo após o compromisso inicial.

O sistema CALDERA, também desenvolvido pela MITRE, concretiza estes princípios na prática, implementando agentes capazes de emular adversários com base no conhecimento estruturado do MITRE ATT&CK. Neste contexto, um planeador automático seleciona e executa táticas, técnicas e procedimentos (TTPs) com o objetivo de atingir metas previamente definidas.

Estudos recentes indicam que o CALDERA utiliza uma representação lógica do ambiente e das técnicas ATT&CK para escolher, de forma autónoma, as ações mais adequadas à progressão do ataque[24]. Em conjunto, estes trabalhos demonstram que é possível automatizar o raciocínio tático em cenários de emulação de adversários, indo além da execução mecânica de comandos.

Em paralelo, têm surgido abordagens baseadas em aprendizagem automática (machine learning) focadas na modelação e previsão da cadeia de ataque (cyber kill chain). Sistemas como o KillChainGraph recorrem a modelos avançados — incluindo algoritmos de gradiente reforçado, modelos de linguagem e redes neuronais de grafos — para prever a evolução de campanhas

maliciosas ao longo das diferentes fases do ataque, atingindo níveis elevados de precisão na previsão de técnicas e transições [25].

Adicionalmente, estudos recentes indicam que a aplicação de redes neuronais de grafos permite prever a progressão de ataques com níveis elevados de precisão quando integradas em estratégias de defesa baseadas em inteligência artificial[26]. Estas abordagens contribuem para melhorar a priorização de eventos e reduzir o esforço de análise em centros de operações de segurança (SOC).

No entanto, o foco dominante destes trabalhos é a perspetiva defensiva — deteção, correlação de eventos e priorização de alertas — e não a execução autónoma de cadeias ofensivas completas em ambiente de testes de intrusão.

Assim, persiste uma lacuna específica: a ausência de ferramentas que integrem planeamento automático e análise de resultados no fluxo completo de decisão ofensiva em contexto controlado de laboratório, alinhado com frameworks como o MITRE ATT&CK, mas orientado para testes de intrusão autónomos, éticos e reprodutíveis. É precisamente esta lacuna que o projeto Venom APT se propõe colmatar.

1.8 Validação por Terceiros e Fundamentação do Problema

A formulação deste problema não é meramente teórica; resulta de circunstâncias reais e é validada por múltiplas fontes independentes, que se complementam.

Validação de Mercado:

Relatórios industriais recentes, como o State of Pentesting 2024 (Pentera)[27] e o State of Pentesting Report 2024 (Cobalt) [28], documentam práticas, constrangimentos e lacunas estruturais no ecossistema de testes de intrusão. Estes estudos evidenciam, entre outros aspetos, que:

• As organizações investem montantes significativos em testes de intrusão manuais, mas continuam a operar com avaliações pontuais (snapshot), com frequência de testes insuficiente face ao ritmo de mudança das infraestruturas.

• A disponibilidade de profissionais qualificados em testes de intrusão é apontada como uma das principais barreiras à realização de testes com a frequência desejada.

• As equipas enfrentam pressão crescente para aumentar a eficiência, num contexto de orçamentos de segurança estagnados ou em redução e volumes cada vez maiores de vulnerabilidades a corrigir.

Em paralelo, relatórios de mercado sobre avaliação de segurança indicam que este segmento apresenta taxas de crescimento anual elevadas, refletindo tanto a criticidade do problema como a necessidade de soluções mais eficientes e automatizadas.

Validação Académica:

Frameworks de referência como o MITRE ATT&CK consolidam um corpo de conhecimento estruturado sobre táticas e técnicas de adversários, baseado em observações reais de campanhas de ataque. Plataformas como o CALDERA demonstram que é tecnicamente viável codificar táticas, técnicas e procedimentos (TTPs) em modelos de planeamento automático e utilizá-los para emular adversários em exercícios automatizados[29].

A literatura científica sobre emulação de adversários e automatização de equipas de teste reforça esta conclusão, demonstrando que a utilização de planeadores automáticos, modelos de decisão e técnicas de inteligência artificial permite a navegação autónoma em redes complexas após o compromisso inicial, com capacidade de adaptação a incerteza e a mudanças no ambiente[30].

Validação Empresarial (Parceiro PDMFC):

A empresa PDMFC, enquanto parceiro empresarial do projeto, atua como consultor de requisitos de mercado e validador externo. Este parceiro fornece:

• Feedback sobre a relevância prática das funcionalidades propostas;

• Informação sobre requisitos reais de clientes e constrangimentos operacionais;

• Validação do potencial utilitário da ferramenta em cenários concretos de cibersegurança.

Desta forma, garante-se que o Venom APT é desenvolvido não em isolamento académico, mas em resposta a necessidades efetivas identificadas no terreno.

Validação Científica:

A problemática da reprodutibilidade está amplamente documentada no domínio da engenharia de software e dos sistemas computacionais[31] . Estudos demonstram que processos complexos, sem automatização estruturada, documentação rigorosa e registo detalhado das execuções, são difíceis de reproduzir de forma consistente.

Neste contexto, torna-se relevante introduzir o conceito de artefactos completos. Em engenharia de software e experimentação computacional, artefactos completos correspondem ao conjunto de elementos necessários para reproduzir integralmente um processo ou experimento, incluindo código, configurações, dados de entrada, comandos executados, resultados gerados e metadados associados.

No contexto dos testes de intrusão manuais, estes artefactos são frequentemente incompletos ou inexistentes. A ausência de registo sistemático das ações executadas, das condições do ambiente e dos resultados obtidos impede a reprodução exata dos cenários de teste.

Como consequência, verificam-se:

• Resultados dificilmente replicáveis entre ciclos de teste;

• Dificuldade em comparar objetivamente o impacto de medidas de reforço de segurança;

• Limitações na auditabilidade e validação dos resultados obtidos.

Este enquadramento reforça a necessidade de sistemas como o Venom APT que, para além de automatizarem o fluxo de decisão ofensiva, geram registos estruturados e artefactos completos, permitindo reexecuções, auditoria e análise comparativa ao longo do tempo.

1.9 Conclusão: Passo no Sentido da Solução

O desenvolvimento do **Venom APT (Autonomous Pentesting Tool)** representa um **passo concreto e mensurável** no sentido de mitigar este problema multifacetado. Ao focar-se especificamente na **automatização do fluxo de decisão**, através de um motor de orquestração baseado em lógica condicional e planeamento de árvores de ataque, o projeto procura endereçar simultaneamente as cinco dificuldades críticas previamente identificadas:

- **Ineficiência temporal**, através da eliminação de parte substancial da latência decisional humana;
- **Cobertura limitada**, através da exploração sistemática de caminhos alternativos definidos em árvores de decisão e regras de orquestração;
- **Inconsistência**, através da padronização da lógica decisional e do comportamento do sistema perante *outputs* equivalentes;
- **Défice de reprodutibilidade**, através de *logging* estruturado automático de todos os passos, parâmetros e resultados;
- **Escalabilidade reduzida**, através da automação do ciclo completo de execução em ambientes laboratoriais controlados.

A solução proposta está alinhada com tendências emergentes na indústria, onde **automated red teaming** e *security validation* contínua demonstram reduções significativas em tempo de investigação e falsos positivos, bem como ganhos de produtividade das equipas de segurança. Contudo, o Venom APT distingue-se por **priorizar, desde a conceção, o rigor ético, a auditabilidade e a conformidade científica**:

- O âmbito é **deliberadamente limitado a ambientes laboratoriais controlados**, sem atuação em infraestruturas de produção.
- São definidos **guardrails rigorosos** quanto ao tipo de ações permitidas, alvos, tipos de payload e condições de execução.
- É privilegiada a **documentação exaustiva**, a transparência do fluxo de decisão e a capacidade de replicar, rever e auditar cada sessão de teste.

Deste modo, o projeto procura demonstrar que é possível **aumentar significativamente a eficiência operacional dos testes de segurança**, mantendo simultaneamente um rigor ético inabalável e uma base metodológica compatível com exigências de validação académica e industrial.

2 Benchmarking

2.1 Análise Comparativa e State of Art

Este capítulo apresenta uma análise comparativa sistemática do **Venom APT** face a soluções existentes no mercado e à investigação académica, enquadrando o projeto no **State Of Art em automação de segurança ofensiva e emulação de adversários**.

A análise estrutura-se em três dimensões principais:

1. **Identificação e caracterização de soluções existentes** (open-source, frameworks de red team e plataformas BAS);
2. **Posicionamento diferenciado do Venom APT** face a essas soluções;
3. **Enquadramento teórico-científico** do problema de automação da decisão em pentesting.

2.2 Categorização do Ecossistema de Soluções

O panorama atual de ferramentas de segurança ofensiva pode ser agrupado, de forma simplificada, em três grandes segmentos:

1. **Frameworks Open-Source Académicos e Comunitários**
   Incluem projetos como **MITRE CALDERA**[32], **Atomic Red Team** e o **Metasploit Framework**. Estas ferramentas priorizam transparência, extensibilidade e investigação, sendo amplamente utilizadas em contexto académico e de laboratório.
2. **Plataformas Comerciais de Red Team Operations / C2 Frameworks**
   Ferramentas como **Cobalt Strike**[33] constituem o padrão de facto para operações avançadas de red teaming, fornecendo capacidades sofisticadas de *command-and-control* (C2) e *post-exploitation*, mas operando primariamente num paradigma **human-in-the-loop**.
3. **Plataformas Enterprise de Breach and Attack Simulation (BAS)**
   Soluções como **Cymulate**, **Picus Security**, **AttackIQ** e **SafeBreach** representam a evolução comercial para **continuous security validation**, automatizando a execução de múltiplos cenários de ataque para validar controlos defensivos em ambientes de produção.

O **Venom APT** posiciona-se de forma complementar a estes segmentos: como um **protótipo académico de pentesting autónomo**, orientado para **laboratórios controlados**, com foco no **motor de decisão**, **guardrails de segurança** e **reprodutibilidade científica**.

2.3 Frameworks Open-Source: Fundação Tecnológica

2.3.1 MITRE CALDERA – Adversary Emulation Platform

O **MITRE CALDERA** é um *framework* open-source de **adversary emulation** desenvolvido pela MITRE e construído nativamente sobre o **MITRE ATT&CK**. A documentação oficial descreve-o

como uma plataforma para **automatizar exercícios de breach-and-attack simulation**, assistir *red teams* manuais e automatizar *incident response*.

Arquiteturalmente, o CALDERA assenta num servidor central que coordena **agentes** responsáveis por executar técnicas ATT&CK com base em perfis e *plugins* como:

- **Sandcat** – agente padrão;
- **Stockpile** – repositório de técnicas e TTPs;
- outros módulos para visualização, *reporting* e integrações.

A componente de planeamento automático está alinhada com o trabalho de Applebaum e Miller, que tratam a emulação de adversários como um problema conjunto de **planeamento e atuação em ambientes com incerteza ("planning and acting with unknowns")**, e não apenas como execução sequencial de scripts.

Limitações face ao Venom APT

Apesar das capacidades avançadas, o CALDERA apresenta limitações relativamente aos objetivos do Venom APT:

- **Foco inicial pós-compromisso** em redes enterprise (principalmente Windows), com cobertura menos integrada das fases iniciais de reconhecimento e descoberta.
- **Complexidade operacional** (instalação, configuração de agentes, definição de perfis), menos adequada a contextos académicos que valorizam simplicidade de *setup* e reprodutibilidade.
- **Guardrails genéricos**: a responsabilidade pela contenção recai sobretudo sobre o operador; o *framework* não foi concebido especificamente como ferramenta "*safety-by-design*" para laboratórios académicos.
- Geração de *logs* e relatórios existe, mas não está orientada, de raiz, para **KPIs formais de eficiência temporal, cobertura e consistência**, como os definidos para o Venom APT.

O Venom APT inspira-se em CALDERA e no ATT&CK, mas foca-se em:

- **full kill chain em laboratório** (recon → exploração → pós-exploração → relatório);
- **modelo de dados relacional completo** para análise científica;
- **guardrails de segurança explícitos** e mensuráveis.

2.3.2 Atomic Red Team – Biblioteca de Testes Atómicos

O **Atomic Red Team**, mantido pela Red Canary, é uma biblioteca open-source de **testes pequenos e focados**, mapeados à matriz **MITRE ATT&CK**, que permite simular atividades adversariais específicas de forma simples e reprodutível.

Cada *atomic test*:

- emula **uma única técnica ATT&CK** (por exemplo, T1059.001 – *PowerShell*; T1003 – *Credential Dumping*);

- é descrito em ficheiros **Markdown/YAML**, com pré-condições, comandos para vários sistemas operativos e instruções de *cleanup*;
- é executável via *frameworks* como Invoke-AtomicTest (PowerShell) ou atomic-operator (Python).

Vantagens:

- transparência total (código aberto);
- simplicidade de execução;
- reprodutibilidade inerente (testes atómicos idempotentes);
- portabilidade entre plataformas.

Limitações face ao Venom APT:

- Não existe **motor de decisão autónomo**; a escolha da técnica seguinte é sempre responsabilidade do operador ou de scripts externos.
- Não há **encadeamento automático**: o *output* de um teste não é usado nativamente como *input* de outro para construir uma cadeia de ataque.
- Geração de relatórios end-to-end é limitada – o foco está em testar técnicas individuais, não em orquestrar **ciclos completos de pentesting**.
- Guardrails são assumidos como responsabilidade do utilizador (o projeto parte do pressuposto de execução em ambiente controlado).

O **Atomic Red Team** funciona, assim, como um conjunto de **blocos de construção**. O Venom APT, por sua vez, posiciona-se como o **motor de orquestração autónomo** que encadeia técnicas num fluxo lógico de ataque em laboratório.

2.3.3 Metasploit Framework / Metasploit Pro – Exploitation e Auto-Exploitation

O **Metasploit Framework** é uma das ferramentas de penetration testing mais estabelecidas, com milhares de *exploit modules*, *payloads* e módulos auxiliares para *scanning* e *post-exploitation*.

A versão comercial **Metasploit Pro** introduz capacidades de **auto-exploitation**: quando o operador ativa a funcionalidade de *automated exploits*, a ferramenta **constrói um plano de ataque** com base na informação conhecida sobre o alvo (sistema operativo, serviços, vulnerabilidades detetadas).

Limitações face ao Venom APT:

- A lógica de decisão é focada na **seleção de exploits** com base em dados de vulnerabilidade e *reliability*, não num planeamento condicional completo da *kill chain*.
- O âmbito centra-se na fase de **exploitation**; as fases de reconhecimento e pós-exploração não são orquestradas de forma autónoma end-to-end.
- A responsabilidade por **ambito, ética e guardrails** recai no operador.

- A versão Pro é uma solução comercial orientada a equipas profissionais, não a protótipos académicos com modelo de dados explícito e KPIs científicos.

Em contraste, o Venom APT utiliza módulos de exploração como **blocos reutilizáveis**, mas coloca o foco na **decisão sequencial** (quando explorar, o que explorar, quando parar, como documentar) e na **reprodutibilidade** do processo em contexto de laboratório.

2.4 Plataformas Comerciais de Red Team Operations

2.4.1 Cobalt Strike – Post-Exploitation e C2 Framework

O **Cobalt Strike** é um *framework* comercial de *adversary simulation* e C2, amplamente utilizado por red teams para emular **ameaças persistentes avançadas** em redes enterprise.

Principais componentes:

- **Team Server** – servidor C2 que coordena a operação;
- **Beacon** – *payload* in-memory usado para persistência e *command execution*;
- **Malleable C2 Profiles** – permitem alterar o comportamento e indicadores de rede do Beacon;
- **Aggressor Script** – linguagem de scripting para automatizar fluxos e personalizar o comportamento.

Embora suporte automação via *scripts* e perfis customizados, o Cobalt Strike continua a ser, essencialmente, uma ferramenta **human-in-the-loop**: o operador decide quais comandos executar, quais hosts pivotar, que técnicas aplicar e quando terminar o ataque.

Limitações face ao Venom APT:

- Foco em **post-exploitation** em ambientes de produção;
- Ausência de **motor de decisão autónomo full kill chain**;
- Enquadramento legal e ético sensível (devido ao abuso por grupos maliciosos), que torna a sua utilização académica mais complexa.

O Venom APT opta deliberadamente por:

- um **contexto exclusivamente laboratorial**,
- uma arquitetura centrada em **planeamento condicional determinístico**,
- e **guardrails rígidos**, evitando os riscos associados a ferramentas C2 em produção.

2.5 Plataformas Enterprise de Breach and Attack Simulation (BAS)

As plataformas de **Breach and Attack Simulation (BAS)** fornecem uma abordagem altamente automatizada para **validar continuamente controlos de segurança** em ambientes de produção.

2.5.1 Cymulate

A **Cymulate** é uma plataforma de BAS que automatiza cenários de ataque para avaliar a postura de segurança, identificar fraquezas em controlos e validar defesas contra ameaças ativas ao longo de toda a *kill chain*.

Características típicas:

- execução recorrente de cenários de ataque baseados em TTPs reais e *threat intelligence* atualizada;
- foco em **ambientes de produção** (on-prem, cloud, endpoints, e-mail, web, etc.);
- relatórios orientados a **exposição ao risco** e eficácia de controlos.

2.5.2 Picus Security

A **Picus Security** combina BAS com validação de controlos, oferecendo **recomendações de mitigação específicas por fornecedor** (vendor-specific mitigation insights) para reduzir o tempo de remediação.

Destaques:

- biblioteca extensa de TTPs e ameaças com atualizações diárias;
- *reporting* focado em lacunas de controlos e recomendações concretas para firewalls, EDRs, etc.

2.5.3 AttackIQ

A **AttackIQ** posiciona-se como uma das principais plataformas de BAS e é **parceiro fundador do Center for Threat-Informed Defense** da MITRE Engenuity. É frequentemente referida como uma das primeiras empresas a operacionalizar o **MITRE ATT&CK** em validação contínua de segurança.

2.5.4 SafeBreach

A **SafeBreach** define BAS como uma solução altamente automatizada que corre ataques reais de forma segura em ambientes de produção, para validar continuamente a eficácia dos controlos de segurança.

Ponto comum das plataformas BAS:

- foco em **produção** e **validação de controlos defensivos**;
- automação forte na execução de cenários, mas com **lógica de decisão proprietária e não transparente**;
- modelo comercial orientado a grandes organizações.

Comparativamente, o Venom APT:

- atua **exclusivamente em laboratório**,
- expõe explicitamente o **motor de decisão e o modelo de dados**,
- e tem como objetivo principal a **investigação académica**, não a operação em ambientes enterprise.

2.6 Análise Comparativa Estruturada

A Tabela 2.6.1 sintetiza o posicionamento do Venom APT face às principais soluções analisadas, numa escala qualitativa (HIGH/MEDIUM/LOW), considerando sete dimensões relevantes para automação de pentesting em contexto laboratorial académico.

Nota: a classificação decorre de análise própria, com base em documentação pública, relatórios técnicos e artigos sobre cada ferramenta. Não corresponde a métricas oficiais dos fabricantes.

Solução	Autonomous Decision	Full Kill Chain	Academic / Lab Focus	Guardrails & Safety	Scientific Reproducibility	Open / Transparent	Cost / Acessibilidade
Venom APT	HIGH (motor condicional é a inovação central)	HIGH (recon → relatório)	HIGH (desenhado para laboratório académico)	HIGH (allowlist, kill-switch, modos seguros)	HIGH (modelo de dados + KPIs de consistência)	HIGH (arquitetura e lógica documentadas)	HIGH (protótipo de investigação, sem licenças comerciais)
MITRE CALDERA	HIGH (agentes com capacidades de planeamento)	MEDIUM (foco histórico pós-compromisso)	MEDIUM (orientado a uso profissional e investigação)	LOW (safety depende do operador)	MEDIUM (logs e relatórios, sem KPIs científicos por defeito)	HIGH (open-source)	HIGH (software livre, custo operacional em setup)
Atomic Red Team	LOW (execução manual ou via scripts)	LOW (testes atómicos, sem orquestração)	HIGH (muito adequado a labs e formação)	LOW (responsabilidade do operador)	HIGH (testes atómicos reprodíveis)	HIGH (open-source)	HIGH (gratuito)
Metasploit Pro	MEDIUM (auto-exploitation com attack plan)	MEDIUM (foco em exploitation)	MEDIUM (há versão community, Pro é enterprise)	LOW (sem guardrails nativos)	LOW (workflow manual, sujeito a variabilidade)	MEDIUM (framework aberto, Pro fechado)	LOW (licenciamento enterprise)
Cobalt Strike	LOW (human-in-the-loop; automação via scripts)	MEDIUM (forte em pós-exploração)	LOW (orientado a operações enterprise)	LOW (riscos elevados se usado sem controlo)	LOW (operações altamente manuais)	LOW (comercial, fechado)	LOW (licença dispendiosa)
Cymulate BAS	HIGH (execução automatizada de cenários)	HIGH (cobertura ao longo da kill chain)	LOW (SaaS enterprise, não labs air-gapped)	HIGH (concebido para produção com segurança)	MEDIUM/HIGH (cenários repetíveis, mas lógica interna proprietária)	LOW (produto fechado)	LOW (custo típico de plataforma enterprise)
Picus Security	MEDIUM (seleção automatizada de cenários)	HIGH (validação de múltiplas etapas)	LOW (foco enterprise)	HIGH (orientado a produção)	MEDIUM (simulações repetíveis)	LOW (produto fechado)	LOW (licenciamento enterprise)

Solução	Autonomous Decision	Full Kill Chain	Academic / Lab Focus	Guardrails & Safety	Scientific Reproducibility	Open / Transparent	Cost / Acessibilidade
AttackIQ	MEDIUM (biblioteca extensa, execução programável)	HIGH (forte alinhamento ATT&CK)	LOW (foco na validação contínua de controlos)	HIGH (produção)	MEDIUM (cenários consistentes, mas sem foco em ciência)	LOW (solução proprietária)	LOW (segmento enterprise)
SafeBreach	MEDIUM (execução automatizada contra produção)	HIGH (simulação de muitos tipos de ataque)	LOW (não orientado a labs académicos)	HIGH (BAS "production-safe")	MEDIUM (campanhas repetíveis, lógica interna opaca)	LOW (produto fechado)	LOW (licenciamento enterprise)

Tabela 2.6.1 - Análise comparativa qualitativa do Venom APT face a soluções de mercado e frameworks open-source.

2.7 Posicionamento Diferenciado do Venom APT

A análise mostra que o **Venom APT** não procura competir com ferramentas comerciais maduras, mas **preencher um conjunto específico de *gaps*** ainda pouco explorados:

1. **Gap 1 – Autonomous Decision Engine para Full Kill Chain em Laboratório**
    - CALDERA introduz planeamento automático, mas com foco histórico pós-compromisso.
    - Atomic Red Team fornece técnicas individuais, sem orquestração.
    - Metasploit Pro automatiza principalmente a seleção de exploits.
    - Cobalt Strike continua centrado no operador humano.
    - Plataformas BAS automatizam cenários, mas com **lógica de decisão proprietária** e foco defensivo.

O Venom APT propõe um **motor de decisão condicional transparente**, que orquestra todo o ciclo de ataque (reconhecimento, exploração, pós-exploração, relatório) em ambiente de laboratório, com regras explícitas **Se–Então** e *knowledge base* documentada.

2. **Gap 2 – Safety-by-Design para Ambientes Académicos**
    - Ferramentas open-source e C2 frameworks delegam a responsabilidade de *safety* ao operador.
    - Plataformas BAS são *production-safe*, mas desenhadas para organizações empresariais, não para laboratórios académicos isolados.

O Venom APT introduz **guardrails hard-coded**: *allowlists* de IP, kill-switch global, modos restritivos de operação e ambito estritamente limitado ao laboratório definido, alinhados com o *scope* do projeto.

3. **Gap 3 – Reprodutibilidade Científica e KPIs Mensuráveis**

- As soluções existentes medem, sobretudo, eficácia de controlos e cobertura de TTPs; raramente definem **KPIs de reprodutibilidade** e consistência dos fluxos de ataque.
- O Venom APT formaliza KPIs como:
  - **Eficiência temporal** (redução percentual de tempo face a execução manual);
  - **Densidade de cobertura de TTPs** por ciclo;
  - **Reprodutibilidade de cenários** (consistência ≥ 95% em ambientes estáveis);
  - **Conformidade com guardrails**.

4. **Gap 4 – Transparência Académica vs. Black-Box Comercial**
   - Plataformas BAS e versões Pro/Enterprise de ferramentas são, em grande medida, **caixas-pretas**, dificultando análise do *decision engine*.
   - O Venom APT prioriza **documentação da arquitetura, modelo de dados e lógica de decisão**, permitindo auditoria, crítica e extensão pela comunidade científica.
5. **Gap 5 – Barreiras de Custo à Investigação Académica**
   - Ferramentas comerciais de C2 e BAS exigem investimentos significativos, típicos de orçamentos enterprise, o que limita a sua adoção em projetos de TFC ou investigação universitária.
   - O Venom APT é concebido como **protótipo de investigação**, com foco em reutilização e partilha em contexto académico.

2.8 2.8 Enquadramento Teórico-Científico

2.8.1 Automação da Decisão como Problema de Planeamento

Trabalhos da MITRE sobre **Automated Adversary Emulation** argumentam que emular adversários de forma autónoma implica tratar o problema como **planeamento e atuação sob incerteza**, e não apenas como sequência fixa de comandos.

O Venom APT incorpora esta perspetiva ao:

- modelar explicitamente **fases, tarefas, ferramentas e outputs** em entidades de dados;
- implementar um **motor de decisão condicional** que analisa o resultado de cada ação (por exemplo, portas abertas, banners, versões de serviços) e seleciona a próxima técnica com base em regras formais.

2.8.2 Reprodutibilidade Científica em Pentesting

Estudos em outras áreas científicas (por exemplo, Begley & Ellis; Open Science Collaboration) mostram que a falta de documentação estruturada e automação rigorosa conduz a taxas de reprodutibilidade muito baixas. Aplicado ao pentesting, isto significa que **processos altamente manuais e ad hoc são dificilmente repetíveis** com resultados equivalentes.

O Venom APT procura mitigar este problema através de:

- **determinismo na lógica de decisão** (regras if–then-else claramente definidas);
- **logs estruturados automáticos** de todas as decisões, comandos e resultados;
- possibilidade de reexecutar cenários com parâmetros idênticos, permitindo avaliação objetiva de consistência.

2.8.3 MITRE ATT&CK como Framework de Validação

O **MITRE ATT&CK** fornece uma linguagem comum para descrever TTPs adversariais e comparar a cobertura de diferentes ferramentas.

No Venom APT:

- cada módulo de ataque é **mapeado a IDs ATT&CK**;
- torna-se possível gerar **matrizes de cobertura** e comparar o perfil de TTPs executados pelo motor autónomo com perfis típicos de testes manuais ou de outras plataformas.

2.9 Conclusão: Posição Diferenciada e Complementaridade

O benchmarking realizado demonstra que o **Venom APT**:

- **não procura substituir** CALDERA, Atomic Red Team, Metasploit, Cobalt Strike ou plataformas BAS;
- posiciona-se como **complemento académico**, focado em:
    - **motor de decisão autónomo full kill chain em laboratório**,
    - **guardrails de segurança integrados por design**,
    - **modelo de dados e KPIs de reprodutibilidade científica**.

Em síntese:

- **Complementa o CALDERA**, ao enfatizar simplicidade de *setup*, full kill chain em laboratório e guardrails académicos;
- **Complementa o Atomic Red Team**, fornecendo o motor de orquestração que encadeia testes atómicos;
- **Distingue-se de Metasploit Pro e Cobalt Strike**, cujo foco é exploitation / C2 em cenários de produção e operação humana;
- **Distingue-se das plataformas BAS**, ao priorizar transparência académica, laboratório isolado e avaliação científica da automação ofensiva.

Esta posição diferenciada, alinhada com os requisitos e arquitetura definidos nos capítulos seguintes, fundamenta a relevância da contribuição do Venom APT para o state of art em **automação de pentesting em contexto académico e laboratorial**.

3 Viabilidade e Pertinência

Este capítulo analisa a viabilidade e a relevância do projeto Venom APT face ao contexto atual de mercado e às necessidades identificadas no capítulo anterior.
Do ponto de vista de viabilidade, demonstram-se as condições económicas e estruturais para que a solução possa evoluir para além do TFC, enquanto protótipo tecnicamente sólido e potencial base de um produto ou serviço continuado.
Na componente de pertinência, demonstra-se de que forma o Venom APT contribui de forma objetiva para mitigar o problema identificado: a forte dependência de trabalho manual em todas as fases do pentesting e a escassez de profissionais qualificados para responder à procura crescente.

3.1 Viabilidade económica: dimensão de mercado e oportunidade

3.1.1 Mercado global de penetration testing e PTaaS

Diversos estudos de mercado apontam para um crescimento acentuado na área de serviços de segurança ofensiva, em particular no segmento de penetration testing.

De acordo com a Grand View Research, o mercado global de **penetration testing** foi avaliado em cerca de **1,82 mil milhões de USD em 2023** e deverá atingir aproximadamente **5,24 mil milhões de USD em 2030**, o que corresponde a uma **taxa de crescimento anual composta (CAGR) de 16,6%** no período 2024–2030[34].

Em paralelo, o subsegmento de **Penetration Testing as a Service (PTaaS)**(modelo em que os testes são prestados de forma contínua e altamente automatizada) apresenta um potencial de crescimento ainda mais expressivo. Um relatório divulgado pela Global Market Insights estima que o mercado global de PTaaS **ultrapasse os 7,1 mil milhões de USD até 2032**, refletindo uma adoção crescente de modelos "as a service" para testes de segurança recorrentes[35].

Estes números evidenciam que:

- Existe um **mercado em expansão** para soluções relacionadas com pentesting, com expectativas de crescimento significativo até 2030–2032.
- A migração de modelos de prestação pontual para **serviços recorrentes e automatizados (PTaaS)** está alinhada com a proposta de valor do Venom APT, que se posiciona como motor de orquestração automática de ataques em ambientes laboratoriais.

Mesmo capturando apenas uma fração muito reduzida deste mercado global, uma solução baseada no Venom APT teria potencial económico para justificar a sua continuação após o TFC, seja enquanto componente tecnológica integrada em serviços de consultoria, seja como base para um produto especializado em pentesting em laboratório e treino de equipas.

3.1.2 Escassez de talento e pressão para automação

A viabilidade do Venom APT não depende exclusivamente da dimensão do mercado, mas também da **escassez estrutural de profissionais de cibersegurança**, que incentiva a adoção de ferramentas de automação para aumentar produtividade e cobertura.

O estudo ISC2 Cybersecurity Workforce Study 2023 estima que a força de trabalho global de cibersegurança ronda 5,5 milhões de profissionais, mas identifica uma escassez significativa de recursos humanos. De acordo com o relatório, existe um défice de aproximadamente 4 milhões de profissionais necessários para proteger adequadamente os ativos digitais a nível mundial, sendo referido que "the cybersecurity workforce gap remains at approximately 4 million professionals worldwide"[36].

O estudo baseia-se em respostas de 14 865 profissionais de cibersegurança, evidenciando uma amostra estatisticamente robusta.

Esta escassez manifesta-se na prática em:

- dificuldade em recrutar e reter pentesters experientes;
- tempo limitado que as equipas conseguem dedicar a testes repetidos em ambiente de laboratório;
- impossibilidade de executar, com regularidade, campanhas extensivas de ataque controlado em todos os ativos críticos.

Neste contexto, soluções que:

- **reduzem a dependência de decisão manual** em cada etapa do pentest;
- **melhoram a reprodutibilidade** entre execuções;
- e **maximizam o uso do tempo de especialistas** (focando-os na análise e não na execução manual dos passos)

tendem a ser bem recebidas pelo mercado. O Venom APT responde diretamente a esta pressão, automatizando o ciclo de ataque em ambiente controlado, com guardrails de segurança e rastreabilidade total, o que reforça a sua viabilidade económica e operacional.

3.2 Viabilidade de continuidade pós-TFC

A solução foi concebida desde o início com potencial de continuidade após o TFC, e não apenas como protótipo pontual. A viabilidade de evolução sustenta-se em três dimensões principais:

1. **Arquitetura modular e orientada a domínio**
    - O modelo de dados e a arquitetura de três camadas (Gestão de Pentest, Orquestração, Knowledge Base) permitem acrescentar novos módulos ofensivos, integrações com ferramentas de terceiros e novas métricas, sem reescrever o núcleo do sistema.
    - Esta modularidade favorece a transformação do Venom APT num **produto "core" reutilizável**, que pode ser licenciado, expandido ou integrado em plataformas existentes de treino ou laboratório.
2. **Modelo de custos baseado em software e infraestruturas standard**

- O protótipo é desenvolvido predominantemente em **Python** e recorre a **bibliotecas open-source**, bem como a containers (por exemplo, Docker) para isolamento e deploy padronizado.
- Este modelo reduz custos de infraestruturas proprietárias e permite escalar ambientes de laboratório com recurso a hardware ou cloud standard, abrindo caminho a provas de conceito com diferentes organizações sem investimento inicial elevado.

3. **Potenciais modelos de negócio**
   A partir do protótipo, é possível evoluir para vários modelos de continuidade:
   - **Ferramenta interna de consultoria**, utilizada por equipas de red team para estandardizar e acelerar pentests em laboratório, mantendo os scripts proprietários numa base de conhecimento própria.
   - **Serviço PTaaS em ambiente controlado**, em que o Venom APT é utilizado para executar campanhas regulares de ataque em redes de treino ou "digital twins" de clientes, beneficiando da tendência de crescimento observada no mercado de PTaaS.
   - **Plataforma de treino e ensino**, em parceria com instituições de ensino ou centros de formação, permitindo simular cadeias completas de ataque de forma repetível e segura.

Em síntese, a combinação de um mercado em crescimento, escassez de talento especializado e uma arquitetura técnica preparada para evolução suporta a conclusão de que o Venom APT tem **viabilidade real de continuidade pós-TFC**, seja como produto autónomo, seja como componente de serviços de consultoria ou formação.

3.2.1 Continuidade pós-TFC e exploração futura

Para além da entrega do protótipo no âmbito do TFC, o Venom APT foi desenhado para suportar vários cenários de continuidade concretos:

• Módulo de PTaaS interno em contexto de consultoria: o núcleo de orquestração e o modelo de dados podem ser integrados em serviços internos de penetration testing recorrente em ambiente de laboratório ou "digital twins" de clientes, servindo como motor técnico de campanhas PTaaS controladas e repetíveis.

• Base tecnológica para dissertação de mestrado: a arquitetura modular (motor de decisão, guardrails, integração com MITRE ATT&CK e modelo de dados) fornece um ponto de partida sólido para trabalhos de 2.º ciclo, por exemplo em automação avançada de PTaaS, integração com SOC/SIEM ou estudo de métricas de eficácia de ataques em laboratório.

• Utilização pedagógica em disciplinas e laboratórios / eventual abertura parcial: o protótipo pode ser adaptado para utilização em unidades curriculares de cibersegurança e laboratórios do curso, permitindo simular cadeias completas de ataque de forma repetível e

segura. Numa fase posterior, partes não sensíveis da solução poderão ser disponibilizadas em regime open-source controlado, reforçando validação e adoção pela comunidade académica.

Este enquadramento evidencia que o Venom APT não se esgota na defesa do TFC, posicionando-se como ativo tecnológico com aplicabilidade prática em contexto profissional, académico e formativo.

3.3 Pertinência e contributo para a resolução do problema

O capítulo 1 identificou como problema central a **dependência de decisão manual em cada etapa do pentesting**, mesmo em ambientes laboratoriais autorizados, o que tem impacto negativo em:

- eficiência (tempo gasto em tarefas repetitivas),
- cobertura (limitações na quantidade de TTPs exercitados por sessão),
- reprodutibilidade (dificuldade em repetir exatamente o mesmo caminho de ataque),
- e escalabilidade (dificuldade em aumentar a frequência de testes sem aumentar proporcionalmente o número de especialistas).

À luz deste enquadramento, a pertinência do Venom APT assenta em vários contributos concretos:

1. **Automatização orquestrada do ciclo de ataque**
   O Venom APT encadeia, de forma autónoma, as fases de reconhecimento, exploração e pós-exploração, reutilizando os outputs de cada etapa como inputs da fase seguinte, com base num motor de decisão condicional e num modelo de dados explícito. Isto reduz a carga operacional sobre o pentester, que passa a desenhar o perfil de ataque e a interpretar resultados, em vez de executar manualmente cada comando.

2. **Explicabilidade e auditabilidade**
   O mapeamento sistemático das ações à matriz **MITRE ATT&CK** e o registo estruturado de eventos (AUDIT_LOG, ATTACK_PATH, EXECUCAO_FERRAMENTA, etc.) permitem reconstruir a lógica seguida pelo motor de decisão, contribuindo para a transparência e para a robustez do processo de auditoria. Esta capacidade é particularmente relevante num contexto em que o mercado de serviços de segurança está a crescer rapidamente e em que os clientes exigem evidências claras de cobertura e rastreabilidade[37].

3. **Foco em ambientes laboratoriais autorizados**
   Ao restringir explicitamente o âmbito a **ambientes de laboratório e redes de teste**, o projeto reduz riscos operacionais e éticos associados à automação de ataques em ambientes de produção. Esta escolha aumenta a pertinência académica e prática do trabalho, pois possibilita experimentar cenários avançados de ataque com segurança e dentro de um quadro ético claro, ao mesmo tempo que prepara o terreno para evoluções futuras alinhadas com o mercado de PTaaS[38].

4. **Resposta à escassez de profissionais**
   Num contexto em que o estudo ISC2 indica um **défice global de 4 milhões de profissionais de cibersegurança**[39], ferramentas que aumentam a produtividade e a capacidade de teste das equipas existentes contribuem diretamente para mitigar o problema identificado. O Venom APT procura precisamente multiplicar o impacto dos especialistas, automatizando o que é repetitivo e deixando para o humano a análise crítica e a tomada de decisão estratégica.

Desta forma, o projeto não é apenas tecnicamente interessante; ele responde de forma prática e mensurável a desafios reais do setor, o que reforça a sua pertinência e potencial impacto.

3.4 Validação por terceiros e estudo de viabilidade

Para além da análise de mercado e da fundamentação teórica, a viabilidade e pertinência do Venom APT serão validadas através de **feedback direto de stakeholders**, conforme recomendado no regulamento.

Está previsto:

1. **Questionário de viabilidade, interesse e pertinência**
   - Aplicado a um conjunto de potenciais utilizadores e decisores, incluindo pentesters, elementos de equipas Blue Team/SOC, gestores de risco e, quando possível, representantes de empresas prestadoras de serviços de cibersegurança.
   - O questionário irá abordar temas como:
     - grau de dor associado ao esforço manual atual em pentesting;
     - interesse em soluções de orquestração automática em ambiente de laboratório;
     - requisitos mínimos para considerar a adoção de uma ferramenta como o Venom APT;
     - perceção de riscos e barreiras à adoção.
   - A versão final do instrumento e a respetiva fundamentação serão apresentados no **Anexo 1 – Questionário de Viabilidade e Pertinência**.
2. **Análise dos resultados**
   - A análise estatística e qualitativa dos dados recolhidos permitirá validar (ou refutar) os pressupostos de mercado e de problema identificados nos capítulos 1 e 2.
   - Serão particularmente valorizados:
     - o nível de interesse declarado na utilização de uma ferramenta com as características do Venom APT;
     - a perceção de valor acrescentado em termos de eficiência, reprodutibilidade e documentação;

- comentários abertos de especialistas que possam orientar a evolução pós-TFC.

Esta validação por terceiros complementa os indicadores de mercado apresentados anteriormente e reforça a credibilidade do projeto como proposta com **viabilidade económica**, **pertinência operacional** e **potencial de continuidade**.

3.5 Proposta de inovação e mais-valias

Com base no benchmarking efetuado e na análise do state of art em ferramentas de pentesting e PTaaS, o Venom APT posiciona-se não como mais um scanner ou mais uma plataforma de execução remota de testes, mas como um motor de orquestração autónomo especificamente desenhado para ambientes laboratoriais autorizados. A proposta de inovação materializa-se nas seguintes mais-valias principais:

• Orquestração condicional ponta-a-ponta: o Venom APT encadeia automaticamente as fases de reconhecimento, exploração e pós-exploração através de um motor de decisão baseado em regras "Se–Então", reduzindo a necessidade de microgestão humana e diferenciando-se de soluções que apenas automatizam scans isolados.

• Knowledge base estruturada e orientada a KPIs: todo o ciclo de ataque é registado numa base de dados relacional (PENTEST, EXECUCAO_FERRAMENTA, ATTACK_PATH, KPI_RESULT, entre outras), permitindo reconstituir o caminho de ataque, medir eficiência, cobertura e reprodutibilidade, e comparar execuções automatizadas com abordagens manuais.

• Guardrails éticos e segurança operacional by design: a solução integra, de raiz, mecanismos de allowlist, kill-switch global, auto-proteção da máquina de controlo, verificação de integridade de código e modos de exploração "safe check"/"dry run", mitigando riscos de uso indevido e de impacto nos ambientes laboratoriais.

• Integração nativa com MITRE ATT&CK e reporting avançado: todas as ações ofensivas são mapeadas à matriz MITRE ATT&CK, com capacidade de exportação para MITRE Navigator e geração de relatórios técnicos/forenses detalhados, algo pouco explorado em ferramentas focadas apenas na execução de exploits.

• Foco em ambientes laboratoriais e treino de equipas: ao privilegiar redes de teste e "digital twins", o Venom APT é particularmente adequado para cenários de treino, simulação e PTaaS em laboratório, permitindo repetir campanhas de ataque completas com elevado grau de controlo e reprodutibilidade.

Este conjunto de características diferencia a solução face às alternativas analisadas e reforça a sua relevância enquanto base tecnológica para serviços de consultoria, plataformas de treino ou ofertas PTaaS especializadas em ambientes laboratoriais.

4 Solução Proposta

Este capítulo descreve a concretização técnica do Venom APT, desde a metodologia de desenvolvimento até às tecnologias adotadas e ao modelo de dados que suporta a recolha e análise dos resultados experimentais. A solução materializa a visão definida para uma ferramenta autónoma de pentesting em ambientes laboratoriais autorizados, focada em automatizar, de ponta a ponta, o ciclo de ataque com guardrails de segurança e capacidade de auditoria total.

A implementação da solução mobiliza conhecimentos de várias áreas científicas do curso, nomeadamente cibersegurança e segurança ofensiva, redes de computadores e protocolos, sistemas operativos, bases de dados e engenharia de software. Na versão final do relatório serão explicitadas as unidades curriculares concretas correspondentes a estas áreas.

4.1 Introdução

O Venom APT foi concebido para responder a um problema concreto identificado nos capítulos anteriores: a forte dependência de decisão manual em cada etapa do pentesting, mesmo em contexto laboratorial controlado, o que degrada eficiência, cobertura, reprodutibilidade e escalabilidade. A solução proposta substitui a microgestão humana por um motor de decisão autónomo, capaz de orquestrar um fluxo completo de ataque – do reconhecimento à geração de relatório – com base em lógica condicional e mapeamento sistemático à matriz MITRE ATT&CK.

Neste capítulo são descritos:

- a metodologia de desenvolvimento adotada (epics, features, user stories, DoR/DoD);
- a arquitetura lógica e o modelo de dados que sustentam o protótipo;
- o ecossistema de tecnologias e ferramentas ofensivas/defensivas integradas;
- o processo de recolha, tratamento e análise exploratória dos dados que serão usados para validar os KPIs definidos (tempo, cobertura, reprodutibilidade e segurança).

A **Figura 4.1.1** apresenta o diagrama de contexto do Venom APT, evidenciando as principais personas envolvidas (Pentester/Red Team, Blue Team/SOC, Auditor e *Client Owner*), o ambiente alvo de laboratório e as fronteiras do sistema relativamente às ferramentas externas e à organização de segurança.

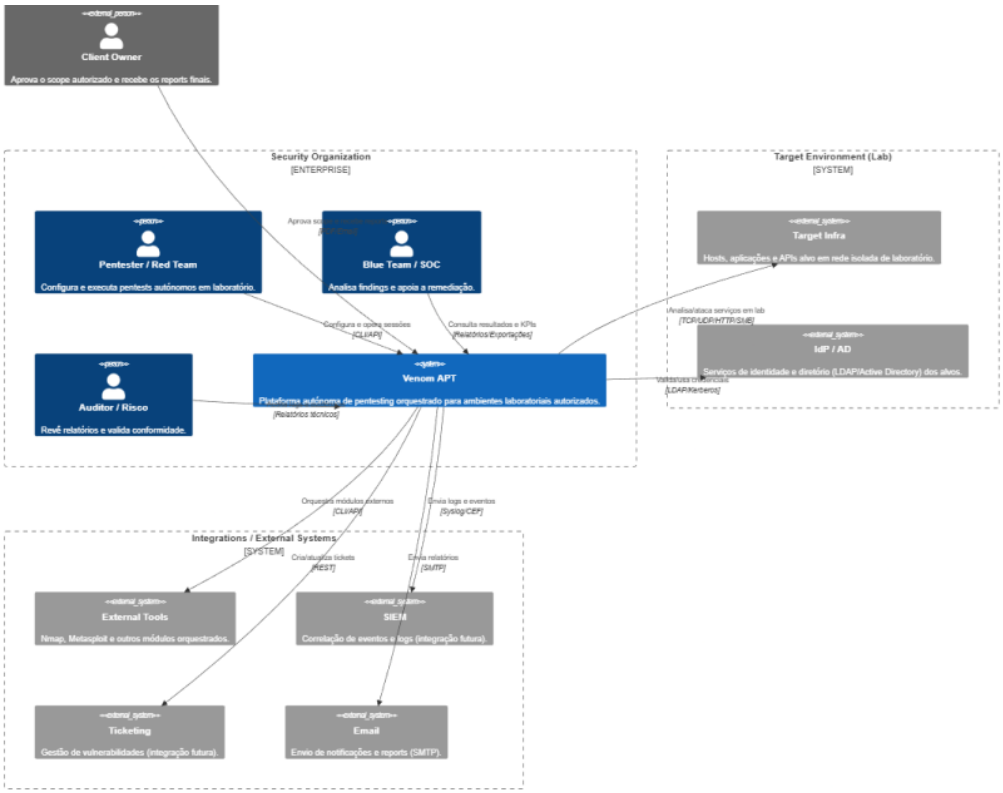

Figura 4.1.1 - Diagrama de contexto / arquitetura lógica de alto nível do Venom APT (C4).

4.2 Metodologia

A metodologia de desenvolvimento do Venom APT combina engenharia de requisitos, modelação orientada ao domínio, boas práticas de desenvolvimento iterativo e um plano de avaliação alinhado com KPIs quantitativos.

A abordagem seguiu, em síntese, os seguintes passos principais:

1. **Derivação de requisitos** a partir da visão, problema, objetivos, restrições e KPIs do projeto (requisitos funcionais RF01–RF10 e não funcionais RNF01–RNF07).
2. **Agrupamento em Epics (EP01–EP05)** e decomposição em Features e User Stories, com mapeamento explícito para o modelo de dados e para a matriz MITRE ATT&CK.
3. **Definição de critérios globais de qualidade (DoR/DoD/AC)** a aplicar transversalmente a todas as stories.
4. **Modelação da arquitetura lógica e do modelo de dados** em três camadas (Gestão de Pentest, Orquestração, Knowledge Base), produzindo os diagramas de arquitetura e ER.
5. **Planeamento da avaliação experimental**, definindo que dados são recolhidos e como serão tratados para medir os KPIs de tempo, cobertura, reprodutibilidade e conformidade com guardrails.

4.2.1 Modelo de requisitos (Requisitos Funcionais e Não Funcionais)

A Tabela 4.2.1 sintetiza os **Requisitos Funcionais (RF)** do Venom APT, que traduzem a visão e o âmbito do projeto em capacidades concretas da ferramenta.

ID	Nome	Descrição
RF01	Reconhecimento Ativo	O sistema deve executar reconhecimento e descoberta de ativos para construir um inventário dinâmico dos alvos dentro do segmento de rede autorizado, identificando hosts, portos abertos e serviços.
RF02	Identificação de Vulnerabilidades	O sistema deve identificar vulnerabilidades e más configurações conhecidas, baseando-se em assinaturas de serviço e CVEs, limitando-se ao conjunto de técnicas definidas no âmbito.
RF03	Motor de Decisão Condicional	O sistema deve possuir um motor de decisão capaz de escolher autonomamente o próximo passo de ataque com base no sucesso/falha da etapa anterior e nos atributos do alvo (lógica Se–Então).
RF04	Mapeamento MITRE ATT&CK	O sistema deve mapear automaticamente todas as ações executadas (findings e táticas) aos IDs oficiais da matriz do MITRE ATT&CK.
RF05	Exploração Controlada	O sistema deve validar a exploração de vulnerabilidades críticas através de provas de conceito inofensivas (*safe checks* ou *flag creation*), evitando danos à integridade do sistema alvo.
RF06	Movimento Lateral e Persistência	O sistema deve suportar técnicas autorizadas de movimento lateral (ex.: reutilização de credenciais) e persistência básica, quando permitido pela configuração da sessão.
RF07	Orquestração de Cadeias de Ataque	O sistema deve permitir o encadeamento automático de múltiplas etapas (Reconhecimento → Exploração → Pós-Exploração), reutilizando *outputs* como *inputs* para as fases seguintes.
RF08	Gestão de Sessões	O sistema deve permitir a configuração, início, pausa e paragem de ciclos de teste, com definição clara de janelas de tempo e alvos.

ID	Nome	Descrição
RF09	Relatórios e Evidências	O sistema deve produzir relatórios técnicos e executivos ao final de cada sessão, detalhando os vetores de ataque bem-sucedidos, o mapeamento MITRE e as evidências recolhidas.
RF10	Segurança Operacional	O sistema deve validar cada pacote ou comando contra uma lista de restrições (allowlist de IPs) antes da execução, bloqueando qualquer interação fora do âmbito.

Tabela 4.2.1 - Requisitos Funcionais (RF01–RF10)

Os **Requisitos Não Funcionais (RNF)** definem propriedades de qualidade e constrangimentos globais de arquitetura, incluindo isolamento de rede, gestão segura de segredos, auditabilidade, reprodutibilidade, usabilidade, limpeza de dados e deployabilidade via containers. A **Tabela 4.2.2** resume estes requisitos.

ID	Nome	Descrição
RNF01	Isolamento de Rede	O sistema deve ser arquitetado para operar em total isolamento (*air-gapped* ou VLAN dedicada), sem dependências críticas da Internet pública durante a fase de ataque.
RNF02	Gestão de Segredos em Memória	O sistema não deve armazenar credenciais capturadas ou chaves de acesso em texto claro em logs persistentes, utilizando gestão segura de variáveis em memória volátil.
RNF03	Auditabilidade e Logging	O sistema deve manter um registo de auditoria (logging) imutável e detalhado de todas as decisões tomadas pelo motor e de todos os comandos enviados para a rede.
RNF04	Determinismo e Reprodutibilidade	O sistema deve garantir que, dada a mesma configuração inicial (seed/perfil) e o mesmo ambiente alvo, a lógica de decisão tende para os mesmos resultados (determinismo).
RNF05	Usabilidade para Operadores	O sistema deve privilegiar uma interface de linha de comandos (CLI) robusta ou um *dashboard* simples para o operador, que forneça feedback claro em tempo real sobre o estado da sessão.
RNF06	Limpeza de Dados	O sistema deve possuir mecanismos para limpar evidências locais (logs temporários) e ficheiros temporários após a conclusão da sessão e geração do relatório.

ID	Nome	Descrição
RNF07	Deployabilidade	O sistema deve ser empacotado em containers (ex.: Docker) para facilitar a instalação rápida e padronizada em ambientes laboratoriais.

Tabela 4.2.2 - Requisitos Não Funcionais (RNF01–RNF07)

Este modelo de requisitos está diretamente alinhado com os objetivos e restrições definidos na visão do projeto (opera apenas em ambientes laboratoriais autorizados, não integra ML/IA generativa, não abrange redes de produção).

4.2.2 Epics, Features e User Stories

Para estruturar o desenvolvimento e garantir o seguimento entre requisitos e implementação, os requisitos foram agrupados em cinco **Epics (EP01–EP05)**, apresentados na **Tabela 4.2.3.**

ID	Epic	Descrição
EP01	Reconhecimento e Descoberta de Ativos	Módulos focados na identificação da superfície de ataque.
EP02	Motor de Decisão e Orquestração	Núcleo lógico do sistema que gere o fluxo de ataque e o estado da sessão.
EP03	Arsenal de Ataque (Exploração e Pós-Exploração)	Conjunto de módulos ofensivos (exploits e TTPs) mapeados ao MITRE ATT&CK.
EP04	Segurança e Governação (Guardrails)	Mecanismos de controlo de limites, segurança e gestão de identidade da ferramenta.
EP05	Reporting e Auditoria	Funcionalidades de geração de saídas estruturadas, relatórios e visualização de dados.

Tabela 4.2.3 - Epics do Venom APT (EP01–EP05)

Cada Epic é decomposto em **Features (FEAT-EPxx-yy)**, que por sua vez são detalhadas em **User Stories** com DoR/DoD e critérios de aceitação. A **Tabela 4.2.4** apresenta um excerto representativo deste mapeamento.

Epic	Feature	Objetivo da Feature
EP01	FEAT-EP01-01 – Varrimento de rede	Descobrir hosts ativos no âmbito autorizado (ICMP/ARP/TCP SYN).
EP01	FEAT-EP01-02 – Enumeração de portas	Identificar portas/serviços e versões, construindo a fingerprint de cada host.
EP01	FEAT-EP01-03 – Classificação de ativos	Classificar automaticamente hosts (DC, Web, DB, Legacy) com base nos serviços detetados.
EP01	FEAT-EP01-04 – Exportação de inventário	Exportar inventário de hosts/serviços para JSON e visualização em tabela CLI.
EP02	FEAT-EP02-01 – Perfis de ataque	Carregar perfis de ataque via YAML, definindo táticas e módulos autorizados por sessão.
EP02	FEAT-EP02-02 – Motor Se–Então	Selecionar dinamicamente o próximo módulo com base em pré-requisitos e resultados anteriores.
EP02	FEAT-EP02-03 – Knowledge Base	Manter memória de credenciais, hosts e descobertas reutilizáveis durante a sessão.
EP02	FEAT-EP02-04 – Encadeamento/Pivoting	Permitir pivoting lógico, descobrindo novos alvos após comprometer um host intermédio.
EP03	FEAT-EP03-01 – Exploração (Exploits)	Disponibilizar módulos de exploração para vulnerabilidades clássicas de laboratório.
EP03	FEAT-EP03-02 – Safe Check	Validar exploração com *safe checks* e modo “Dry Run” sem payload malicioso real.
EP03	FEAT-EP03-03 – Movimento lateral	Reutilizar credenciais para autenticação remota em hosts vizinhos.
EP03	FEAT-EP03-04 – Persistência	Criar mecanismos de persistência reversível e rotinas de *cleanup* automático.
EP04	FEAT-EP04-01 – Allowlist	Implementar lista branca de CIDRs e auto-proteção da máquina de controlo.
EP04	FEAT-EP04-02 – Kill-switch global	Permitir interrupção imediata da sessão e dos agentes remotos.
EP04	FEAT-EP04-03 – Integridade	Validar integridade de scripts/módulos com hashes no arranque.

Epic	Feature	Objetivo da Feature
EP05	FEAT-EP05-01 – Relatório técnico	Gerar relatório técnico em PDF com severidades e evidências.
EP05	FEAT-EP05-02 – Matriz MITRE	Produzir ficheiro JSON compatível com MITRE Navigator para visualização de TTPs exercitados.
EP05	FEAT-EP05-03 – Attack Path	Representar graficamente a cadeia de compromisso entre "patient zero" e alvo final.
EP05	FEAT-EP05-04 – Logs forenses	Exportar logs brutos em CSV/JSON lines para análise forense e auditoria.

Tabela 4.2.4 - Features por Epic (FEAT-EP01-01 – FEAT-EP05-04)

O backlog de User Stories foi construído a partir destas Features. A **Tabela 4.2.5** apresenta um excerto representativo (resumo) com, para cada User Story, a associação ao Epic e Feature, persona, DoR, DoD e critérios de aceitação positivos/negativos.

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
US-EP01-01.001	EP01 – Reconhecimento e Descoberta	FEAT-EP01-01 – Varrimento de Rede	Operador	Definir intervalo de IPs (CIDR) no ficheiro de configuração.	Estrutura YAML definida; biblioteca ipaddress disponível.	Parser de CIDR implementado; testes de validação de input criados.	Aceita 192.168.1.0/24 e expande para 254 hosts.	Rejeita 192.168.1.999/strings inválidas com "Invalid CIDR Format" e aborta o arranque.
US-EP01-01.002	EP01 – Reconhecimento e Descoberta	FEAT-EP01-01 – Varrimento de Rede	Motor de Execução	Identificar hosts ativos via ICMP, ARP e TCP SYN.	Scapy configurado; permissões de root/admin para raw sockets verificadas.	Função de discovery multi-protocolo funcional.	Deteta host que bloqueia ICMP mas responde a ARP.	Em falta de permissões, lança "Permission Denied: Root required for ARP/SYN scan" em vez de falhar silenciosamente.
US-EP01-01.003	EP01 – Reconhecimento e Descoberta	FEAT-EP01-01 – Varrimento de Rede	Motor de Execução	Ignorar hosts sem resposta após nº configurável de tentativas.	Parâmetros timeout e retries definidos na configuração.	Implementação assíncrona/threads que respeitam o timeout.	Scan a IP inexistente termina após timeout * retries segundos.	Não deixa sockets abertos indefinidamente nem provoca leaks de recursos.
US-EP01-	EP01 – Reconhecim	FEAT-EP01-01 –	Operador	Ajustar velocidade do varrimento	Mecanismo de rate limiting (Token Bucket ou	Envio de pacotes respeita	Com 10 pacotes/s, scan de 100	Velocidade 0 ou negativa gera erro de validação e

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
01.004	ento e Descoberta	Varrimento de Rede		(pacotes/s) para não saturar a rede.	sleep) desenhado.	intervalo configurado.	hosts demora no mínimo 10 s.	reverte para valor *default*.
US-EP01-02.001	EP01 – Reconhecimento e Descoberta	FEAT-EP01-02 – Enumeração de Portas e Serviços	Motor de Execução	Varrer as 1000 portas mais comuns nos hosts descobertos.	Lista de Top 1000 portas carregada.	Scanner TCP Connect/SYN funcional.	Identifica corretamente portas abertas (80, 443) e fechadas num alvo padrão.	Distingue portas "Filtradas" (sem resposta) sem as marcar como fechadas.
US-EP01-02.002	EP01 – Reconhecimento e Descoberta	FEAT-EP01-02 – Enumeração de Portas e Serviços	Motor de Execução	Capturar o banner dos serviços encontrados.	Função de leitura de socket pós-conexão definida.	Banners são guardados na BD associados ao serviço.	Captura Apache/2.4.41 (Ubuntu) na porta 80.	Se o serviço não envia dados (timeout), regista "Unknown Service" e prossegue.
US-EP01-02.003	EP01 – Reconhecimento e Descoberta	FEAT-EP01-02 – Enumeração de Portas e Serviços	Auditor	Registar duração exata de cada scan por host.	Campos start_time e end_time definidos no modelo de dados.	Logs incluem delta temporal calculado.	Log mostra Host 192.168.1.5 scanned in 4.23s.	Duração nunca é negativa ou nula (precisão de ms).
US-EP01-03.001	EP01 – Reconhecimento e Descoberta	FEAT-EP01-03 – Classificação Automática	Motor de Decisão	Classificar host como Domain Controller com base em portas Kerberos/LDAP/DNS.	Regra lógica definida: portas 88 + 389 + 53 = DC.	Atributo role do host atualizado pela regra.	Host com as 3 portas abertas recebe tag ROLE: DC.	Host só com DNS (53) não é marcado como DC.
US-EP01-03.002	EP01 – Reconhecimento e Descoberta	FEAT-EP01-03 – Classificação Automática	Motor de Decisão	Classificar host como Legacy/Obsoleto com base em versões de SO.	Lista de versões obsoletas (Windows XP, Server 2003, SMBv1) definida.	Regex de comparação implementada nos banners.	Banner Windows 5.1 aciona tag OS: Legacy.	Versões modernas ou desconhecidas não recebem tag de Legacy.
US-EP01-03.003	EP01 – Reconhecimento e Descoberta	FEAT-EP01-03 – Classificação Automática	Operador	Corrigir manualmente a classificação de um ativo no inventário.	CLI ou ficheiro de override definido.	Sistema lê overrides e atualiza memória antes do ataque.	Operador pode forçar IP X a "DB Server" e essa info é usada na fase seguinte.	Sistema avisa se o IP manual não existe no inventário de discovery.
US-EP01-04.001	EP01 – Reconhecimento e Descoberta	FEAT-EP01-04 – Exportação	Auditor	Exportar inventário final de ativos em JSON estruturado.	Schema JSON definido.	Ficheiro gerado validado contra o schema.	inventory.json contém array de hosts com IPs e portas.	Erro de I/O (disco cheio) é tratado e logado sem crash do sistema.

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
US-EP01-04.002	EP01 – Reconhecimento e Descoberta	FEAT-EP01-04 – Exportação	Operador	Visualizar tabela resumo no terminal (CLI).	Biblioteca de tabelas CLI (tabulate/rich) definida.	Output formatado e legível na CLI.	Tabela alinhada mostra IP, Hostname, OS, Ports.	Tabela não quebra layout com hostnames muito longos (truncar).
US-EP02-01.001	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-01 – Perfis de Ataque	Operador	Carregar perfil de ataque via YAML que define módulos/táticas ativos.	Estrutura do perfil YAML definida (chaves, módulos, modos).	Motor configura job queue com base no YAML.	Perfil "ReconOnly" não agenda nenhum exploit.	Perfil vazio/corrompido impede início da sessão.
US-EP02-01.002	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-01 – Perfis de Ataque	Sistema	Validar sintaxe do perfil no arranque.	Validador de schema YAML disponível.	Validação executada no init.	Ficheiros corretos passam validação.	Erro descritivo "Key 'modules' missing in profile" é apresentado para perfis inválidos.
US-EP02-02.001	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-02 – Motor Lógico Condicional	Motor de Decisão	Verificar pré-requisitos na Knowledge Base antes de agendar módulo (Se–Então).	Mapeamento módulo ↔ requisitos (porta/OS/creds) definido.	Motor verifica pré-requisitos antes de agendar.	Exploit SMB só corre se porta 445 estiver aberta.	Exploit não corre se porta estiver fechada ou desconhecida.
US-EP02-02.002	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-02 – Motor Lógico Condicional	Motor de Decisão	Evitar executar o mesmo módulo duas vezes no mesmo alvo.	Histórico de execuções (hash Modulo+Alvo) definido.	Verificação de duplicados na fila implementada.	Segunda tentativa de agendar o mesmo exploit é descartada.	Reexecução só é permitida se flag force_retry estiver ativa.
US-EP02-02.003	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-02 – Motor Lógico Condicional	Motor de Decisão	Detetar e impedir loops lógicos nas cadeias de decisão.	Grafo de dependências ou limite de profundidade definido.	Contador de recursividade implementado.	Cadeia A->B->A é interrompida e aviso registado.	Cadeias longas legítimas (A->B->C->D) não são indevidamente bloqueadas.
US-EP02-03.001	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-03 – Knowledge Base	Sistema	Armazenar credenciais/hashes capturados em memória para reutilização.	Estrutura de dados segura (dicionário protegido) definida.	Dados acessíveis globalmente aos módulos.	Credencial capturada no Host A fica disponível para ataque ao Host B.	Credenciais não são escritas em texto claro em logs de disco.
US-EP02-03.003	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-03 – Knowledge Base	Sistema	Limpar dados sensíveis da memória ao encerrar.	Destrutor ou função de limpeza desenhada.	del/memset executado sobre variáveis sensíveis.	Dump de memória pós-execução não mostra passwords.	Mesmo em caso de crash, rotina de limpeza é chamada via try/finally sempre que possível.

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
US-EP02-04.001	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-04 – Encadeamento e Pivoting	Motor de Decisão	Identificar novos alvos visíveis após comprometer máquina e adicioná-los ao discovery.	Parser de tabelas ARP/routing do alvo comprometido definido.	Novos IPs são adicionados à fila de discovery.	Ao comprometer o gateway, descobre rede interna 10.0.0.x.	Não adiciona IPs já presentes no âmbito inicial (evita duplicados).
US-EP02-04.002	EP02 – Motor de Decisão e Orquestração	FEAT-EP02-04 – Encadeamento e Pivoting	Operador	Definir profundidade máxima de pivoting.	Parâmetro max_depth definido na configuração.	Contador de saltos passado entre módulos.	Com max_depth =1, sistema não ataca "netos" do alvo inicial.	Profundidade 0 significa "apenas alvos diretos"; valores inválidos são rejeitados.
US-EP03-01.001	EP03 – Arsenal de Ataque	FEAT-EP03-01 – Exploração (Exploits)	Penteste r	Tentar explorar vulnerabilidades clássicas (ex.: MS17-010).	Script Python do exploit validado.	Módulo retorna sucesso/falha e sessão (quando aplicável).	Obtém shell em Windows 7 vulnerável.	Em Windows 10 não vulnerável, reporta falha e limpa sockets.
US-EP03-01.002	EP03 – Arsenal de Ataque	FEAT-EP03-01 – Exploração (Exploits)	Motor de Execução	Interromper exploit se alvo não responder (timeout).	Timeout definido no socket.	Tratamento de exceção SocketTimeout implementado.	Thread é libertada após X segundos de silêncio.	Não deixa sockets "zombie" abertos no SO do atacante.
US-EP03-01.003	EP03 – Arsenal de Ataque	FEAT-EP03-01 – Exploração (Exploits)	Sistema	Usar payloads não destrutivos como prova de conceito.	Payload benigno definido (ex.: echo pwned > C:\proof.txt).	Validação de que payloads destrutivos (rm -rf, etc.) não são utilizados.	Cria ficheiro de prova e termina sem impacto destrutivo.	Não provoca reinício/BSOD nem danos permanentes no alvo.
US-EP03-02.001	EP03 – Arsenal de Ataque	FEAT-EP03-02 – Validação "Safe Check"	Operador	Executar modo "Dry Run" sem enviar payloads de exploit.	Flag dry_run prevista na configuração.	Módulos implementam ramo if dry_run: return SimulatedSuccess.	Sistema reporta "Vulnerável (Simulado)" sem tráfego de exploit.	Em modo Dry Run, nenhum pacote de exploit é enviado pela interface de rede.
US-EP03-02.002	EP03 – Arsenal de Ataque	FEAT-EP03-02 – Validação "Safe Check"	Auditor	Distinguir nos logs vulnerabilidade confirmada vs inferida.	Tipos de evento de log definidos (campos de estado).	Logs usam status="CONFIRMED" vs status="POTENTIAL".	Relatório final separa vulnerabilidades confirmadas das inferidas.	Nunca reporta "Sucesso" se apenas foi inferência sem validação.
US-EP03-03.001	EP03 – Arsenal de Ataque	FEAT-EP03-03 – Movimento Lateral	Motor de Execução	Tentar autenticação WMI/SMB em hosts vizinhos com	Biblioteca impacket integrada; lista de hosts vizinhos disponível.	Iteração sobre hosts vizinhos implementada com tratamento de erros.	Login bem-sucedido num vizinho permite propagar o agente.	Falhas de login são tratadas sem bloquear contas (respeito pela política de lockout).

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
				credenciais capturadas.				
US-EP03-04.001	EP03 – Arsenal de Ataque	FEAT-EP03-04 – Persistência	Motor de Execução	Criar utilizador/tarefa de persistência com tag específica Venom_*.	Convenção de nomes (Venom_*) definida.	Código de criação de persistência usa a convenção.	Tarefa agendada chama-se Venom_Update em vez de Update.	Não substitui tarefas de sistema existentes.
US-EP03-04.002	EP03 – Arsenal de Ataque	FEAT-EP03-04 – Persistência	Operador	Executar módulo de cleanup no final da sessão.	Scripts de reversão para cada técnica de persistência definidos.	Cleanup remove os artefactos de persistência criados.	Utilizador Venom_User desaparece após o cleanup.	Se remoção falhar, é gerado alerta crítico "MANUAL CLEANUP REQUIRED".
US-EP04-01.001	EP04 – Segurança e Governação	FEAT-EP04-01 – Allowlist	Módulo de Segurança	Intercetar pacotes e verificar allowlist de IPs.	Classe wrapper de socket definida.	Todos os módulos usam o wrapper em vez de sockets nativos.	Conexão para IP autorizado prossegue normalmente.	Conexão para IP não autorizado levanta SecurityGuardrailException.
US-EP04-01.002	EP04 – Segurança e Governação	FEAT-EP04-01 – Allowlist	Módulo de Segurança	Bloquear conexões para porta 22 da máquina de controlo (self-protection).	IP da máquina local identificado automaticamente.	Regra hardcoded de auto-proteção aplicada.	Tentativa de atacar a própria máquina é bloqueada.	Tráfego legítimo de loopback (localhost) necessário ao sistema não é bloqueado.
US-EP04-01.003	EP04 – Segurança e Governação	FEAT-EP04-01 – Allowlist	Auditor	Registar log de “Security Violation”.	Canal de log de segurança dedicado definido.	Violações são gravadas em security.log.	Registo inclui IP origem, IP destino bloqueado e timestamp.	Log não pode ser desativado por configuração.
US-EP04-02.001	EP04 – Segurança e Governação	FEAT-EP04-02 – Kill-Switch Global	Operador	Parar instantaneamente todos os processos (Ctrl+C) de forma controlada.	Signal handlers (SIGINT) definidos.	Encerramento gracioso mas rápido implementado.	Processo termina em < 2 segundos.	Não deixa processos órfãos a correr em background.
US-EP04-02.002	EP04 – Segurança e Governação	FEAT-EP04-02 – Kill-Switch Global	Sistema	Enviar sinal de terminação aos agentes remotos.	Protocolo de C2 suporta comando DIE.	Kill-switch envia DIE antes de fechar sockets.	Agentes remotos desinstalam-se/terminam após comando.	Se rede falhar, agente tem timeout próprio para terminar sozinho.

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
US-EP04-03.001	EP04 – Segurança e Governação	FEAT-EP04-03 – Integridade	Sistema	Calcular hash dos scripts no arranque.	Manifesto de hashes gerado na build.	Verificação de integridade executada antes do arranque.	Arranque prossegue se hashes coincidirem.	Arranque aborta se exploit.py (ou outro script) tiver hash alterado.
US-EP04-03.002	EP04 – Segurança e Governação	FEAT-EP04-03 – Integridade	Operador	Confirmar explicitamente o âmbito (Y/N) antes de iniciar ataque.	Prompt de input no CLI definido.	Execução pausa até receber input válido.	Continuação só ocorre se utilizador digitar "Y".	Programa termina se utilizador digitar "N" ou outra tecla.
US-EP05-01.001	EP05 – Reporting e Análise	FEAT-EP05-01 – Relatório Técnico	Penetester	Gerar relatório técnico em PDF com severidades.	Template PDF desenhado.	Geração de PDF com dados da sessão implementada.	Vulnerabilidades críticas aparecem no topo com cor destacada.	Caracteres especiais no nome do host não quebram a geração do PDF.
US-EP05-01.002	EP05 – Reporting e Análise	FEAT-EP05-01 – Relatório Técnico	Cliente	Ver evidências (outputs de comandos) no relatório.	Outputs armazenados na Knowledge Base.	Outputs incluídos como blocos de código no PDF.	Output de whoami visível no relatório.	Outputs binários ou demasiado longos são truncados para preservar a formatação.
US-EP05-02.001	EP05 – Reporting e Análise	FEAT-EP05-02 – Matriz MITRE ATT&CK	Analista	Obter ficheiro JSON importável no MITRE Navigator com mapeamento visual das técnicas.	JSON layer do MITRE Navigator compreendido.	Ficheiro JSON gerado compatível com Navigator.	Importar JSON no site MITRE mostra cores corretas para técnicas usadas.	Técnicas não usadas aparecem sem cor/realce.
US-EP05-03.001	EP05 – Reporting e Análise	FEAT-EP05-03 – Visualização de Attack Path	Operador	Visualizar graficamente a cadeia de compromisso (attack path).	Estrutura de grafo (nodes/edges) definida.	Representação em texto (ASCII tree) ou imagem gerada.	Mostra, por exemplo, Internet -> Host A -> Host B.	Ciclos no grafo são tratados corretamente na visualização.
US-EP05-03.002	EP05 – Reporting e Análise	FEAT-EP05-03 – Visualização de Attack Path	Auditor	Saber quem é o “Patient Zero” no caminho de ataque.	Identificação do nó raiz definida.	Destaque visual/textual do primeiro compromisso implementado.	Relatório identifica claramente o vetor de entrada inicial.	(Não explicitado na definição original; quando não há raiz inequívoca, aplica-se o comportamento definido nos AC globais de robustez.)
US-EP05-04.001	EP05 – Reporting e Análise	FEAT-EP05-04 – Logs Forenses	Investigador	Aceder aos logs brutos da sessão em formatos adequados a	Formatos CSV/JSON Lines definidos.	Ficheiro session.log persistido no disco.	Logs contêm timestamp, IP origem, IP destino,	Logs não contêm passwords ou dados confidenciais em claro (aplicada redação).

ID	Epic	Feature	Persona	User Story (resumo)	DoR (resumo)	DoD (resumo)	AC Positivo (resumo)	AC Negativo (resumo)
				análise forense.			porta, protocolo, tamanho do payload.	
US-EP05-04.002	EP05 – Reporting e Análise	FEAT-EP05-04 – Logs Forenses	Sistema	Rodar logs entre sessões, criando ficheiro novo por execução.	Naming convention session_TIMESTAMP.log definida.	Novo ficheiro de log criado em cada arranque.	Execuções consecutivas criam ficheiros distintos.	Sistema verifica espaço em disco antes de criar novo log e reage a erros de falta de espaço.

Tabela 4.2.5 - Backlog de User Stories (resumo)

4.2.3 Critérios de qualidade (DoR, DoD e critérios globais de aceitação)

Foram definidos critérios globais de **Definition of Ready (DoR)** e **Definition of Done (DoD)** aplicáveis a todas as User Stories:

- **DoR – READY**
  Uma User Story é considerada pronta a iniciar se:
  - estiver no formato "Como [Persona], quero [Ação], para [Benefício]";
  - tiver dependências técnicas mapeadas e plano para as satisfazer;
  - tiver avaliação prévia de risco operacional (caso inclua ações ofensivas);
  - estiver alinhada com o MITRE ATT&CK quando representa uma ação ofensiva;
  - possuir critérios de aceitação definidos para cenário de sucesso e de erro.
- **DoD – DONE**
  Uma User Story é considerada concluída se:
  - o código estiver documentado e em conformidade com PEP8;
  - a lógica estiver coberta por testes unitários adequados;
  - a funcionalidade tiver sido validada num ambiente de laboratório controlado;
  - respeitar os guardrails (allowlist, kill-switch, integridade);
  - estiver instrumentada com logging estruturado;
  - incluir rotinas de limpeza (ficheiros temporários, sockets, artefactos criados);
  - a documentação técnica (README/Wiki) estiver atualizada.

Foram ainda definidos **critérios de aceitação globais** para garantir robustez no tratamento de erros (graceful degradation), não bloqueio da interface de controlo e sanitização sistemática de dados sensíveis em logs.

4.2.4 Integração com a arquitetura e modelo de dados (Figuras 4.1 e 4.2)

A modelação de requisitos e o backlog estão alinhados com uma arquitetura em **três camadas**:

- **Camada de Gestão de Pentest (Management Layer):** entidades USER, ENVIRONMENT, GUARDRAILS, PENTEST.
- **Camada de Orquestração (Orchestration Layer):** FASE_PENTEST, EXECUCAO_FASE, TAREFA_FASE, DECISAO_AUTOMATICA, FERRAMENTA, EXECUCAO_FERRAMENTA.
- **Camada de Alvos e Knowledge Base (Knowledge Base Layer):** VITIMA, HOST, SERVICE, VULNERABILITY, CREDENTIAL, ATTACK_PATH, ATTACK_PATH_STEP, TECHNIQUE, FICHEIRO_GERADO, OUTPUT_FERRAMENTA, KPI_RESULT, AUDIT_LOG, entre outras.

A **Figura 4.2.1** representa o fluxo lógico de orquestração do Venom APT, desde a definição do PENTEST e dos GUARDRAILS até à execução das fases, invocação de ferramentas externas, registo de resultados e consolidação dos KPIs.

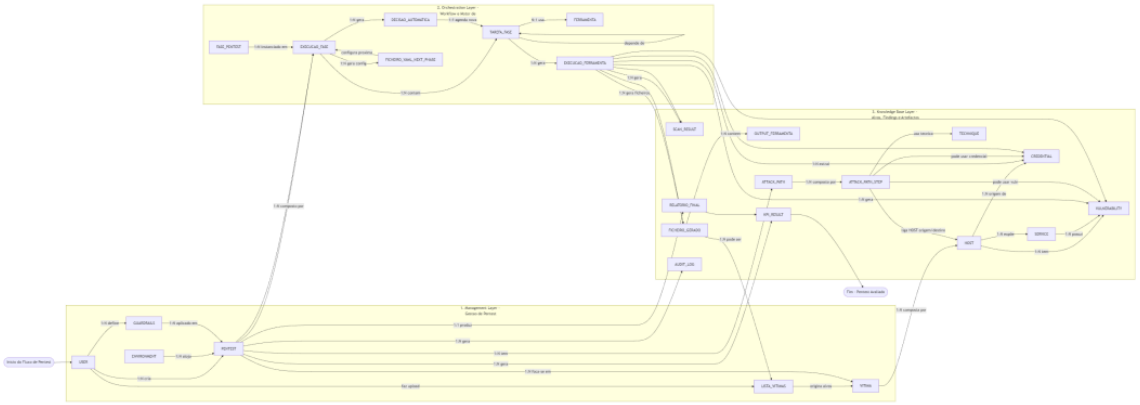

Figura 4.2.1 - Fluxo lógico de orquestração e recolha de dados do Venom APT.

O modelo de dados relacional que suporta este fluxo, incluindo as entidades mencionadas e as suas relações, encontra-se sintetizado na **Figura 4.2.2**

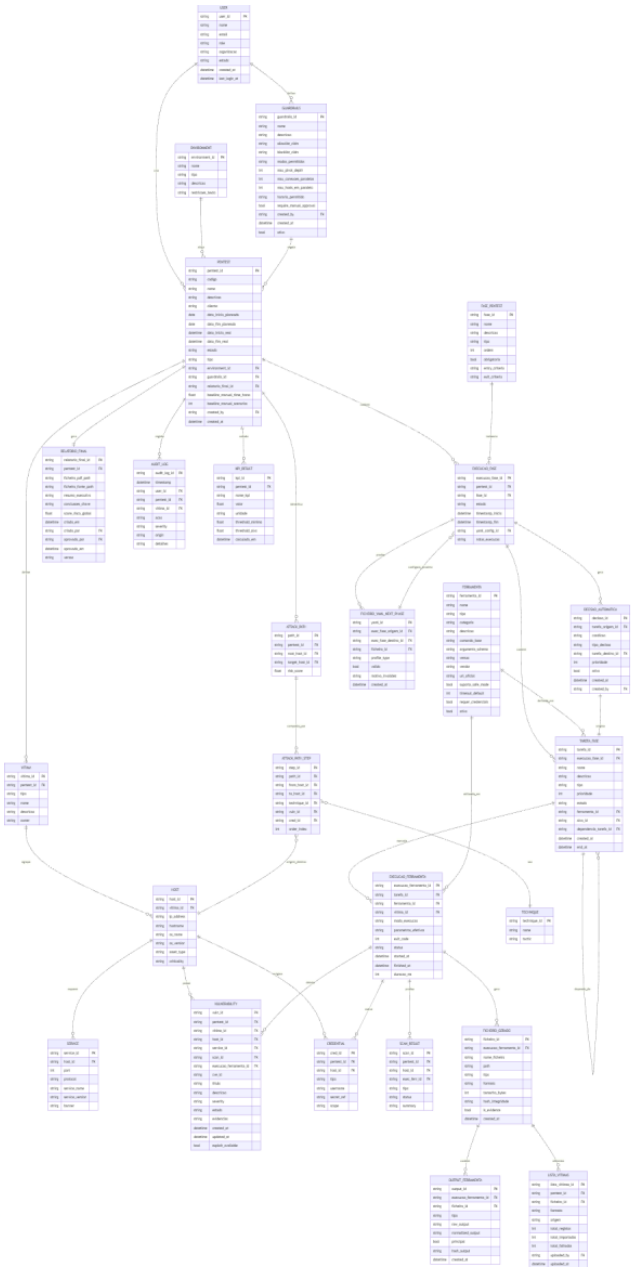

Figura 4.2.2 - Diagrama Entidade–Relacionamento (ER) do Venom APT.

Por fim, a **Figura 4.2.**3 apresenta um diagrama UML de classes que consolida a visão arquitetural ao nível do código, explicitando os principais componentes do núcleo do sistema (por exemplo, PentestManager, DecisionEngine, TaskScheduler, ToolExecutor, GuardrailsEnforcer e camada de acesso a dados) e as respetivas responsabilidades.

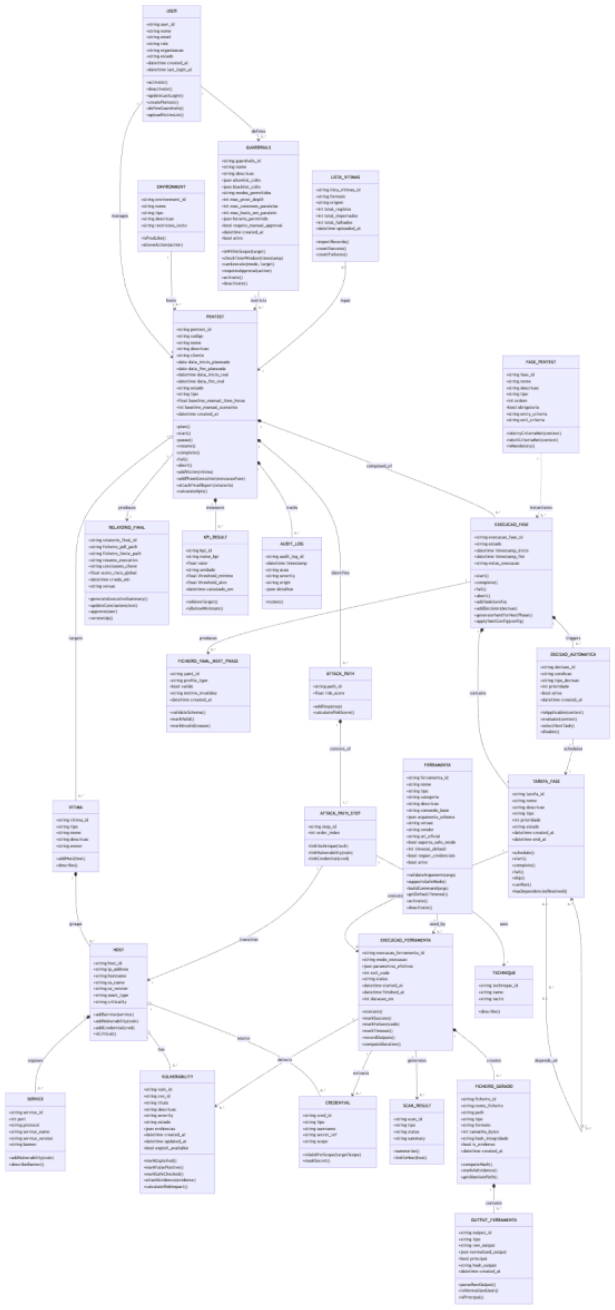

Figura 4.2.3 - Diagrama UML de classes do núcleo do Venom APT.

Em síntese, a solução assegura uma cadeia de rastreabilidade explícita:

- dos requisitos funcionais e não funcionais (Tabelas 4.2.1 e 4.2.2);
- para os Epics e Features que estruturam o backlog (Tabelas 4.2.3 e 4.2.4);
- para o backlog de User Stories com DoR/DoD e critérios de aceitação (Tabela 4.2.5);
- para a arquitetura lógica, modelo de dados e fluxo de orquestração (Figuras 4.1.1 a 4.2.3);

• culminando nos KPIs de avaliação (tempo, cobertura, reprodutibilidade e conformidade com guardrails), que serão medidos com base nos dados recolhidos e descritos neste capítulo.

Esta rastreabilidade garante que a solução proposta não é apenas tecnicamente consistente, mas também objetivamente alinhada com os objetivos de avaliação definidos para o Venom APT.

4.2.5 KPIs de avaliação e justificação dos valores-alvo

Para avaliar o sucesso do Venom APT foram definidos indicadores quantitativos que medem o desempenho da automação em comparação com um processo manual tradicional (baseline). Cada KPI explicita:

- a métrica e a fórmula de cálculo;
- a meta-alvo (objetivo de sucesso);
- o valor mínimo aceitável (limiar de viabilidade).

Os intervalos propostos (por exemplo, redução de 50–60 % do tempo e fator 2,5× na cobertura) são metas de engenharia definidas de forma conservadora para um protótipo académico em ambiente laboratorial: ambiciosas o suficiente para evidenciar ganhos claros face à abordagem manual, mas realistas face às limitações de tempo, maturidade do código e recursos disponíveis. A validação empírica destes valores será efetuada no capítulo de avaliação experimental, com base nos dados recolhidos segundo o modelo descrito na Secção 4.4–4.7.

4.2.5.1.1 Eficiência temporal (redução do tempo de ciclo)

Objetivo
Quantificar a redução do tempo total necessário para concluir um fluxo de pentesting completo (do reconhecimento à geração de relatório) face a uma abordagem manual equivalente.

Métrica
Percentagem de redução do tempo médio de execução.

Fórmula

$$\text{Redução (\%)} = \frac{T_{\text{manual}} - T_{\text{venom}}}{T_{\text{manual}}} \times 100$$

onde T_{manual} é o tempo médio de execução manual e T_{venom} o tempo médio com a ferramenta.

Metas de desempenho

- **Meta-alvo:** redução entre **50 % e 60 %**
  – o Venom APT conclui o fluxo completo em cerca de metade do tempo do teste manual.

- **Mínimo aceitável:** redução **≥ 30 %**
  – abaixo deste valor, o custo de desenvolvimento e operação da automação pode não justificar o benefício.

Exemplo prático
Num cenário em que o fluxo manual demora **20 horas úteis**, o Venom APT deve completar a execução em aproximadamente **8 a 10 horas** para cumprir a meta-alvo, e **≤ 14 horas** para cumprir o limiar mínimo.

4.2.5.1.2 Densidade de cobertura (cenários de ataque por janela temporal)

Objetivo
Demonstrar que a automação permite exercitar mais caminhos de ataque e TTPs (táticas, técnicas e procedimentos) na mesma janela temporal do que uma equipa humana.

Métrica
Fator de multiplicação de TTPs distintos por sessão.

Fórmula

$$\text{Fator de Cobertura} = \frac{\text{Nº TTPs Venom APT}}{\text{Nº TTPs Manuais}}$$

Metas de desempenho

- **Meta-alvo:** fator entre **2,5× e 3,0×**
  – por exemplo, executar **18 a 25 TTPs** por ciclo, comparado com **6 a 10 TTPs** em abordagem manual equivalente.
- **Mínimo aceitável:** fator **≥ 1,5×**
  – aumento de, pelo menos, **50 %** na cobertura de TTPs.

Justificação
A automação elimina tempo de consulta, hesitação e repetição manual entre comandos, permitindo explorar de forma sistemática caminhos laterais e variações de ataque que um operador humano tende a ignorar por falta de tempo ou fadiga operacional.

4.2.5.1.3 Índice de reprodutibilidade (consistência)

Objetivo
Garantir que o sistema gera resultados consistentes e auditáveis, reduzindo a variabilidade subjetiva inerente a diferentes operadores humanos.

Métrica
Taxa de sucesso em que, dado o mesmo cenário inicial (seed/perfil) e infraestrutura inalterada, a ferramenta reproduz a mesma sequência de passos e os resultados-chave.

Fórmula

$$\text{Reprodutibilidade (\%)} = \frac{\text{Execuções reproduzidas}}{\text{Total de execuções}} \times 100$$

Metas de desempenho

- **Meta-alvo: 95 %** de consistência em cenários estáveis.
- **Mínimo aceitável: ≥ 90 %**.

Justificação
A meta não é fixada em 100 % porque, mesmo em ambiente laboratorial, existem fatores não determinísticos (latência de rede, tempos de resposta inconsistentes dos serviços alvo, load da infraestrutura) que podem introduzir desvios pontuais, independentemente da lógica da ferramenta.

4.2.5.1.4 Conformidade de segurança (safety & guardrails)

Objetivo
Avaliar a capacidade do Venom APT de respeitar rigidamente as políticas de segurança e os guardrails definidos (escopo de IPs, janelas temporais, logging obrigatório e kill-switch).

Métrica
Percentagem de execuções que cumprem integralmente todos os guardrails, sem violações de perímetro ou quebra das políticas definidas.

Fórmula

$$\text{Conformidade (\%)} = \frac{\text{Execuções sem violação}}{\text{Total de execuções}} \times 100$$

Metas de desempenho

- **Meta-alvo:** entre **95 % e 99 %** de execuções em plena conformidade.
- **Mínimo aceitável: ≥ 95 %**.

Valores inferiores a 95 % indicam falha crítica no mecanismo de contenção do protótipo e obrigam à revisão da arquitetura de guardrails antes de qualquer evolução futura.

Nota
Embora se ambicione um comportamento ideal (100 %), a margem técnica admite falhas residuais relacionadas com problemas de logging ou timeouts de verificação, desde que **não resultem em tráfego fora do escopo autorizado** nem em ações ofensivas não previstas no perfil de ataque.

Indicador (KPI)	Métrica chave	Meta-alvo (sucesso)	Mínimo aceitável
1. Eficiência temporal	% de redução do tempo de ciclo	50 % – 60 %	≥ 30 %
2. Densidade de cobertura	Fator de TTPs distintos por sessão	2,5× (≈18–25 TTPs vs 6–10)	1,5×
3. Reprodutibilidade	% de execuções reproduzidas	95 %	≥ 90 %
4. Segurança (guardrails)	% de execuções em plena conformidade	95 % – 99 %	≥ 95 %

Tabela 4.2.6 – Síntese dos KPIs de avaliação do Venom APT

4.3 Estado Atual da Implementação e Pipeline Funcional

A presente secção descreve o estado atual da implementação do sistema Venom APT, com foco nos componentes efetivamente desenvolvidos e no pipeline funcional atualmente operacional. Nesta fase do projeto, observa-se a transição entre a definição conceptual da arquitetura e a sua concretização técnica, particularmente ao nível da fase de reconhecimento (reconnaissance).

Ao contrário da primeira entrega intercalar, onde o trabalho se encontrava predominantemente ao nível da definição de requisitos, arquitetura e planeamento, o sistema apresenta agora um núcleo funcional implementado, capaz de executar ferramentas externas, recolher evidência e transformar outputs em dados estruturados.

A implementação seguiu uma abordagem incremental e orientada por contrato (contract-driven), privilegiando a construção de uma base sólida, auditável e extensível, antes da evolução para camadas mais avançadas, como persistência do domínio e tomada de decisão autónoma.

4.3.1 Estado Atual da Implementação

À data da presente entrega, o sistema encontra-se funcional ao nível da execução de ferramentas de reconhecimento e do processamento dos respetivos outputs.

Foram implementados os seguintes componentes principais:

Em primeiro lugar, um sistema de **controlo de execução baseado em políticas (*guardrails*)**, responsável por validar o *scope* dos alvos e garantir conformidade com regras de segurança. Este sistema segue um modelo *deny-by-default*, suportando allowlists de domínios e CIDRs, controlo de portas, limites de concorrência, *rate limiting*, *timeouts* e mecanismos de segurança como *safe mode* e *kill switch*.

Em segundo lugar, foi desenvolvido um sistema de **definição declarativa de ferramentas**, baseado em *manifests* YAML, permitindo descrever ferramentas, argumentos, outputs esperados e parâmetros de execução de forma estruturada e extensível.

Em terceiro lugar, foi implementada uma camada de **execução de ferramentas**, responsável por construir comandos dinamicamente, invocar processos externos e monitorizar a sua execução. Esta camada assegura controlo de tempo, captura de outputs e gestão de erros.

Adicionalmente, foi implementado um modelo estruturado de **resultados de execução**, encapsulando informação como comando executado, duração, código de saída, outputs e ficheiros gerados.

Foi também desenvolvida uma **Evidence Layer**, responsável pela preservação estruturada de todos os artefactos gerados durante a execução.

Por fim, foi implementada uma **camada de normalização baseada em contrato**, que transforma outputs heterogéneos em estruturas consistentes e semanticamente ricas.

No entanto, subsistem ainda componentes não implementados ou incompletos:

- Persistence Layer do domínio (modelação e armazenamento estruturado de entidades);
- Decision Engine para orquestração autónoma;
- Execução completa da *full kill chain* de pentesting.

Deste modo, o sistema atual deve ser entendido como um **núcleo funcional centrado no reconnaissance**, já validado tecnicamente, mas ainda não completo face à visão global do projeto.

4.3.2 Pipeline Atual do Sistema

O pipeline implementado no sistema reflete uma abordagem estruturada, orientada a controlo, execução e transformação de dados, podendo ser representado da seguinte forma:

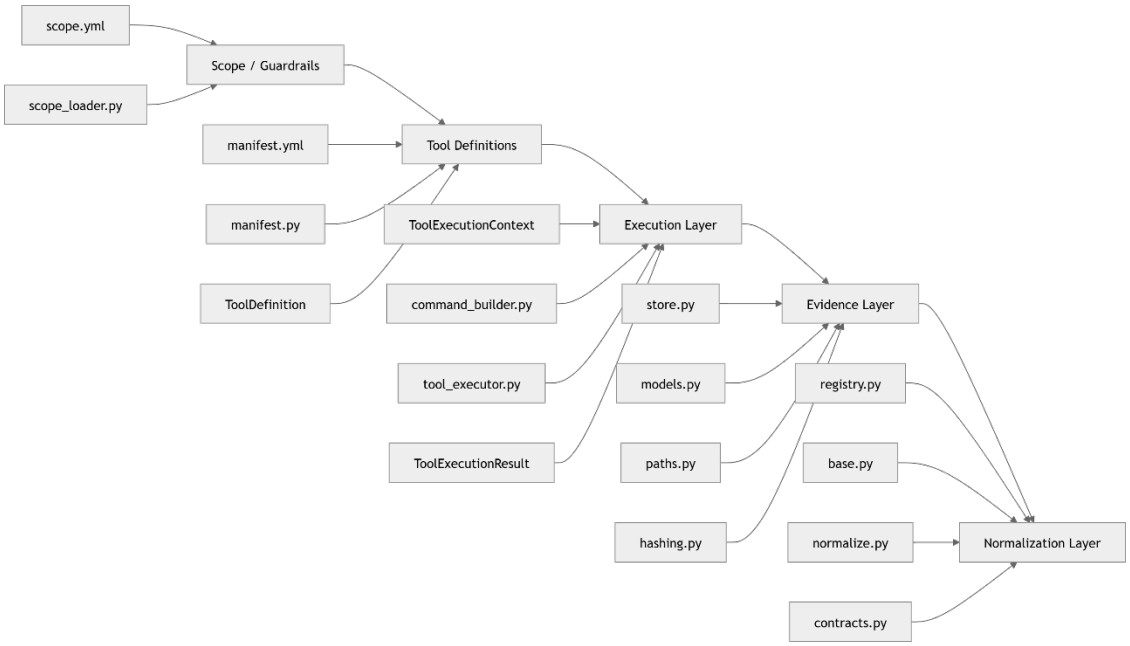

Figura 4.3.1 - Pipeline Sistema

Não foi encontrada nenhuma entrada do índice de ilustrações.O processo inicia-se com o carregamento e validação das políticas de **scope e guardrails**, que definem os limites operacionais do sistema.

Seguidamente, são carregadas as **definições de ferramentas**, permitindo configurar dinamicamente o comportamento do sistema.

Na fase de **execution**, os comandos são construídos dinamicamente e executados, sendo monitorizados todos os parâmetros relevantes.

Os outputs gerados são capturados e armazenados na **Evidence Layer**, garantindo preservação integral da execução.

Por fim, os dados são processados pela **camada de normalização**, que os transforma numa estrutura consistente baseada num contrato de dados previamente definido.

Este pipeline assegura separação clara entre execução, evidência e abstração de dados, constituindo a base para evolução futura do sistema.

Para além da organização sequencial das fases, este pipeline foi concebido com uma arquitetura modular. Cada componente — validação de políticas (scope e guardrails), execução de ferramentas, recolha de evidência e normalização — está desacoplado dos restantes, comunicando através de interfaces e estruturas de dados bem definidas.

Esta separação permite que cada módulo evolua de forma independente, sem impacto direto nas restantes componentes do sistema. Por exemplo, novas ferramentas podem ser integradas sem alterações na camada de decisão, e diferentes mecanismos de normalização podem ser introduzidos sem modificar o processo de execução.

Adicionalmente, a utilização de contratos de dados entre camadas garante consistência na comunicação e facilita a substituição ou extensão de componentes, suportando a evolução progressiva do sistema.

Desta forma, o pipeline não representa apenas uma sequência de operações, mas sim uma arquitetura modular orientada à extensibilidade, reutilização e manutenção a longo prazo. Esta abordagem segue princípios de engenharia de software como separação de responsabilidades e baixo acoplamento, fundamentais para sistemas escaláveis e evolutivos.

4.3.3 Implementação Técnica

A implementação técnica do sistema assenta numa arquitetura modular, baseada em abstrações claras entre definição de ferramentas, contexto de execução e resultados operacionais.

A estrutura **ToolDefinition** representa cada ferramenta, incluindo o binário, argumentos e outputs esperados.

O **ToolExecutionContext** encapsula o contexto dinâmico de execução, permitindo resolver variáveis como o alvo e diretórios de output.

A construção dos comandos é realizada de forma dinâmica, garantindo flexibilidade e reutilização.

A execução é realizada pelo **ToolExecutor**, que invoca ferramentas externas, controla *timeouts* e captura outputs de forma completa.

Os resultados são encapsulados em estruturas **ToolExecutionResult**, uniformizando o tratamento dos outputs.

Adicionalmente, o sistema implementa um mecanismo de **registo dinâmico de normalizadores**, permitindo associar automaticamente cada ferramenta ao respetivo módulo de normalização, facilitando a extensibilidade do sistema.

4.3.4 Camada de Normalização

A camada de normalização assume um papel central na arquitetura do sistema, indo além de um simples mecanismo de parsing.

Dada a heterogeneidade dos outputs das ferramentas, foi definido um **contrato de dados normalizado**, que permite representar de forma consistente entidades como:

- Execuções de ferramentas
- Hosts
- Serviços
- URLs
- Registos DNS

A normalização segue um processo estruturado:

1. Receção do resultado de execução
2. Leitura da evidência
3. Extração e transformação de dados
4. Construção de um *bundle* normalizado

Este *bundle* agrega todas as entidades produzidas durante a execução, garantindo consistência e alinhamento com o contrato definido.

Adicionalmente, o sistema inclui mecanismos de **validação estrutural dos dados normalizados**, assegurando que todos os outputs cumprem o formato esperado, reduzindo inconsistências e erros.

Deste modo, a normalização estabelece a base para a futura persistência e tomada de decisão autónoma.

4.3.5 Evidence Layer

Evidence Layer constitui uma componente crítica do sistema, responsável pela preservação integral dos artefactos gerados.

Cada execução gera um **EvidenceBundle**, agregando todos os ficheiros e metadados associados.

Os ficheiros são representados através de estruturas tipadas que incluem:

- nome
- tipo
- localização
- tamanho
- hash SHA256

A organização dos dados segue uma estrutura hierárquica baseada em diretórios, separando:

- dados brutos
- logs
- metadados

Durante a persistência são armazenados:

- stdout e stderr
- ficheiros gerados
- metadados de execução
- hashes de integridade

Esta abordagem garante:

- auditabilidade

- reprodutibilidade
- rastreabilidade completa

A Evidence Layer estabelece ainda uma ligação direta com a camada de normalização, permitindo transformar evidência em dados estruturados.

4.3.6 Limitações Atuais

O sistema apresenta ainda algumas limitações relevantes:

- ausência de uma Persistence Layer completa
- inexistência de Decision Engine autónomo
- pipeline ainda parcialmente manual
- ausência de execução completa da *full kill chain*
- dependência de ferramentas externas

Estas limitações definem o estado atual do sistema e orientam o desenvolvimento futuro.

4.3.7 Próximos Passos

Os próximos desenvolvimentos incluem:

- implementação da Persistence Layer
- desenvolvimento do Decision Engine
- automatização do pipeline completo
- expansão para fases adicionais do pentest

4.3.8 Validação Funcional do Pipeline

Para validar o funcionamento do sistema, foi desenvolvido um conjunto de testes *end-to-end* que exercitam o pipeline completo.

Estes testes incluem:

- execução de ferramentas reais em ambiente controlado
- validação de acessibilidade dos alvos
- verificação da geração de evidência
- validação dos outputs normalizados

O processo de validação percorre todas as etapas:

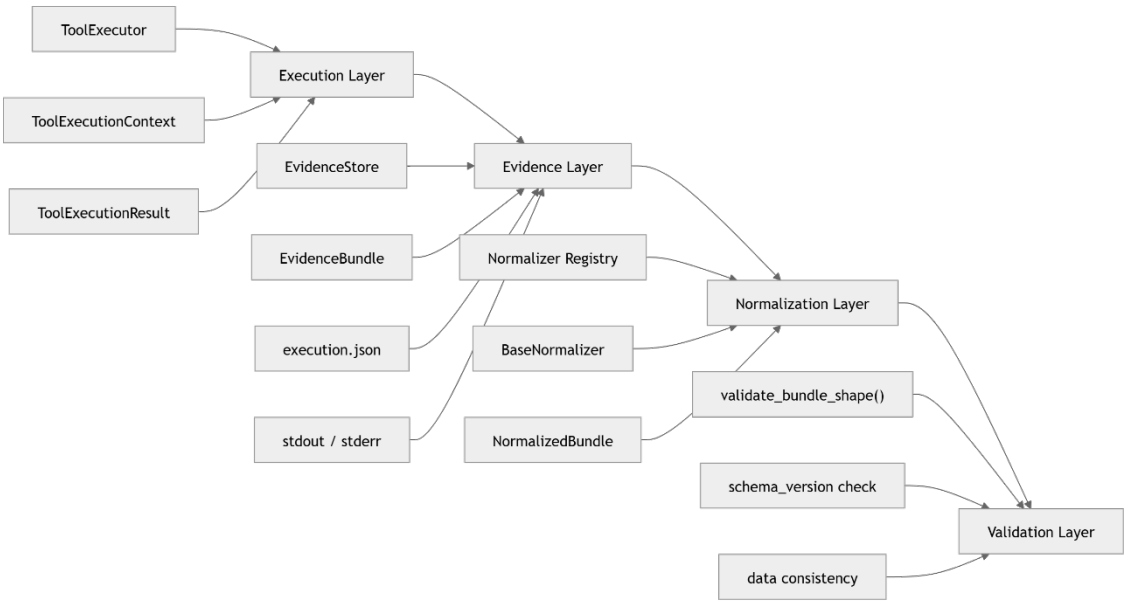

Figura 4.3.2 - Processo De Validação

São verificadas condições como:

- existência de ficheiros de execução
- geração de artefactos esperados
- consistência dos dados normalizados

Esta abordagem permite demonstrar o funcionamento do sistema em condições reais, validando a sua robustez e operacionalidade.

4.4 Tecnologias e Ferramentas Utilizadas

A solução foi concebida com base numa stack tecnológica realista e alinhada com práticas modernas de engenharia de software aplicadas à cibersegurança ofensiva, garantindo simultaneamente o cumprimento dos requisitos funcionais e não funcionais definidos.

a) Linguagem de programação e ambiente de execução

O núcleo do Venom APT é desenvolvido em Python, tirando partido da sua flexibilidade e ecossistema para orquestração de ferramentas de segurança, manipulação de processos e tratamento de dados. A escolha desta linguagem permite uma integração eficiente com utilitários externos, bem como a implementação de lógica de controlo e decisão.

O código encontra-se estruturado de forma modular, refletindo as diferentes componentes do sistema (execução, evidência, normalização e decisão), facilitando a manutenção, extensibilidade e evolução incremental do projeto.

b) Integração de ferramentas ofensivas

O sistema não implementa diretamente todas as capacidades de ataque, mas integra ferramentas especializadas amplamente utilizadas na prática de testes de intrusão. Estas ferramentas são executadas através de um componente dedicado (Tool Executor), responsável por construir comandos, gerir a execução em ambiente isolado e capturar os resultados.

No contexto da fase de reconhecimento, são utilizadas ferramentas como:

• **dig** e **host**, para recolha de informação DNS e identificação de registos associados a domínios;

• **nmap**, para identificação de portas abertas e serviços disponíveis;

• **httpx**, para validação de serviços web ativos e recolha de metadados HTTP;

• **katana**, para descoberta de URLs e mapeamento de superfície de ataque web.

A integração destas ferramentas é definida através de configurações estruturadas (manifestos), onde são especificados parâmetros de execução, formatos de saída esperados e regras de processamento. Sempre que necessário, são utilizados mecanismos auxiliares para uniformizar a execução e facilitar a posterior normalização dos resultados.

Desta forma, o Venom APT atua como uma camada de orquestração inteligente, coordenando a execução de ferramentas e integrando os seus resultados no fluxo de decisão do sistema. A seleção destas ferramentas baseia-se na sua robustez, ampla adoção na comunidade e capacidade de integração em pipelines automatizados.

c) Guardrails, governação e segurança operacional

A camada de controlo operacional (guardrails) assegura que todas as ações executadas respeitam limites definidos, sendo implementada através de:

• Definição de âmbito (scope), incluindo listas de alvos autorizados;

• Validação de destinos antes da execução de qualquer ação de rede;

• Mecanismo de interrupção global (kill-switch), capaz de terminar execuções em curso;

• Restrições de execução que garantem a utilização de técnicas controladas e não destrutivas.

Estas capacidades garantem conformidade com requisitos de segurança operacional e asseguram que o sistema pode ser utilizado em contexto ético e controlado.

d) Persistência de dados e gestão de evidência

Os resultados das execuções são geridos através de uma camada dedicada de evidência (Evidence Layer), responsável por recolher, armazenar e organizar todos os artefactos gerados durante o processo.

Para cada execução, são registados comandos executados, resultados obtidos, ficheiros gerados e metadados associados. Estes dados são armazenados de forma estruturada, permitindo rastreabilidade completa e reprodutibilidade das execuções.

Adicionalmente, uma camada de normalização transforma os resultados em formatos estruturados, facilitando a sua análise, correlação e integração com outras componentes do sistema.

e) Empacotamento e execução

O sistema é executado em ambientes isolados baseados em containers (por exemplo, Docker), garantindo consistência entre execuções e independência relativamente ao ambiente subjacente.

Esta abordagem permite reproduzir cenários de teste de forma controlada, assegurando previsibilidade, isolamento e facilidade de deploy em contexto laboratorial.

4.5 Recolha dos Dados

A solução foi desenhada para que todo o ciclo de ataque produza dados estruturados suficientes para avaliação quantitativa dos KPIs e para auditoria detalhada.

Durante cada sessão PENTEST, o sistema recolhe:

- **Eventos operacionais** em AUDIT_LOG, incluindo timestamp, utilizador (quando aplicável), tipo de ação (PENTEST_STARTED, TASK_EXECUTED, GUARDRAIL_BLOCKED, KILL_SWITCH, etc.), severidade e origem (ENGINE, UI, API, SYSTEM).
- **Execuções de ferramentas** em EXECUCAO_FERRAMENTA, com registo da tarefa associada (TAREFA_FASE), alvo lógico (VITIMA/HOST), modo de execução (SAFE, FULL, SIMULATION), parâmetros efetivos, códigos de saída, estado (SUCCESS, FAIL, PARTIAL, TIMEOUT) e duração.
- **Outputs de ferramentas e artefactos** em FICHEIRO_GERADO e OUTPUT_FERRAMENTA, armazenando logs, pcaps, screenshots e resultados normalizados em JSON.
- **Findings de segurança** em VULNERABILITY, CREDENTIAL, ATTACK_PATH e ATTACK_PATH_STEP, com mapeamento à matriz MITRE ATT&CK através de TECHNIQUE.

Em paralelo, os dados necessários para cálculo dos KPIs (tempo, cobertura, reprodutibilidade, conformidade) são consolidados em KPI_RESULT, ao nível de cada PENTEST.

A recolha é automática e integrada na operação normal da ferramenta, evitando instrumentação manual adicional e garantindo uma trilha completa desde cada comando enviado até à evidência registada.

4.6 Descrição dos Dados

Os dados recolhidos podem ser descritos em três níveis principais, alinhados com o modelo de dados do Venom APT.

a) Nível de sessão (PENTEST / KPI_RESULT / RELATORIO_FINAL)

Cada registo em PENTEST representa uma sessão completa de pentest num ambiente e âmbito definidos, incluindo: tipo de teste (BLACKBOX, GREYBOX, WHITEBOX), estado (PLANEADO, EM_EXECUCAO, CONCLUIDO, etc.), timestamps planeados e reais, ambiente (ENVIRONMENT), políticas (GUARDRAILS), ligação ao RELATORIO_FINAL e aos KPIs (KPI_RESULT). Estes dados permitem calcular, por sessão:

- a duração total (para o KPI de eficiência temporal);
- o número de TTPs exercitados (via mapeamento MITRE em ATTACK_PATH_STEP);
- o nível de conformidade com guardrails (via AUDIT_LOG).

b) Nível operacional (fases, tarefas, ferramentas e logs)

Ao nível operacional, o modelo regista o fluxo interno do motor:

- EXECUCAO_FASE e TAREFA_FASE representam a decomposição da sessão em fases e tarefas concretas (scans, enumerações, exploits, reporting);
- DECISAO_AUTOMATICA regista as regras de decisão aplicadas (“Se–Então”), assegurando explicabilidade das escolhas do motor;
- EXECUCAO_FERRAMENTA detalha cada invocação de módulo, incluindo alvo, parâmetros, modo de execução, estado e tempos;
- AUDIT_LOG e SCAN_RESULT consolidam o rasto de eventos e os resultados agregados de scans.

Este nível suporta análises de desempenho interno, como identificação de tarefas mais demoradas, ferramentas com maior taxa de falha ou padrões de decisão recorrentes.

c) Nível de findings e conhecimento (alvos, vulnerabilidades, caminhos de ataque)

Por fim, o nível de conhecimento inclui:

- VITIMA, HOST e SERVICE, que estruturam a superfície de ataque (alvos lógicos, máquinas, serviços);
- VULNERABILITY e CREDENTIAL, que registam vulnerabilidades encontradas, o seu contexto técnico e as credenciais/segredos extraídos;
- ATTACK_PATH e ATTACK_PATH_STEP, que codificam cadeias de compromisso entre um “patient zero” e um alvo final, ligando hosts, técnicas MITRE, vulnerabilidades e credenciais usadas;
- RELATORIO_FINAL, que consolida resultados num documento, incluindo resumo executivo, conclusões e score global de risco.

Este nível alimenta tanto a análise quantitativa (contagem de vulnerabilidades, severidades, comprimento de caminhos de ataque) como a análise qualitativa (narrativa de ataque, impacto potencial, recomendações).

4.7 Pré-processamento dos Dados

Antes da análise quantitativa e comparação com a baseline manual, os dados recolhidos serão sujeitos a um conjunto de operações de pré-processamento, com dois objetivos principais: assegurar qualidade e integridade dos dados, e garantir comparabilidade entre execuções.

As etapas previstas incluem, entre outras:

- **Filtragem de sessões:** identificação e eventual exclusão ou marcação de PENTEST com estado FALHOU ou ABORTADO que não cumpram o fluxo mínimo necessário para cálculo de todos os KPIs, mantendo-os para análise de robustez do sistema.
- **Normalização temporal:** cálculo de durações a partir de started_at/finished_at (execuções de ferramentas) e data_inicio_real/data_fim_real (sessões), convertendo todos os valores para unidades homogéneas (segundos/horas).
- **Deduplicação de findings:** consolidação de vulnerabilidades duplicadas para o mesmo host/serviço (por exemplo, detetadas por múltiplas ferramentas) de forma a que cada VULNERABILITY represente um finding único na análise.
- **Tratamento de timeouts e falhas parciais:** classificação explícita de execuções como SUCCESS, FAIL, PARTIAL ou TIMEOUT em EXECUCAO_FERRAMENTA e SCAN_RESULT, para evitar enviesamentos nas métricas de desempenho.
- **Normalização MITRE ATT&CK:** verificação de que cada ATTACK_PATH_STEP possui técnica associada (TECHNIQUE); passos sem mapeamento podem ser marcados como "não classificados" e tratados separadamente nas análises de cobertura.

O resultado deste pré-processamento será um conjunto de tabelas limpas e integradas, aptas para a construção de métricas comparáveis entre o Venom APT e a abordagem manual.

4.8 Análise Exploratória dos Dados

A Análise Exploratória de Dados (AED) incidirá sobre o conjunto de execuções recolhidas (manual vs automatizada), com o objetivo de avaliar preliminarmente os KPIs definidos e identificar padrões relevantes de comportamento do sistema.

De forma resumida, estão previstas as seguintes linhas de análise:

- **Eficiência temporal:**
    - distribuição da duração das sessões PENTEST (histogramas, médias, medianas, desvios padrão);
    - breakdown de tempos por fase (EXECUCAO_FASE) e por tipo de tarefa (TAREFA_FASE), identificando gargalos e ganhos específicos.

- **Cobertura de TTPs e superfície de ataque:**
    - contagem de técnicas MITRE distintas exercitadas por sessão (via ATTACK_PATH_STEP e TECHNIQUE);
    - cálculo do fator de cobertura (TTPs Venom / TTPs manuais), em linha com o KPI de densidade de cobertura.
- **Perfil de vulnerabilidades e caminhos de ataque:**
    - análise da distribuição de severidades em VULNERABILITY (LOW, MEDIUM, HIGH, CRITICAL);
    - estudo da estrutura dos caminhos de compromisso (ATTACK_PATH), incluindo comprimento médio e tipos de hosts envolvidos em movimentos laterais.
- **Reprodutibilidade e estabilidade:**
    - avaliação da consistência entre execuções repetidas com a mesma configuração (seed/perfil) e ambiente inalterado, medindo o KPI de reprodutibilidade;
    - identificação de fontes de variabilidade (latência, timeouts, serviços intermitentes).
- **Conformidade com guardrails:**
    - cálculo da percentagem de execuções sem qualquer violação registada em AUDIT_LOG;
    - análise de eventos GUARDRAIL_BLOCKED como indicador da eficácia das barreiras de segurança.

Os resultados detalhados desta análise, bem como a comparação quantitativa com a baseline manual e a discussão crítica face aos objetivos do projeto, serão apresentados no capítulo dedicado à avaliação experimental e conclusões.

5 Calendário, método e planeamento

Esta secção apresenta o plano de trabalho para o TFC, estruturado em fases com inspiração em boas práticas de gestão de projeto: definição de marco mínimo funcional, iterações curtas com feedback do orientador e controlo explícito de riscos (âmbito, tempo e complexidade técnica).
O foco é o período até à 2.ª entrega intercalar, complementado com uma visão de alto nível até à entrega final.

5.1 Plano detalhado até à 2.ª entrega intercalar (Dez. 2025 – Mar. 2026)

O objetivo desta fase é atingir um **protótipo mínimo funcional (MVP)** do Venom APT, capaz de:

- executar um ciclo de pentesting automatizado end-to-end em ambiente de laboratório limitado;
- aplicar guardrails básicos (allowlist, kill-switch, integridade);
- registar dados suficientes para cálculo preliminar de KPIs.

Macro-fases e tarefas (detalhe):

- **Fase A – Consolidação de engenharia e preparação do ambiente (Dez. 2025)**
    - **T1 – Revisão da visão, problema, RF/RNF e KPIs**
      Refinar os requisitos funcionais e não funcionais e validar, com o orientador, o âmbito técnico final da primeira versão do protótipo.
    - **T2 – Fecho da arquitetura lógica e modelo de dados**
      Avaliação da versão inicial do modelo relacional (PENTEST, EXECUCAO_FERRAMENTA, AUDIT_LOG, KPI_RESULT, etc.) e da arquitetura em três camadas (Gestão de Pentest, Orquestração, Knowledge Base) para implementação.
    - **T3 – Preparação do ambiente de laboratório**
      Definir e montar o ambiente de teste (máquina de controlo, VMs/alvos, rede isolada), incluindo cenário para baseline manual.
- **Fase B – Implementação do núcleo do protótipo (Jan. – Fev. 2026)**
    - **T4 – Implementação da camada de Gestão de Pentest (EP04/EP05)**
      Modelos e serviços para USER, ENVIRONMENT, GUARDRAILS, PENTEST; criação/edição de sessões de teste; configuração de âmbito e janelas temporais.
    - **T5 – Implementação do módulo de Discovery e Enumeração (EP01)**
      Módulos para varrimento de rede, enumeração de portas/serviços e construção do inventário (hosts, serviços, banners), alinhados com RF01, RF02 e as user stories US-EP01-01/02.

  - **T6 – Implementação inicial do Motor de Decisão e Knowledge Base (EP02)**
    Lógica Se–Então básica, estrutura de Knowledge Base em memória, registo de dependências entre módulos, prevenção de loops e duplicações (RF03, RF07).
  - **T7 – Implementação dos guardrails essenciais (EP04)**
    Wrapper de sockets com allowlist de IPs, mecanismo de auto-proteção, kill-switch global e verificação de integridade no arranque (RF10, RNF01–RNF03).

- **Fase C – Primeira integração, logging e preparação de avaliação (Fev. – Mar. 2026)**
  - **T8 – Integração end-to-end e logging estruturado**
    Encadear Discovery → Enumeração → Execução de um conjunto limitado de exploits de laboratório, garantindo registo em EXECUCAO_FERRAMENTA, ATTACK_PATH, AUDIT_LOG e mapeamento básico a MITRE ATT&CK.
  - **T9 – Definição da baseline manual e plano de testes preliminar**
    Documentar o fluxo de pentesting manual que servirá de comparação (passos, ferramentas, tempos) e alinhar com o orientador o conjunto de execuções necessárias para a fase de validação.
  - **T10 – Desenho do questionário de viabilidade e pertinência**
    Elaborar a versão inicial do questionário dirigido a pentesters, Blue Team/SOC e decisores, em linha com os objetivos de validação externa descritos no Capítulo 3.
  - **T11 – Revisão intermédia com o orientador e ajustes ao plano**
    Rever resultados da integração, identificar riscos (atrasos, complexidade excessiva de módulos, constrangimentos do laboratório) e, se necessário, re-priorizar funcionalidades para garantir um MVP estável para a 2.ª entrega intercalar.

Marco da 2.ª entrega intercalar (M2)
Na data da 2.ª entrega intercalar, prevê-se:

- protótipo funcional com ciclo automatizado completo num cenário de laboratório reduzido;
- guardrails básicos implementados e testados;
- logging estruturado e modelo de dados operacional;
- baseline manual definida e documentada;
- questionário de viabilidade desenhado (ainda não necessariamente aplicado).

5.2 Visão de alto nível até à entrega final (Abr. – Jun. 2026)

Após a 2.ª entrega intercalar, o foco passa da construção do núcleo da solução para:

- **alargamento controlado de funcionalidades**;
- **execução sistemática de testes e recolha de dados**;

- **análise quantitativa dos KPIs e redação final do relatório**.

- **Fase D – Expansão funcional e estabilização (Abr. 2026)**
    - Alargar o arsenal de ataque (EP03) com novos módulos de exploração e movimento lateral, mantendo o foco em vulnerabilidades clássicas de laboratório e em payloads não destrutivos.
    - Refinar a visualização de Attack Path, reporting técnico e exportações MITRE Navigator (EP05).
    - Completar a instrumentação de KPIs (tempo, cobertura, reprodutibilidade, conformidade).
- **Fase E – Avaliação experimental e análise de resultados (Abr. – Mai. 2026)**
    - Executar campanhas de testes manuais vs Venom APT sobre o ambiente alvo, segundo o plano definido.
    - Aplicar o questionário de viabilidade e pertinência e compilar as respostas.
    - Realizar o pré-processamento dos dados (limpeza, normalização, deduplicação) e a análise exploratória descrita no Capítulo 4, produzindo resultados quantitativos para cada KPI.
- **Fase F – Consolidação da solução e relatório final (Mai. – Jun. 2026)**
    - Incorporar correções e melhorias identificadas na fase de testes (robustez, desempenho, usabilidade).
    - Finalizar a documentação técnica (manual técnico, instruções de deploy em ambiente produtivo/laboratorial) e o repositório Git.
    - Redigir e rever o relatório final, garantindo coerência entre problema, solução, metodologia, resultados e trabalhos futuros, de acordo com o regulamento.

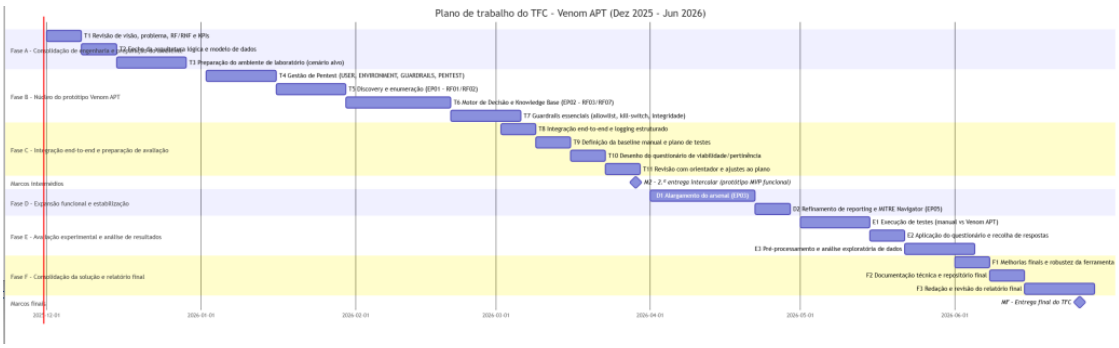

Figura 5.2.1 - Cronograma em formato Gantt

5.3 Progresso realizado, dificuldades e ajustes ao plano inicial

5.3.1 Progresso Realizado

Desde a primeira entrega intercalar, o projeto evoluiu significativamente, passando de uma fase predominantemente conceptual para a implementação efetiva de um núcleo funcional do sistema.

Na fase inicial, o trabalho encontrava-se centrado na definição do problema, levantamento de requisitos, análise do estado da arte e desenho da arquitetura global da solução. Nesta fase, foram estabelecidas as bases conceptuais do sistema Venom APT, incluindo o modelo de dados, a integração com o framework MITRE ATT&CK e a definição das principais camadas arquiteturais.

Na presente fase, verificou-se uma transição para a implementação técnica, tendo sido desenvolvido um conjunto consistente de componentes que suportam a fase de reconhecimento (*reconnaissance*).

Em particular, foram alcançados os seguintes progressos:

- Implementação de um sistema de **controlo de execução baseado em políticas (*guardrails*)**, garantindo que todas as operações respeitam um *scope* definido e operam dentro de limites seguros;
- Definição de um modelo **declarativo de ferramentas**, baseado em *manifests*, permitindo configurar e integrar múltiplas ferramentas de forma estruturada;
- Desenvolvimento de uma camada de **execução de ferramentas**, responsável por invocar processos externos, controlar a sua execução e capturar resultados;
- Implementação de um modelo estruturado de **resultados de execução**, encapsulando informação relevante para análise posterior;
- Construção de uma **Evidence Layer**, responsável pela preservação integral dos artefactos gerados, incluindo outputs, ficheiros e metadados;
- Desenvolvimento de uma **camada de normalização baseada em contrato**, permitindo transformar outputs heterogéneos em estruturas de dados consistentes;
- Integração de um sistema de **validação de dados normalizados**, assegurando conformidade com o schema definido;
- Implementação de **testes funcionais end-to-end**, que validam o pipeline completo, desde execução até à produção de dados normalizados.

Estes progressos demonstram que o sistema já possui um pipeline funcional completo ao nível da fase de reconhecimento, incluindo execução, recolha de evidência, transformação de dados e validação

5.3.2 Dificuldades Encontradas

Durante o desenvolvimento, foram identificados vários desafios técnicos que influenciaram a evolução do projeto.

Uma das principais dificuldades esteve relacionada com a heterogeneidade dos outputs das ferramentas de pentesting. Cada ferramenta produz dados em formatos distintos, o que exigiu o desenvolvimento de uma camada de normalização robusta e extensível.

Outra dificuldade relevante foi a necessidade de garantir auditabilidade e reprodutibilidade, o que levou à conceção de uma Evidence Layer estruturada, com suporte a metadados detalhados e mecanismos de verificação de integridade.

Adicionalmente, a definição de um sistema de execução controlado implicou a implementação de mecanismos de guardrails, assegurando que o sistema opera dentro de limites seguros e previsíveis.

Por fim, a integração de múltiplas ferramentas externas introduziu desafios ao nível de gestão de dependências, consistência de execução e validação de outputs, exigindo a criação de mecanismos adicionais de validação e teste.

5.3.3 Ajustes ao Plano Inicial

Face aos desafios identificados e à complexidade inerente ao desenvolvimento de uma ferramenta autónoma de pentesting, foi necessário proceder a ajustes ao plano inicial.

Inicialmente, o objetivo do projeto previa a implementação de uma solução completa, capaz de cobrir toda a *kill chain* de um processo de pentesting. No entanto, verificou-se que esta abordagem não seria viável dentro do tempo disponível, sem comprometer a qualidade técnica do sistema.

Deste modo, optou-se por uma estratégia de desenvolvimento incremental, centrada na construção de um **núcleo sólido e funcional na fase de reconnaissance**, que serve de base para futuras extensões.

Esta decisão permitiu:

- garantir a implementação de um pipeline completo e validado;
- desenvolver uma arquitetura consistente e extensível;
- assegurar qualidade e robustez na execução e tratamento de dados;

Como consequência desta reorientação, foram adiados para fases posteriores:

- a implementação da Persistence Layer do domínio;
- o desenvolvimento do Decision Engine completo;
- a integração de fases adicionais do pentest (exploração, movimento lateral, etc.).

Esta abordagem reflete uma opção consciente de privilegiar **qualidade sobre quantidade**, garantindo que o sistema desenvolvido apresenta coerência, robustez e valor técnico real.

5.3.4 Síntese

Em síntese, o projeto evoluiu de uma fase conceptual para a implementação de um sistema funcional ao nível da fase de reconhecimento, com um pipeline completo que integra execução, evidência, normalização e validação.

Apesar de ainda não representar a totalidade da visão inicial, o estado atual do sistema demonstra viabilidade técnica, coerência arquitetural e potencial de evolução, constituindo uma base sólida para o desenvolvimento das fases seguintes do projeto.

6 Plano de Testes e Validação

6.1 Objetivo dos Testes

O objetivo dos testes desenvolvidos no âmbito do projeto Venom APT consiste em validar o funcionamento do sistema ao nível do pipeline de reconhecimento, assegurando que as diferentes camadas da arquitetura — execução, evidência, normalização e validação — operam de forma integrada e consistente.

Em particular, pretende-se verificar:

- a correta execução de ferramentas externas de pentesting;
- a captura integral dos outputs gerados;
- a persistência estruturada da evidência;
- a transformação dos dados em estruturas normalizadas;
- a consistência e validade dos dados produzidos.

Os testes procuram, assim, demonstrar que o sistema é capaz de executar um fluxo funcional completo, desde a invocação de ferramentas até à produção de dados estruturados, suportando as fases subsequentes do projeto.

6.2 Ambiente de Teste

Os resultados dos testes são automaticamente recolhidos e persistidos pelo sistema Venom APT através de uma camada dedicada de gestão de evidência (Evidence Layer). Cada execução de ferramenta gera um conjunto de artefactos que inclui: comandos executados, saídas produzidas (stdout/stderr), ficheiros gerados pelas ferramentas e metadados associados, como identificador da execução (run_id), timestamps, alvo e contexto operacional.

Estes artefactos são armazenados de forma estruturada no diretório de dados do sistema (data/), organizados por execução e por fase do teste, permitindo uma separação clara entre diferentes sessões e garantindo rastreabilidade completa. Para além do armazenamento em bruto, os resultados são processados por uma camada de normalização que converte os outputs das ferramentas em formatos estruturados (nomeadamente JSON), facilitando a sua análise, correlação e reutilização.

Adicionalmente, esta informação é integrada na camada de persistência do domínio, onde os dados são modelados como entidades do sistema (por exemplo, execuções, resultados, hosts e serviços), permitindo associar cada artefacto ao seu contexto lógico dentro do fluxo de ataque.

Esta abordagem garante não só a preservação integral da evidência produzida durante os testes, mas também a capacidade de reproduzir execuções, auditar decisões tomadas pelo sistema e realizar análise comparativa entre diferentes ciclos de execução, constituindo um dos pilares fundamentais da reprodutibilidade e rastreabilidade no Venom APT.

6.3 Casos de Teste

Foram definidos vários casos de teste com o objetivo de validar diferentes componentes do sistema.

6.3.1 Testes de Execução de Ferramentas

Estes testes verificam a capacidade do sistema para invocar ferramentas externas e obter resultados válidos.

Incluem:

- execução de ferramentas de DNS (dig, host);
- execução de ferramentas de scanning (nmap);
- execução de ferramentas web (httpx, katana).

Os testes validam:

- código de saída das ferramentas;
- presença de outputs esperados;
- ausência de erros críticos durante execução.

6.3.2 Testes de Ambiente

Estes testes asseguram que os alvos e o ambiente estão corretamente configurados antes da execução.

Incluem:

- verificação de acessibilidade HTTP do alvo local;
- validação de resolução DNS;
- verificação de disponibilidade das ferramentas via Docker.

Estes testes garantem que falhas não são causadas por problemas externos ao sistema.

6.3.3 Testes da Evidence Layer

Estes testes validam a correta geração e armazenamento de artefactos.

São verificadas condições como:

- existência de ficheiros stdout.txt e stderr.txt;
- criação de ficheiros de output específicos (ex.: nmap.xml, httpx.ndjson);
- geração de ficheiros de metadados (execution.json);
- organização correta dos diretórios de execução.

Adicionalmente, são verificados mecanismos de integridade, como geração de hashes para os ficheiros produzidos.

6.3.4 Testes de Normalização

Estes testes validam a transformação dos outputs brutos em dados estruturados.

São verificadas condições como:

- geração de entidades esperadas (hosts, services, web_urls, dns_records);
- consistência dos dados produzidos;
- eliminação de duplicados;
- associação correta entre execuções e dados gerados.

6.3.5 Testes de Validação de Dados

Para garantir a consistência dos dados normalizados, foram implementados mecanismos de validação estrutural.

Estes testes verificam:

- presença de todas as secções obrigatórias no bundle normalizado;
- conformidade dos tipos de dados (listas e dicionários);
- consistência global da estrutura.

Esta validação assegura que os dados produzidos são utilizáveis por outras camadas do sistema.

6.4 Resultados Preliminares

Os testes realizados demonstraram que o sistema é capaz de executar corretamente o pipeline de reconhecimento.

Em particular, verificou-se que:

- as ferramentas são executadas com sucesso e produzem outputs válidos;
- os outputs são corretamente capturados e armazenados na Evidence Layer;
- os dados são transformados em estruturas normalizadas consistentes;
- os mecanismos de validação garantem a integridade dos dados produzidos.

Adicionalmente, os testes end-to-end demonstraram que o sistema consegue percorrer todas as etapas do pipeline:

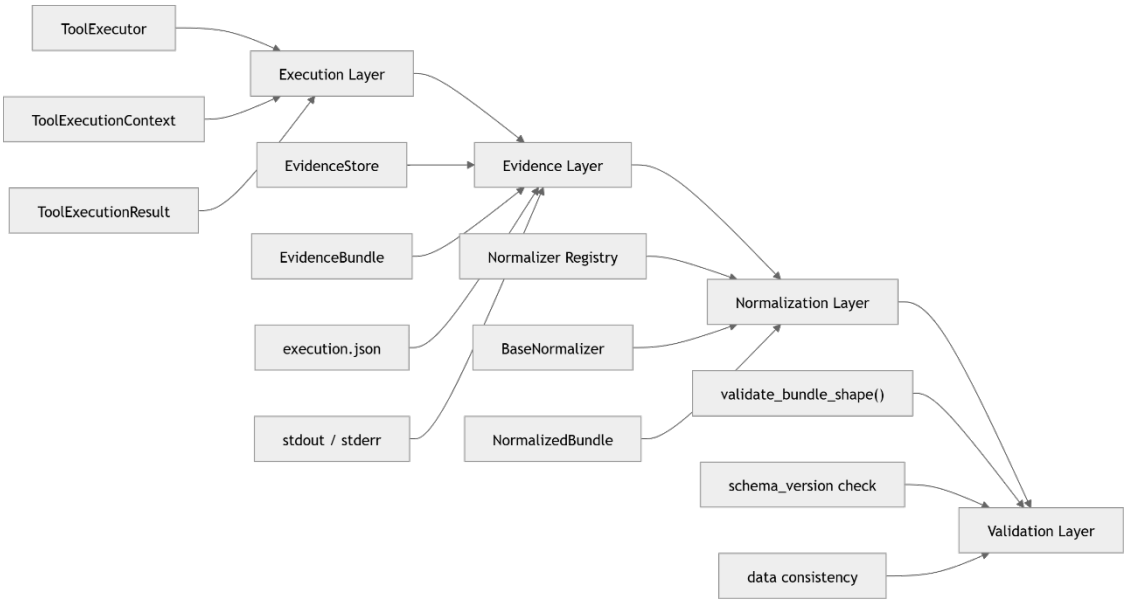

Figura 6.4.1 - Etapas De Teste

sem falhas críticas, produzindo resultados coerentes.

Estes resultados confirmam a viabilidade técnica do sistema na fase de reconhecimento.

6.5 Limitações da Validação Atual

Apesar dos resultados positivos, a validação realizada apresenta algumas limitações.

Em primeiro lugar, os testes incidem apenas sobre a fase de reconhecimento, não abrangendo ainda as restantes fases do pentest.

Em segundo lugar, os testes são realizados num ambiente controlado, podendo não refletir totalmente a complexidade de cenários reais.

Adicionalmente, a ausência de uma Persistence Layer completa limita a capacidade de validação ao nível da gestão de estado do sistema.

Por fim, o sistema ainda não inclui mecanismos avançados de decisão autónoma, o que reduz a complexidade dos cenários testados.

Estas limitações serão abordadas nas fases seguintes do projeto.

Bibliografia

[DEISI21] DEISI, Regulamento de Trabalho Final de Curso, Set. 2021.

[TaWe20] Tanenbaum,A. e Wetherall,D., *Computer Networks*, 6ª Edição, Prentice Hall, 2020.

[ULHT21] Universidade Lusófona de Humanidades e Tecnologia, www.ulusofona.pt, acedido em Out. 2021.

[GVR24] Grand View Research, Penetration Testing Market Size, Share & Trends, 2024–2030, 2024. Disponível em: https://www.grandviewresearch.com/industry-analysis/penetration-testing-market-report

[GMI24] Global Market Insights, Penetration Testing Market Size, Share & Trends Report 2024–2032, 2024. Disponível em: https://www.gminsights.com/industry-analysis/penetration-testing-market

[GMI24b] Global Market Insights, Penetration Testing as-a-Service Market to exceed $7.1 Bn by 2032, Says Global Market Insights Inc., 2024. Disponível em: https://www.globenewswire.com/news-release/2024/11/20/2984328/0/en/Penetration-Testing-as-a-Service-Market-to-exceed-7-1-Bn-by-2032-Says-Global-Market-Insights-Inc.html

[IMARC24] IMARC Group, Security Testing Market Report 2024–2033, 2024. Disponível em: https://www.imarcgroup.com/security-testing-market

[CyVe24] Cybersecurity Ventures, Penetration Testing Statistics 2024, 2024. Disponível em: https://cybersecurityventures.com/penetration-testing-statistics-2024/

[Cyph24] Cyphere, Penetration Testing Statistics: Key Industry Insights, 2024. Disponível em: https://thecyphere.com/blog/penetration-testing-statistics/

[Pent23] Pentera, The State of Pentesting 2023: Survey Report, 2023. Disponível em: https://pentera.io/wp-content/uploads/2024/01/2023-state-of-pentesting-survey-report-1.pdf

[Pent24] Pentera, The State of Pentesting 2024: Survey Report, 2024. Disponível em: https://pentera.io/resources/reports/the-state-of-pentesting-2024-survey-report/

[Coba24] Cobalt, State of Pentesting Report 2024, 2024. Disponível em: https://www.cobalt.io/hubfs/State%20of%20Pentesting%202024/State-of-Pentesting-Report-2024_Cobalt.pdf

[THN24] The Hacker News, Pentera's 2024 report reveals hundreds of security exposures in enterprise environments, 2024. Disponível em: https://thehackernews.com/2024/04/penteras-2024-report-reveals-hundreds.html

[MSSP24] MSSP Alert, Pentesting study exposes security gaps, signals MSSP prospects, 2024. Disponível em: https://www.msspalert.com/news/pentesting-study-exposes-security-gaps-signals-mssp-prospects

[UnAI24] Unite.AI, 2024 Cybersecurity Outlook: Key Takeaways from Pentera's State of Pentesting Report, 2024. Disponível em: https://www.unite.ai/2024-cybersecurity-outlook-key-takeaways-from-penteras-state-of-pentesting-report/

[ISC24a] ISC2, ISC2 Cybersecurity Workforce Study 2024, 2024. Disponível em: https://www.isc2.org/Insights/2024/10/ISC2-2024-Cybersecurity-Workforce-Study

[ISC24b] ISC2, 2024 Cybersecurity Workforce Study (versão PDF), 2024. Disponível em: https://edu.arrow.com/media/wtjfmszx/2024-isc2-wfs.pdf

[ISC23] ISC2, ISC2 Reveals Growth in Global Cybersecurity Workforce, But Record-Breaking Gap of 4 Million Cybersecurity Professionals Looms, 2023. Disponível em: https://www.isc2.org/Insights/2023/10/ISC2-Reveals-Workforce-Growth-But-Record-Breaking-Gap-4-Million-Cybersecurity-Professionals

[UKGov23] UK Department for Science, Innovation and Technology, Cyber security skills in the UK labour market 2023, 2023. Disponível em: https://www.gov.uk/government/publications/cyber-security-skills-in-the-uk-labour-market-2023

[IBM24] IBM, Cost of a Data Breach Report 2024, 2024. Disponível em: https://www.ibm.com/reports/data-breach

[BeEl12] Begley, C. G., Ellis, L. M., Drug development: Raise standards for preclinical cancer research, Nature, 483(7391), 531–533, 2012. DOI: 10.1038/483531a. Disponível em: https://www.culturecollections.org.uk/culture-collection-news/reproducibility-in-pre-clinical-life-science-research/

[OSC15] Open Science Collaboration, Estimating the reproducibility of psychological science, Science, 349(6251), aac4716, 2015. Disponível em: https://gwern.net/doc/statistics/bias/2015-opensciencecollaboration.pdf

[OWASP25] OWASP, GenAI Red Teaming Guide, 2025. Disponível em: https://breached.company/content/files/2025/02/genairedteamguideowasp.pdf

[TeMa25] TechMagic, Automated vs Manual Penetration Testing: What's the Difference?, 2025. Disponível em: https://www.techmagic.co/blog/automated-vs-manual-penetration-testing-whats-the-difference

[CSA24] Cloud Security Alliance, Aligning Security Testing with IT Infrastructure Changes, 2024. Disponível em: https://cloudsecurityalliance.org/blog/2024/10/03/aligning-security-testing-with-it-infrastructure-changes

Anexo 1 – Questionário

Formulário de declaração de uso de ferramentas de Inteligência Artificial a anexar a relatório

Todos os relatórios deverão incluir anexo com cópia, devidamente preenchida, do formulário abaixo.

Assinalar as opções aplicáveis e completar os campos solicitados.

1. Utilização de IA

[ ] Não foram utilizadas ferramentas de IA na realização deste trabalho.
[X] Foram utilizadas ferramentas de IA na realização deste trabalho.

---

2. Ferramentas utilizadas

Assinalar todas as que se aplicam.

Assistência geral à escrita, análise ou ideação

[X] ChatGPT
[ ] Microsoft Copilot
[ ] Gemini
[ ] Claude
[ ] Perplexity
[ ] Outras. Quais? _____

Assistência à programação / desenvolvimento

[ ] GitHub Copilot
[ ] Claude
[ ] OpenAI Codex
[ ] Cursor
[ ] Tabnine
[ ] Amazon CodeWhisperer / Amazon Q
[X] Outras. Quais? __Chat GPT

Geração de imagem / design / multimédia

[ ] DALL·E
[ ] Midjourney
[ ] Stable Diffusion
[ ] Canva AI / Magic Design
[ ] Outras. Quais? _____

Outros usos

[ ] Contexto: Ferramentas? _____

3. Fases do trabalho em que foi utilizada IA

[ ] Planeamento do trabalho
[X] Pesquisa exploratória / levantamento inicial de informação
[ ] Documentação técnica
[ ] Redação do relatório
[ ] Desenho / modelação / arquitetura
[ ] Design / prototipagem / interface
[ ] Geração de código
[X] Revisão / refatoração / debugging de código
[ ] Criação de testes / casos de teste
[X] Análise de resultados
[ ] Preparação de apresentação ou materiais auxiliares
[ ] Outros. Quais? _____

---

4. Tipo de utilização

Descrever sucintamente como a IA foi utilizada.

Exemplos: brainstorming, estruturação de secções, revisão linguística, sugestão de arquitetura, geração de exemplos, explicação de conceitos, geração parcial de código, correção de erros, criação de casos de teste, apoio ao design.

- brainstorming e estruturação de ideias
- explicação de conceitos técnicos relacionados com cibersegurança
- revisão e melhoria da qualidade do texto técnico.
- revisão e melhoria da qualidade de código.
- correção de erros.

---

5. Partes do trabalho afetadas

Indicar as secções, componentes, módulos, ficheiros, entregáveis ou atividades que foram influenciados pelo uso de IA.

- Revisão do relatório
- Revisão de erros nos testes
- Revisão de código

---

6. Exemplos de *prompt*

Inserir exemplos de *prompt*, diferenciando por âmbito (enquadrado na questão 2) e fase (enquadrado na questão 4)

“Chat escrevi esta secção do relatório sobre o trabalho realizado até agora. Sugeres alguma melhoria na escrita?”

“Este é o meu código de teste que testa tudo até a fase em que me encontro no desenvolvimento. É possível verificares se não me esta a faltar alguma situação de teste“

7. Validação, revisão e intervenção dos autores

Descrever que verificação, revisão, correção, adaptação ou reescrita foi realizada pelos autores.

Nota: se a IA tiver sido usada em código, testes, scripts, modelos, consultas, configurações ou outros artefactos técnicos, deve ser indicado de que forma os autores validaram o funcionamento e confirmaram a sua compreensão.

Em relação a código foi testado em ambiente de desenvolvimento.

---

8. Grau de utilização

[ ] Residual
[x] Moderado
[ ] Extensivo

[ ] Utilização homogénea
[ ] Grau de uso diferenciado por fase ou componente de trabalho

Descrever sucintamente os diferentes usos.

9. Trabalhos em parceria

Protecção de dados confidenciais e recursos proprietários de parceiros

[ ] O trabalho foi realizado em parceria com entidade externa ao DEISI

No caso da resposta anterior ser verdadeira, responder às seguintes questões:

[ ] O parceiro tem regras para restringir submissão de dados
[ ] As submissões validam aplicação de regras de tratamento de dados
[ ] Foram implementados mecanismos para restringir a partilha de recursos proprietários

10. Declaração de responsabilidade

Ao assinarem a presente declaração, os autores declaram que:

- a informação acima é verdadeira e reflete o uso efetivo de ferramentas de IA na realização do trabalho;
- compreendem que a IA não substitui autoria nem responsabilidade académica;
- verificaram a validaram e veracidade das referências bibliográficas incluídas no relatório
- assumem integralmente a responsabilidade técnica, científica, ética e académica por todo o conteúdo submetido, incluindo texto, código, modelos, testes, imagens, diagramas e restantes artefactos entregues.

11. Identificação dos autores

Nome(s): _____Pedro Ramos_____

Número(s): __A22309710_____

Data: __03__ / _04___ / ____2026____

Assinatura(s): _____Pedro Ramos_____

Glossário

APT (Venom APT – Autonomous Pentesting Tool)
No contexto deste trabalho, designa a ferramenta autónoma de pentesting em ambiente laboratorial, com motor de decisão condicional e guardrails de segurança. (Nota: em cibersegurança, APT é também usado para *Advanced Persistent Threat*, mas aqui o significado é "Autonomous Pentesting Tool".)

Attack Path (Caminho de Ataque)
Sequência encadeada de passos e nós (hosts, serviços, credenciais, técnicas) que conduz de um ponto inicial de compromisso ("patient zero") até um ativo de alto valor.

BAS (Breach and Attack Simulation)
Categoria de plataformas que automatizam a execução recorrente de cenários de ataque em ambientes de produção, para validar controlos de segurança e exposição a risco.

Blue Team
Equipa responsável pela defesa, monitorização e resposta a incidentes de segurança, frequentemente integrada num SOC.

CALDERA
Framework open-source da MITRE para *adversary emulation* automatizada baseada na matriz MITRE ATT&CK, usada para simular comportamentos de atacantes em ambientes controlados.

CLI (Command Line Interface)
Interface de linha de comandos utilizada para interagir com a ferramenta (por exemplo, arrancar sessões de pentest, consultar estado e resultados).

CVE (Common Vulnerabilities and Exposures)
Sistema de identificação normalizada de vulnerabilidades publicamente conhecidas, atribuindo a cada vulnerabilidade um identificador único.

Digital Twin ("gémeo digital")
Réplica virtual ou ambiente de laboratório que imita, de forma controlada, um ambiente de produção real, permitindo realizar testes de segurança com risco reduzido.

ENISA (European Union Agency for Cybersecurity)
Agência da União Europeia para Cibersegurança, responsável por relatórios como o *ENISA Threat Landscape*, que analisam tendências de ameaças.

Guardrails
Conjunto de restrições técnicas e políticas (allowlist, limites de tempo, modos "safe", kill-switch, regras de escopo) que garantem que o Venom APT opera apenas dentro dos limites autorizados e de forma ética.

JSON / CSV / YAML
Formatos textuais estruturados usados para representação e troca de dados. JSON (*JavaScript Object Notation*) e CSV (*Comma-Separated Values*) são usados em exportações e logs; YAML é usado para ficheiros de configuração e perfis de ataque.

KPI (Key Performance Indicator)
Indicador quantitativo utilizado para medir o desempenho da solução, nomeadamente

eficiência temporal, densidade de cobertura, reprodutibilidade e conformidade com guardrails.

Kill Chain (Cyber Kill Chain)
Modelo que descreve as fases de um ataque cibernético (por exemplo, reconhecimento, weaponization, entrega, exploração, instalação, comando e controlo, ações sobre o objetivo).

Kill-switch
Mecanismo de segurança que permite interromper de forma imediata e centralizada todas as atividades da ferramenta, terminando sessões e agentes em curso.

Knowledge Base (Base de Conhecimento)
Estrutura de dados mantida pela ferramenta com informação sobre hosts, serviços, credenciais, vulnerabilidades e resultados de execuções, utilizada pelo motor de decisão para escolher próximos passos.

LEI (Licenciatura em Engenharia Informática)
Curso de 1.º ciclo em Engenharia Informática, no âmbito do qual é desenvolvido o presente Trabalho Final de Curso.

LIG (Licenciatura em Informática de Gestão)
Curso de 1.º ciclo em Informática de Gestão, referenciado no contexto institucional da escola.

MITRE
Organização de investigação sem fins lucrativos que mantém, entre outros, a matriz MITRE ATT&CK e o framework CALDERA, amplamente utilizados em cibersegurança ofensiva e defensiva.

MITRE ATT&CK
Base de conhecimento estruturada sobre táticas, técnicas e procedimentos (TTPs) utilizados por adversários, usada como linguagem comum para descrever, mapear e avaliar cenários de ataque.

OWASP (Open Worldwide Application Security Project)
Comunidade aberta focada em segurança de aplicações, responsável por documentos como o *OWASP Testing Guide* e o *OWASP Top 10*.

Pentest / Pentesting / Teste de Intrusão
Processo controlado de simulação de ataques para avaliar a segurança de sistemas, redes e aplicações, identificando vulnerabilidades e demonstrando o seu impacto, seguindo metodologias como PTES, OWASP Testing Guide e MITRE ATT&CK.

Pentester
Profissional especializado em testes de intrusão, responsável por definir o escopo, conduzir os ataques controlados, recolher evidências e produzir relatórios técnicos.

PTaaS (Penetration Testing as a Service)
Modelo de prestação de serviços de pentesting recorrentes e altamente automatizados, tipicamente em regime *as a service*, com campanhas regulares de ataque controlado.

PTES (Penetration Testing Execution Standard)
Standard que estrutura a execução de testes de intrusão em fases bem definidas (pré-

engagement, recolha de informação, modelação de ameaças, exploração, pós-exploração e reporting).

Red Team
Equipa ofensiva responsável por emular atacantes reais, testando a eficácia dos controlos e a capacidade de deteção e resposta da organização.

Security Operations Center (SOC)
Centro operacional de segurança responsável por monitorizar eventos, detetar incidentes, coordenar resposta e manter a visibilidade contínua do estado de segurança da organização.

SIEM (Security Information and Event Management)
Classe de ferramentas que agregam, normalizam e correlacionam logs e eventos de segurança, suportando deteção de incidentes, *threat hunting* e reporting.

Seed (Semente)
Valor inicial (configuração, perfil, parâmetros) usado para tornar execuções reproduzíveis, permitindo comparar sessões em condições controladas.

Stakeholder
Parte interessada no projeto (por exemplo, pentesters, Blue Team, gestores de risco, clientes, instituição académica), que pode fornecer requisitos e feedback.

Threat Intelligence
Conjunto de informações estruturadas sobre ameaças, campanhas, TTPs e indicadores de compromisso (IoCs), usado para orientar testes de segurança e melhorar a deteção e resposta.

TTPs (Tactics, Techniques and Procedures)
Táticas, técnicas e procedimentos utilizados por atacantes; no Venom APT, as ações ofensivas são mapeadas para TTPs específicos da matriz MITRE ATT&CK para efeitos de cobertura e reporting.

TFC (Trabalho Final de Curso)
Projeto académico que consolida competências adquiridas ao longo da licenciatura; no presente caso, materializado no desenho, implementação e avaliação do Venom APT.

Vulnerabilidade
Fraqueza num sistema, aplicação, configuração ou processo que pode ser explorada por um atacante para comprometer confidencialidade, integridade ou disponibilidade.

VLAN (Virtual Local Area Network)
Segmentação lógica de rede usada para isolar ambientes de laboratório do resto da infraestrutura de produção.

[1] Cybersecurity Ventures, "Penetration Testing Statistics 2024", 2024. Disponível em: https://cybersecurityventures.com/penetration-testing-statistics-2024/

[2] MARC Group, "Security Testing Market Report 2024–2033", 2024. Disponível em: https://www.imarcgroup.com/security-testing-market.

[3] Cyphere, "Penetration Testing Statistics: Key Industry Insights", 2024. Disponível em: https://thecyphere.com/blog/penetration-testing-statistics/

[4] Global Market Insights, "Penetration Testing Market Size, Share & Trends Report 2024–2032", 2024. Disponível em: https://www.gminsights.com/industry-analysis/penetration-testing-market

[5] ISC2, *2024 Cybersecurity Workforce Study*, 2024. Disponível em: https://www.isc2.org/Insights/2024/10/ISC2-2024-Cybersecurity-Workforce-Study

[6] ISC2, *2024 Cybersecurity Workforce Study*, 2024, p. X. Disponível em: https://edu.arrow.com/media/wtjfmszx/2024-isc2-wfs.pdf

[7] IBM, *Cost of a Data Breach Report 2024*, 2024. Disponível em: https://www.ibm.com/reports/data-breach.

[8] Department for Science, Innovation and Technology, *Cyber security skills in the UK labour market 2023*, UK Government, 2023. Disponível em: https://www.gov.uk/government/publications/cyber-security-skills-in-the-uk-labour-market-2023.

[9] Pentera, *The State of Pentesting 2024: Survey Report*, 2024. Disponível em: https://pentera.io/resources/reports/the-state-of-pentesting-2024-survey-report/

[10] Aligning Security Testing with IT Infrastructure Changes. Disponível em: https://cloudsecurityalliance.org/blog/2024/10/03/aligning-security-testing-with-it-infrastructure-changes

[11] Pentera, *The State of Pentesting 2023: Survey Report*, 2023. Disponível em: https://pentera.io/wp-content/uploads/2024/01/2023-state-of-pentesting-survey-report-1.pdf

[12] Cobalt, *The State of Pentesting Report 2024*, 2024. Disponível em: https://www.cobalt.io/hubfs/State%20of%20Pentesting%202024/State-of-Pentesting-Report-2024_Cobalt.pdf

[13] Pentera, *The State of Pentesting 2023: Survey Report*, 2023. Disponível em: https://pentera.io/wp-content/uploads/2024/01/2023-state-of-pentesting-survey-report-1.pdf

[14] Cobalt, *The State of Pentesting Report 2024*, 2024. Disponível em: https://www.cobalt.io/hubfs/State%20of%20Pentesting%202024/State-of-Pentesting-Report-2024_Cobalt.pdf

[15] The Hacker News, "Pentera's 2024 report reveals hundreds of security exposures in enterprise environments", 2024. Disponível em: https://thehackernews.com/2024/04/penteras-2024-report-reveals-hundreds.html

[16] WASP, *GenAI Red Teaming Guide*, 2025. Disponível em: https://breached.company/content/files/2025/02/genairedteamguideowasp.pdf

[17] TechMagic, "Automated vs Manual Penetration Testing: What's the Difference?", 2025. Disponível em: https://www.techmagic.co/blog/automated-vs-manual-penetration-testing-whats-the-difference

[18] Collberg, C., Proebsting, T. A., Moraila, G., Shankaranarayana, A., Shi, Z., & Warren, A. (2015). Measuring Reproducibility in Computer Systems Research. ACM SIGOPS Operating Systems Review, 49(1), 3–16. Disponível em: https://reproducibility.cs.arizona.edu/v2/RepeatabilityTR.pdf

[19] Vandewalle, P., Kovacevic, J., & Vetterli, M. (2009). Reproducible research in signal processing: What, why, and how. IEEE Signal Processing Magazine, 26(3), 37–47. Disponível em: https://infoscience.epfl.ch/server/api/core/bitstreams/1560c1d5-8b1b-4a30-b7bf-27d8d99b9281/content

[20] Collberg, C., Proebsting, T. A., Moraila, G., Shankaranarayana, A., Shi, Z., & Warren, A. (2015). Measuring Reproducibility in Computer Systems Research. ACM SIGOPS Operating Systems Review, 49(1), 3–16. Disponível em: https://reproducibility.cs.arizona.edu/v2/RepeatabilityTR.pdf

[21] Pentera, *The State of Pentesting 2024: Survey Report*, 2024. Disponível em: https://www.unite.ai/2024-cybersecurity-outlook-key-takeaways-from-penteras-state-of-pentesting-report/

[22] MSSP Alert, "Pentesting study exposes security gaps, signals MSSP prospects", 2024. Disponível em: https://www.msspalert.com/news/pentesting-study-exposes-security-gaps-signals-mssp-prospects
[23] Applebaum, A., Miller, B., Strom, B., Korban, C., & Wolf, R. (2016). Automated Adversary Emulation: A Case for Planning and Acting with Unknowns. MITRE Corporation.
Disponível em: https://www.mitre.org/sites/default/files/2021-11/prs-18-0944-1-automated-adversary-emulation-planning-acting.pdf
[24] MITRE Corporation. (2018). CALDERA: Automated Adversary Emulation Platform. Disponível em: https://github.com/mitre/caldera

[25] Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2011). Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. Lockheed Martin. Disponível em:https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf

[26] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2020). A Comprehensive Survey on Graph Neural Networks. IEEE Transactions on Neural Networks and Learning Systems.
Disponível em: https://arxiv.org/abs/1901.00596

[27] Pentera, *The State of Pentesting 2023: Survey Report*, 2023. Disponível em: https://pentera.io/wp-content/uploads/2024/01/2023-state-of-pentesting-survey-report-1.pdf

[28] Cobalt, *The State of Pentesting Report 2024*, 2024. Disponível em: https://www.cobalt.io/hubfs/State%20of%20Pentesting%202024/State-of-Pentesting-Report-2024_Cobalt.pdf

[29] MITRE Corporation. (2018). CALDERA: Automated Adversary Emulation Platform. Disponível em: https://github.com/mitre/caldera

[30] MITRE Corporation. (2018). CALDERA: Automated Adversary Emulation Platform. Disponível em: https://github.com/mitre/caldera

[31] Collberg, C., Proebsting, T. A., Moraila, G., Shankaranarayana, A., Shi, Z., & Warren, A. (2015). Measuring Reproducibility in Computer Systems Research.
ACM SIGOPS Operating Systems Review, 49(1), 3–16. Disponível em:
https://reproducibility.cs.arizona.edu/v2/RepeatabilityTR.pdf
[32] MITRE Corporation. (2018). CALDERA: Automated Adversary Emulation Platform. Disponível em: https://github.com/mitre/caldera

[33]Cobalt Strike: Adversary Simulations and Red Team Operations. Disponível em: https://www.cobaltstrike.com

[34] Penetration Testing Market (2024 - 2030)Disponível em:
https://www.grandviewresearch.com/industry-analysis/penetration-testing-market-report
[35] Penetration Testing as-a-Service Market to exceed $7.1 Bn by 2032, Says Global Market Insights Inc. Disponivel em: https://www.globenewswire.com/news-release/2024/11/20/2984328/0/en/Penetration-Testing-as-a-Service-Market-to-exceed-7-1-Bn-by-2032-Says-Global-Market-Insights-Inc.html
[36] ISC2 Reveals Growth in Global Cybersecurity Workforce, But Record-Breaking Gap of 4 Million Cybersecurity Professionals Looms. Disponivel em: https://www.isc2.org/Insights/2023/10/ISC2-Reveals-Workforce-Growth-But-Record-Breaking-Gap-4-Million-Cybersecurity-Professionals
[37] Penetration Testing Market (2024 - 2030). Disponivel em:
https://www.grandviewresearch.com/industry-analysis/penetration-testing-market-report
[38] Penetration Testing as-a-Service Market to exceed $7.1 Bn by 2032, Says Global Market Insights Inc. Disponivel em: https://www.globenewswire.com/news-release/2024/11/20/2984328/0/en/Penetration-Testing-as-a-Service-Market-to-exceed-7-1-Bn-by-2032-Says-Global-Market-Insights-Inc.html

[39] ISC2 Reveals Growth in Global Cybersecurity Workforce, But Record-Breaking Gap of 4 Million Cybersecurity Professionals Looms. Disponivel em: https://www.isc2.org/Insights/2023/10/ISC2-Reveals-Workforce-Growth-But-Record-Breaking-Gap-4-Million-Cybersecurity-Professionals