KAUST Academy Summer Cybersecurity Program
Capstone Project Final Report

APT-29 LOTL Adversary Emulation in Active Directory Environment, Threat Hunting, and Incident Response

Abdullah Alsaleh, Ahmed Almuhaisin, Murtadha Alsaleh, Hassan Ibrahim, Essam Alsulayyih
Mentor: Mudassar Aslam
abdullah.alsaleh@kaust.edu.sa, ahmed.almuhaisin@kaust.edu.sa, Murtadha.alsaleh@kaust.edu.sa, hassan.ibrahim@kaust.edu.sa, essam.alsulayyih@kaust.edu.sa, maslam@brookes.ac.uk

KAUST Academy Summer Cybersecurity Program

Abstract

This report documents the end-to-end design, execution, and evaluation of a full purple-team exercise built around the tactics, techniques, and procedures (TTPs) of APT29 (Cozy Bear / Midnight Blizzard). A five-host, isolated Active Directory (AD) environment was constructed across a shared 172.20.10.0/24 subnet, comprising a Windows Server 2022 domain controller, two privilege-differentiated Windows 10 workstations, a MITRE CALDERA adversary-emulation server, and an Elastic Security SIEM stack. A behavioral telemetry pipeline (Sysmon, PowerShell Script Block/Transcription logging, and Elastic Agent/Fleet) was deployed and empirically validated before any offensive activity began.

A custom eleven-technique attack chain — spanning initial access, execution, discovery, persistence, credential access, lateral movement, collection, and exfiltration — was built as a CALDERA adversary profile and executed against the environment. In parallel, sixteen MITRE ATT&CK-mapped detection rules were engineered in Elastic Security, tuned against five distinct sources of false positives, and validated technique-by-technique. A dedicated honeypot account was deployed as a deception control. The final emulation run produced 39 alerts with 11/11 (100%) technique coverage across the attack chain and a measured mean time to detect (MTTD) of 18.4 seconds. Six threat-hunting investigations subsequently correlated the isolated alerts into a single coherent attack timeline, and a full incident-response cycle — detection and triage, containment, eradication, recovery, and post-incident review — was carried out against the NIST SP 800-61 Rev. 2 framework.

The exercise is presented alongside a root-cause analysis of the environmental weaknesses that allowed the attack chain to proceed unimpeded, and a set of lessons learned on behavioral detection engineering against living-off-the-land tradecraft. The results validate the central premise of the project: signature-based defenses are insufficient against an adversary that relies on native operating-system tooling, and durable detection requires correlating process lineage, command-line content, registry activity, and network destinations rather than blocking tools by name.

Index Terms *— APT29, Living-off-the-Land, Active Directory, MITRE ATT&CK, MITRE CALDERA, Elastic Security, SIEM, Threat Hunting, Purple Team, Incident Response, NIST SP 800-61.*

Table of Contents

I. Introduction 4

A. Motivation 4
B. Objectives 4
C. Report Structure 5

II. Related Work 5

A. APT29 and the MITRE ATT&CK Framework 5
B. Adversary Emulation with MITRE CALDERA 5
C. SIEM Platform Selection 5
D. Endpoint Telemetry 6
E. Incident Response Frameworks 6

III. Methodology 6

A. Scenario Selection 6
B. SIEM Platform Comparison 6
C. Lab Architecture 7
D. Active Directory Design 9
E. Adversary Emulation Platform Setup 11
F. Logging and Telemetry Pipeline 12
G. Validation Approach 13
H. Adversary Profile Design 14
I. Detection Rule Engineering Methodology 17
J. Deception Technology — Honeypot Account 20
K. Threat-Hunting Methodology 21
L. Incident-Response Methodology 21

IV. Results 21

A. Environment Build-Out 21
B. Logging Pipeline Validation 21
C. Detection Engineering Results 22
D. Adversary Emulation Execution 24
E. Final Attack-Chain Run and Alert Summary 26
F. Threat Hunting Results 27
G. Incident Response Execution 29
H. Security Dashboards 31
I. Evaluation Metrics 32

V. Burhan: A Prototype for Unified SOC and IR Validation 33

A. Motivation 33
B. Extended Conceptual Lifecycle 33
C. Prototype Demonstration 33

D. Scope and Future Work 35
VI. Discussion 35
A. Distributed Lab Topology 35
B. SIEM Choice 35
C. Privilege Asymmetry by Design 35
D. Validate-Before-Attack Discipline 35
E. Detection Engineering as an Iterative Process 36
F. Defense in Depth in Detection 36
VII. Root Cause Analysis 36
A. Unrestricted LOLBin Execution at Initial Access 36
B. Absence of PowerShell Hardening Controls 36
C. Excessive Privilege Assignment 37
D. Weak Credential Hygiene and Absent Segmentation 37
E. Detective but Not Preventive Persistence Controls 37
F. No Network-Level Exfiltration Controls 37
G. Synthesis 37
VIII. Lessons Learned 38
A. Detection Relies on Context, Not Signatures 38
B. Script Block Logging Is the Highest-Fidelity Telemetry Source Available 38
C. Audit Policy Gaps Are a Common and Consequential Misconfiguration 38
D. Validation Must Be Empirical, Not Assumed 38
E. False-Positive Tuning Is Non-Trivial and Environment-Specific 38
F. Defense in Depth in Detection Provides Resilience 39
IX. Limitations and Future Work 39
A. Limitations 39
B. Future Work 39
X. Conclusion 40
References 41
Appendix A — Incident Response Playbook 42
A. Quick Reference — Artifacts and Hosts 42
B. Phase 1 — Detection and Triage 43
C. Phase 2 — Containment 43
D. Phase 3 — Eradication 44
E. Phase 4 — Recovery 45
F. Post-Incident Actions 46
Appendix B — Complete Detection Rule Catalogue 47

I. INTRODUCTION

Modern enterprise networks are routinely targeted by well-resourced, patient adversaries who favor stealth over speed. Nation-state groups such as APT29 — also tracked as Cozy Bear, the Dukes, and, most recently, Midnight Blizzard — are known for long-dwell-time intrusions, heavy use of legitimate administrative tools, and credential-based lateral movement rather than noisy malware deployment [1]. Defending against this style of adversary requires more than perimeter controls: it requires an organization to have the endpoint visibility, log pipeline, and detection-engineering capability to notice an attacker who is deliberately trying to look like a normal user.

This project was carried out as the capstone deliverable for the KAUST Academy Summer Cybersecurity Program. Rather than studying APT29's tradecraft purely on paper, the team built a small but realistic Active Directory environment and used it as a live testbed for adversary emulation and Security Operations Center (SOC) response, following the same core idea as purple-team exercises: red-team activity and blue-team detection engineering developed side by side against the same environment. The project was carried out over five weeks, progressing from environment design and telemetry validation, through iterative detection-rule engineering and full attack-chain execution, to threat hunting and a formal incident-response cycle.

A. Motivation

Three candidate capstone scenarios were initially proposed: a human-operated ransomware case modeled on Conti, an identity- and remote-access compromise modeled on the Colonial Pipeline incident, and a stealthy living-off-the-land (LOTL) intrusion modeled on APT29. The APT29-based scenario was selected because it offered the richest opportunity to practice detection engineering against identity infrastructure (Active Directory) and living-off-the-land techniques, which are considerably harder to detect than commodity malware and more representative of the threats a modern SOC team is expected to catch.

B. Objectives

The project set out to:

- Build an isolated, safe-to-attack Active Directory environment spanning a domain controller, victim workstations, an adversary-emulation platform, and a SIEM.
- Establish full endpoint visibility (process, PowerShell, and Sysmon telemetry) and empirically validate that logs reliably reach the SIEM before any offensive activity begins.
- Emulate a realistic subset of APT29 TTPs against the environment using MITRE CALDERA, spanning the full kill chain from initial access to exfiltration.
- Engineer MITRE ATT&CK-mapped detection rules and threat-hunting queries for every emulated technique, tuning them against real false positives.
- Deploy a deception control (honeypot account) to evaluate an additional, non-signature detection layer.
- Execute a structured, NIST-aligned incident-response cycle against the emulated intrusion and document the process as a reusable playbook.

- Evaluate the resulting SOC capability using quantitative detection-engineering metrics.

C. Report Structure

Section II reviews background on APT29, MITRE ATT&CK, MITRE CALDERA, and the endpoint telemetry sources used. Section III describes the lab architecture, tool selection, Active Directory design, attack-profile design, and detection-engineering methodology. Section IV presents the results: environment build-out, pipeline validation, the executed attack chain, the sixteen detection rules and their validation, threat-hunting outcomes, incident-response execution, and evaluation metrics. Section V introduces Burhan, a lightweight prototype extension developed after the original results to explore unified validation of emulation, detection, and response evidence. Section VI discusses the design decisions and their trade-offs. Section VII presents a root-cause analysis of the environmental weaknesses that enabled the attack chain. Section VIII distills the lessons learned during the exercise. Section IX covers limitations and remaining future work, and Section X concludes the report. Appendix A reproduces the incident-response playbook produced by the project in full, and Appendix B reproduces the complete detection-rule catalogue.

II. RELATED WORK

A. APT29 and the MITRE ATT&CK Framework

APT29 is a Russian state-sponsored threat actor, attributed to Russia's Foreign Intelligence Service (SVR), that has been active since at least 2008 [1]. It is best known for the 2020 SolarWinds supply-chain compromise, but its broader tradecraft is characterized by long dwell times, credential-based access, and heavy reliance on legitimate administrative tooling — "living off the land" — rather than custom malware, which makes it a demanding target for signature-based defenses. The MITRE ATT&CK framework [2] catalogues this and other groups' tactics, techniques, and procedures (TTPs) in a structured, publicly available knowledge base that is now the de facto standard vocabulary for describing adversary behavior and building detections around it. Every technique emulated in this project is mapped to its corresponding ATT&CK identifier so that detection coverage can be evaluated systematically rather than anecdotally.

B. Adversary Emulation with MITRE CALDERA

MITRE CALDERA [3] is an open-source adversary-emulation platform built directly on top of the ATT&CK framework. It provides an asynchronous command-and-control server, a plugin architecture for agents and TTP collections, and both autonomous and manually directed operation modes, which makes it well suited to purple-team exercises where red and blue activity are developed together against the same environment. Because CALDERA ships with well-known default agents and abilities, it is generally understood as a validation and training tool for exercising detection coverage rather than a covert offensive platform, which matches the goals of this project.

C. SIEM Platform Selection

Security Information and Event Management (SIEM) platforms are the backbone of any detection-engineering exercise: they centralize log collection, provide a query language for threat hunting, and host the alerting and detection-rule logic. The two open-source-friendly candidates considered were Elastic Security [4] and Wazuh; the comparison that informed the final choice is presented in Section III-B.

D. Endpoint Telemetry

Windows Sysmon [7] and native PowerShell Script Block, Module, and Transcription logging [6] are widely used, low-cost sources of high-fidelity endpoint telemetry for detecting process creation, command-line activity, and script-based attacks — both of which are central to how APT29-style actors operate once inside a network. This project relies on both as its primary visibility sources, forwarded to the SIEM via Elastic Agent and Fleet.

E. Incident Response Frameworks

The incident-response phase of this project follows the lifecycle defined in NIST Special Publication 800-61 Revision 2, "Computer Security Incident Handling Guide" [5]: preparation, detection and analysis, containment/eradication/recovery, and post-incident activity. This framework was chosen because it is vendor-neutral, widely adopted across enterprise SOCs, and maps cleanly onto the phase structure already used for the CALDERA attack chain, allowing the same technique-level timeline to be reused for both the offensive and defensive halves of the exercise.

III. METHODOLOGY

A. Scenario Selection

Three candidate scenarios were evaluated in Week 1: a human-operated ransomware scenario modeled on Conti, an identity- and remote-access compromise modeled on the Colonial Pipeline incident, and a stealthy living-off-the-land intrusion modeled on APT29. After discussion with the project mentor, the APT29 safe-emulation scenario was selected as the final project direction for the reasons outlined in Section I-A.

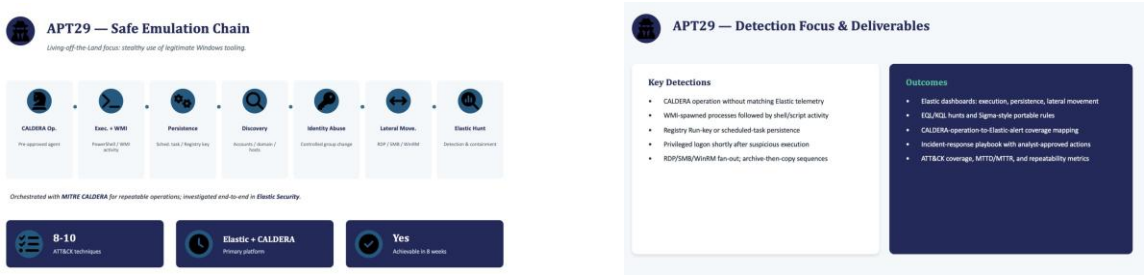


Fig. 1. Scenario-selection artifacts produced during Week 1: (left) the APT29 Safe Emulation Chain decision summary; (right) the resulting detection focus and expected deliverables.

B. SIEM Platform Comparison

Elastic Security and Wazuh were compared across eight criteria relevant to a detection-engineering-focused exercise, summarized in Table I.

TABLE I
Elastic Security vs. Wazuh Comparison

Feature	Elastic Security	Wazuh
Threat Hunting	Excellent (KQL & EQL)	Good
Dashboards & Visualization	Excellent	Good

Feature	Elastic Security	Wazuh
Detection Rules	Advanced & flexible	Good
MITRE ATT&CK Integration	Native & strong	Supported
CALDERA Integration	Better suited	Supported, less common
Performance	Better for large datasets	Good for small/medium labs
Learning Value	High for SOC analysts	High for SIEM fundamentals
Industry Adoption	Widely used in enterprise SOCs	Popular in open-source SOCs

Elastic Security was selected because it offered stronger threat-hunting capabilities (KQL/EQL), richer dashboards, tighter integration with MITRE CALDERA, and a more realistic enterprise SOC experience, making it the better fit for a project centered on adversary emulation and detection engineering. This choice is revisited critically in Section VI-B.

C. Lab Architecture

Rather than hosting every machine as a virtual machine on a single laptop, the team distributed one to two VMs per member's laptop and connected all machines over a shared local-area network on the 172.20.10.0/24 subnet. This was a deliberate choice: it spread the resource load across five machines instead of overloading one, and it more closely mirrors how a real network is distributed across physical hosts rather than a single hypervisor. The resulting architecture is summarized in Table II.

TABLE II
Lab Environment Architecture

Host	Owner	Role	OS	IP Address
DC01	Abdullah	Domain Controller (AD DS, DNS, file share)	Windows Server 2022	172.20.10.30
WS01	Essam	Victim workstation — Employee OU, standard user	Windows 10	172.20.10.40
WS02	Essam	Victim workstation — IT OU, elevated/admin	Windows 10	172.20.10.50
CALDERA01	Ahmed	Adversary emulation / C2 server	Ubuntu Desktop	172.20.10.20
ELASTIC01	Hassan	Elastic SIEM (Elastic Agent, Fleet, Kibana)	Ubuntu Server	172.20.10.10

All hosts were connected to the same local network, and reachability was validated with ICMP ping tests between every pair of machines before proceeding with configuration.

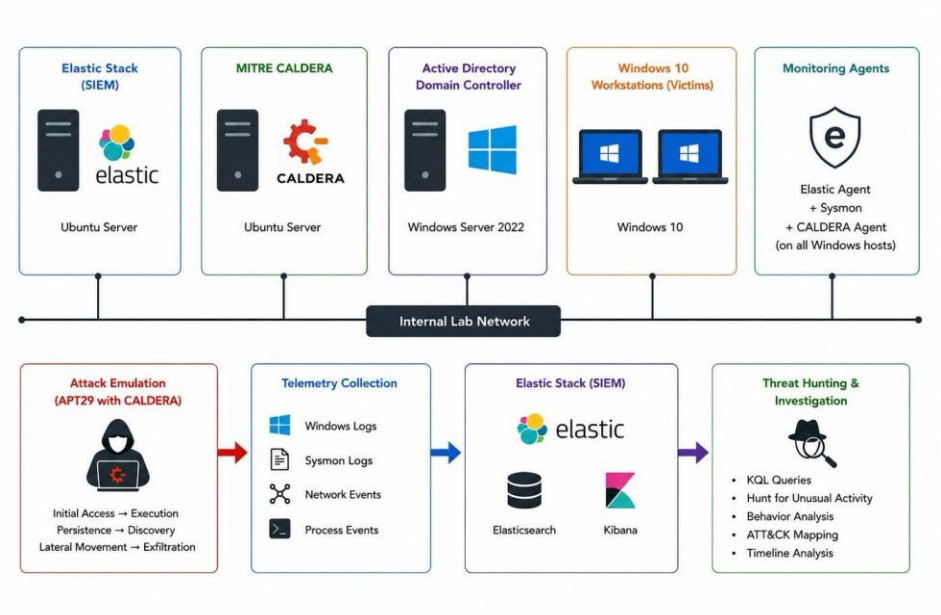


Fig. 2. Conceptual network layout distributing the five lab hosts — Elastic stack, MITRE CALDERA, the Active Directory domain controller, and two Windows 10 workstations — across team members' laptops on the shared internal lab network.

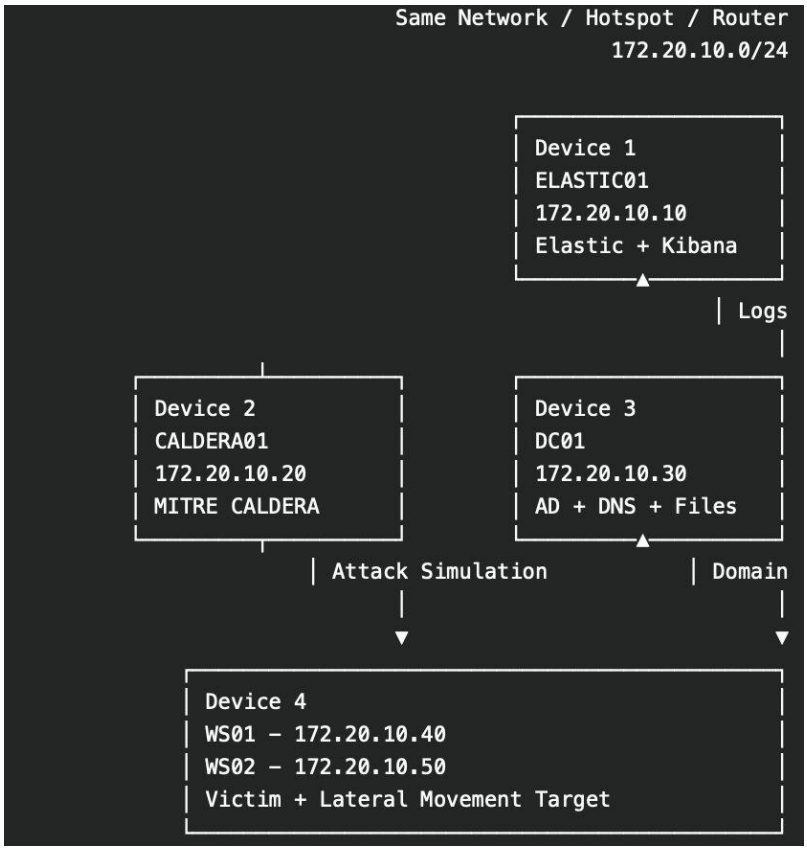


Fig. 3. Per-device role assignment on the shared network: ELASTIC01 (SIEM), CALDERA01 (adversary emulation), DC01 (Active Directory), and the two victim workstations, WS01 and WS02.

D. Active Directory Design

DC01 was configured as the Active Directory domain controller for the domain corp.local, also hosting DNS and a file share. Rather than leaving all accounts and computers in the default AD containers, dedicated Organizational Units (OUs) were created for the lab, reflecting standard enterprise practice: custom OUs allow Group Policy Objects (GPOs) to be scoped precisely, keep the directory organized, and mirror how real enterprise AD environments are structured.

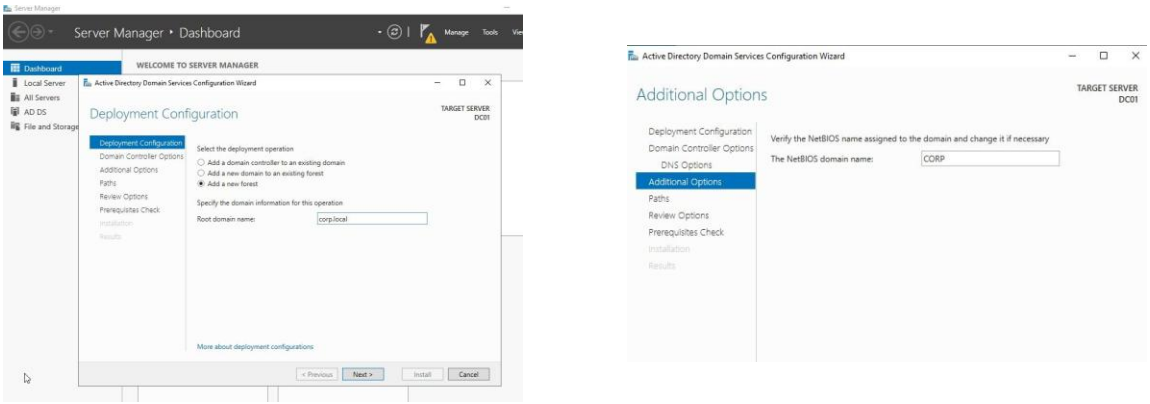


Fig. 4. Active Directory forest deployment: (left) root domain name configuration (corp.local); (right) NetBIOS domain name configuration (CORP).

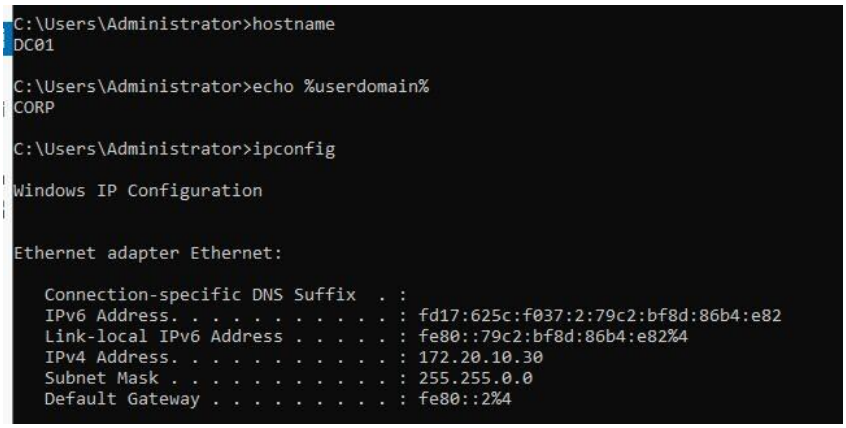


Fig. 5. Domain controller (DC01) role and system information confirming successful forest deployment.

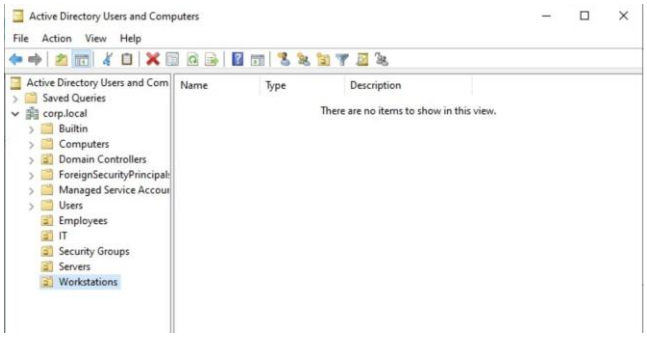


Fig. 6. Custom Organizational Units created inside the corp.local directory: Employees, IT, Security Groups, Servers, and Workstations.

Two OUs were used to create a deliberate privilege asymmetry between the victim workstations: WS01 was placed in the Employee OU and configured with standard (non-administrative) user privileges, while

WS02 was placed in the IT OU and configured with elevated (administrative) privileges. This mirrors a common real-world condition — an IT/support workstation with broader rights than a typical employee machine — that is a frequent lateral-movement and privilege-escalation target for identity-focused adversaries such as APT29. All lab AD user accounts were provisioned with a shared, lab-only password (Password123@) for setup convenience; this is appropriate only for an isolated training environment and is deliberately revisited as a root cause in Section VII.

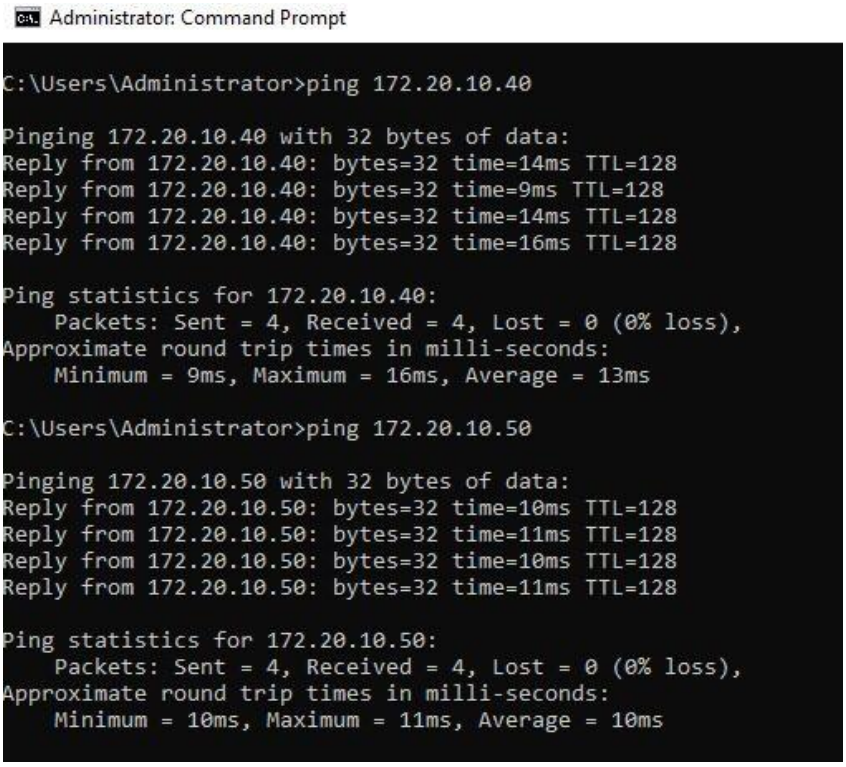


Fig. 7. Connectivity validation: ICMP ping tests confirming WS01 and WS02 reachability to DC01 over the lab LAN.

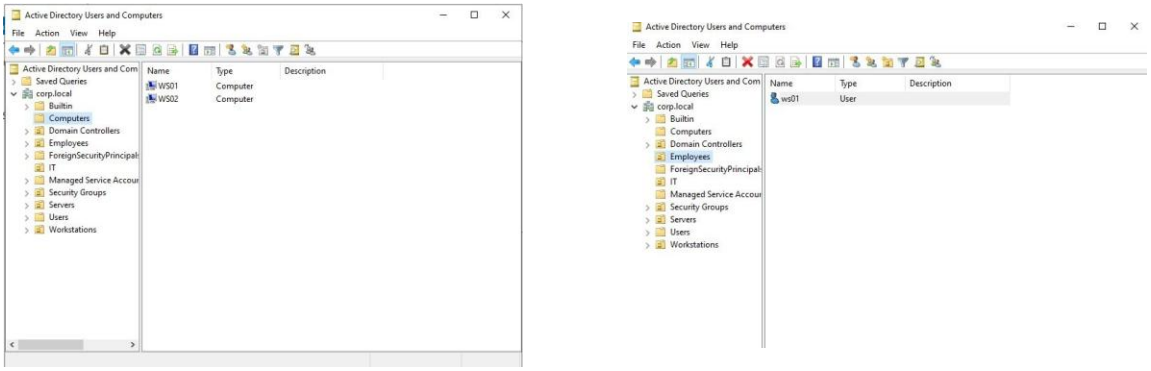


Fig. 8. (left) WS01 and WS02 registered as domain computer objects in Active Directory; (right) WS02 group membership confirming elevated (local Administrators) privilege for the IT OU workstation.

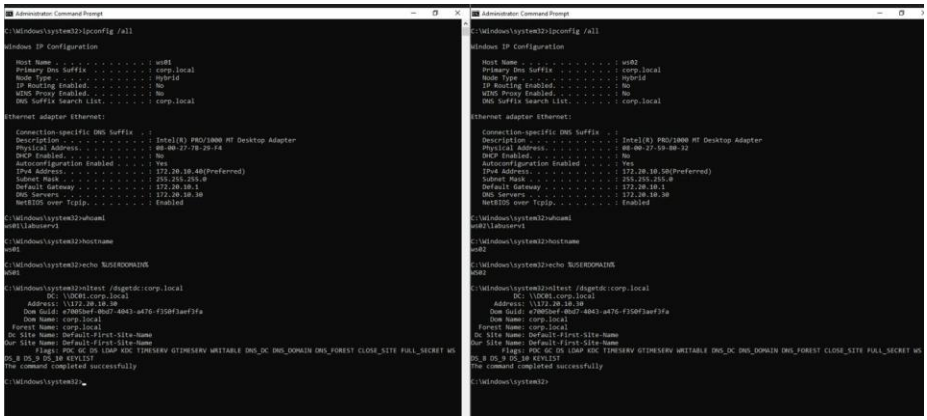


Fig. 9. IP configuration of WS01 and WS02, both pointed at DC01 for DNS resolution.

E. Adversary Emulation Platform Setup

CALDERA01 was initially deployed on Ubuntu Server, but configuration issues led the team to redeploy it on Ubuntu Desktop, which proved simpler to work with. The host was provisioned with Python 3.1, Node.js 22.23.1, and Go 1.24 to support building CALDERA agents, after which the CALDERA server was brought up and connected to the lab LAN. CALDERA was reachable at its web interface using the credentials username: red, password: admin. CALDERA agents were subsequently deployed to WS01 and WS02, and later to DC01, to enable emulation of adversary TTPs against those hosts.

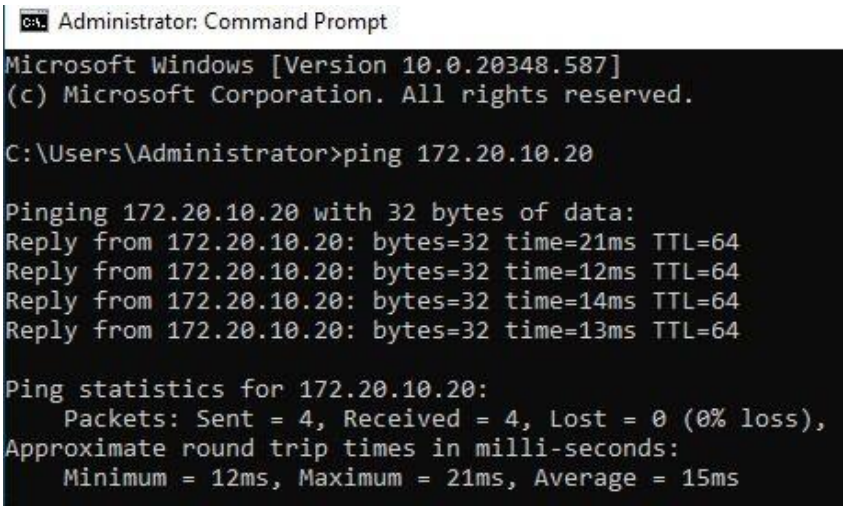


Fig. 10. Ping connectivity validated between DC01 and CALDERA01.



Fig. 11. (left) MITRE CALDERA server started on CALDERA01; (right) the CALDERA web dashboard reachable over the lab LAN prior to agent deployment.

F. Logging and Telemetry Pipeline

Endpoint visibility was built up in stages and validated at each step before moving on:

- Sysmon was deployed first to DC01, then to WS01 and WS02, to provide detailed process-creation, network-connection, and file-system event telemetry beyond what native Windows Event Logs offer.
- PowerShell Script Block Logging and Transcription were enabled on DC01, WS01, and WS02 via Group Policy pushed centrally from the domain controller, to capture the command-line and scripting activity typically used by living-off-the-land techniques.
- Elastic Agent was installed on all three Windows hosts and enrolled with a Fleet Server hosted on ELASTIC01 (172.20.10.10:8220), giving centralized policy management and log shipping into Elastic Security/Kibana.

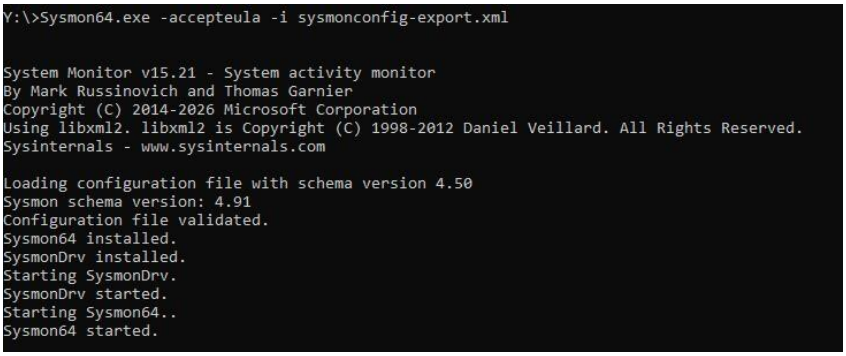


Fig. 12. Sysmon v15.21 (schema 4.91) successfully installed and running on a victim workstation.

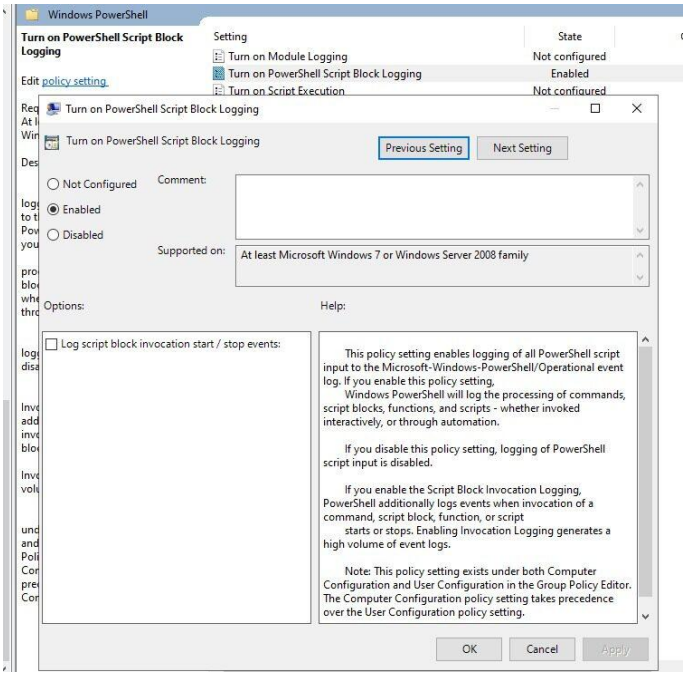


Fig. 13. PowerShell Script Block Logging and Transcription enabled via Group Policy, pushed centrally from DC01 to WS01 and WS02.

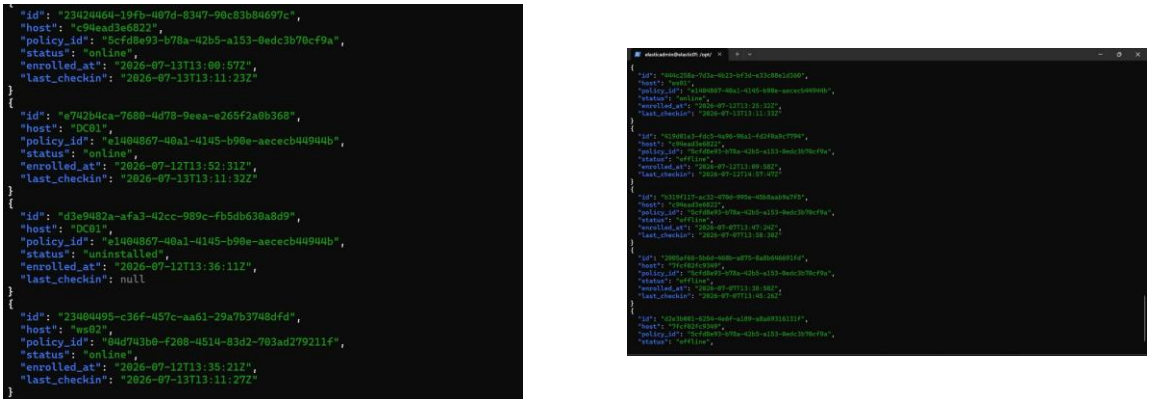


Fig. 14. Fleet Server confirming DC01, WS01, and WS02 agents enrolled and reporting Healthy/Connected status.

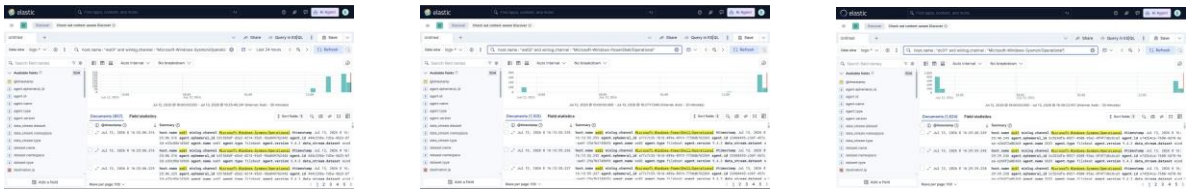


Fig. 15. Sysmon and PowerShell events confirmed visible in Kibana Discover on a per-host basis: WS01 (left), WS02 (center), and DC01 (right).

G. Validation Approach

Before any offensive activity was executed against the environment, the logging pipeline was validated end-to-end: confirming Fleet-reported agent health for all three Windows hosts, confirming that Sysmon and PowerShell events were arriving in Kibana for each host individually, and running a simple KQL query together with a benign, observable action (opening Calculator on WS02) to confirm that a known event

could be reliably located in the SIEM. This "trust but verify" step was treated as a prerequisite for any subsequent detection-engineering work, since detection rules built on top of an unvalidated pipeline would be meaningless. One transient issue was observed during this phase: WS01 briefly reported an Unhealthy Fleet status while the Elastic Agent policy was being applied; this resolved automatically once policy application completed, and log collection resumed without further intervention.

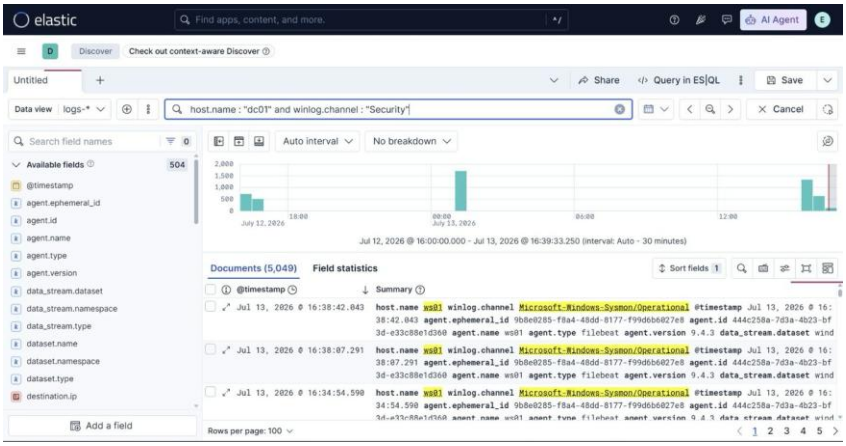


Fig. 16. A simple KQL query used to functionally test the pipeline against a known, benign event.

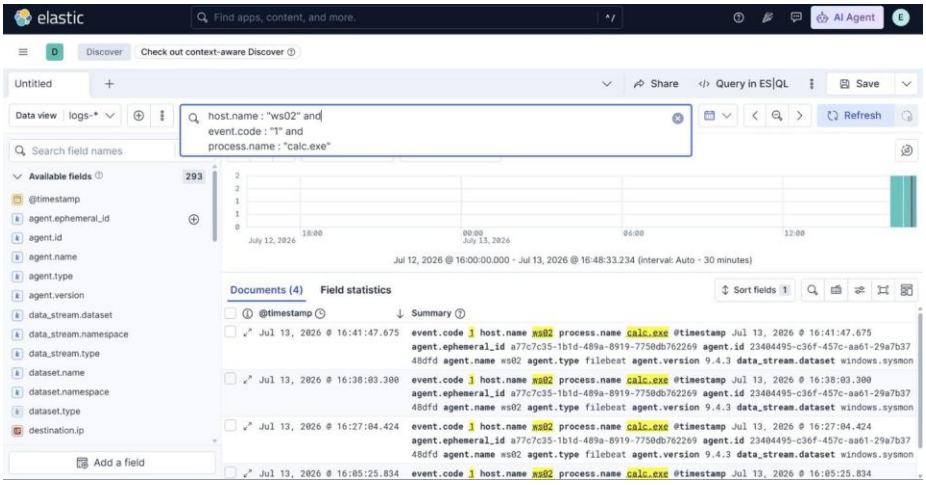


Fig. 17. Calculator launched on WS02 and the corresponding event captured in Kibana, confirming event-level traceability end-to-end.

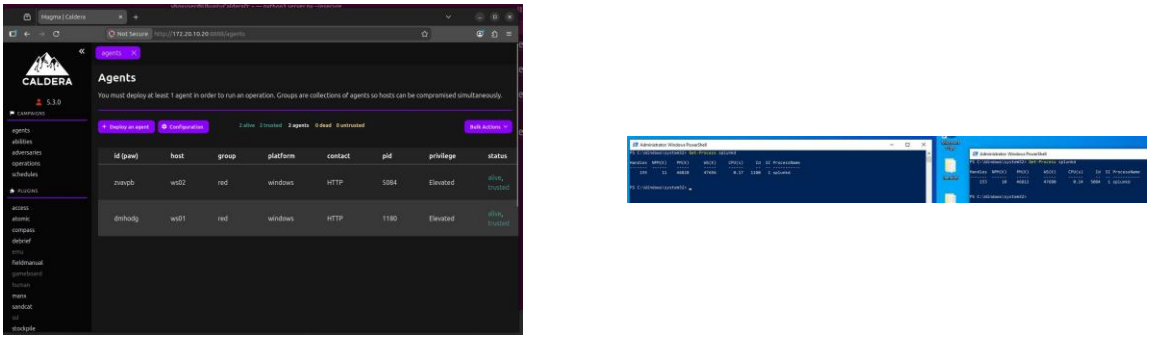


Fig. 18. CALDERA Sandcat agents deployed and confirmed alive and trusted on WS01 (dmhodg) and WS02 (zvavpb).

H. Adversary Profile Design

With the environment and telemetry pipeline validated, a custom adversary profile named APT29-FirstTry was built in MITRE CALDERA. During early scoping, the team sketched the intended attack lifecycle and the corresponding defense/detection workflow, shown in Fig. 19, before committing to specific CALDERA abilities.

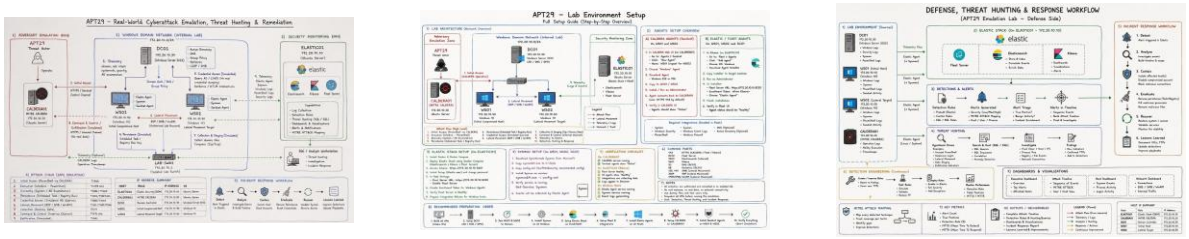


Fig. 19. Early concept diagrams produced while scoping the adversary profile: the intended attack lifecycle, the overall lab/detection architecture, and the corresponding defense and threat-hunting workflow.

The initial profile consisted of nine sequenced abilities across four phases:

TABLE III
Initial APT29-FirstTry Adversary Profile (9 Abilities)

Phase	Ability	ATT&CK ID
1 — Execution	PowerShell -EncodedCommand parameter variations	T1059.001
1 — Execution	Execute base64-encoded PowerShell from the Windows Registry	T1027
2 — Discovery	Hostname Discovery	T1082
2 — Discovery	System Discovery via whoami	T1033
2 — Discovery	System Network Configuration Discovery	T1016
2 — Discovery	Local Account Discovery	T1087.001
3 — Persistence	Executable shortcut link in the user Startup folder	T1547.001
4 — Lateral Movement / Persistence	Enable Windows Remote Management	T1021.006
4 — Lateral Movement / Persistence	Persistence via WMI Event Subscription	T1546.003

The Golden Ticket forgery technique (T1558.001) was deliberately excluded from the main chain and scheduled as a separate, dedicated operation, since it requires domain-level credential access (the krbtgt hash) that is only obtainable after the credential-access phase has already succeeded. As documented in Section IX, this follow-up operation remained a planned rather than executed step by the end of the project timeline.

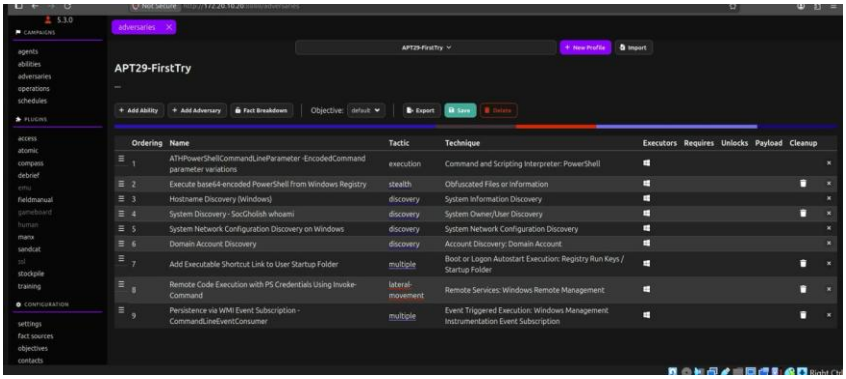


Fig. 20. Building the initial CALDERA adversary profile and sequencing abilities across phases.

1) Attack-Chain Refinement

During the attack-execution phase, running the full APT29-FirstTry profile produced consistent failures across multiple abilities due to syntax mismatches and environment incompatibilities specific to the lab setup. Rather than troubleshooting the profile as a single unit, the team rebuilt and validated each ability individually before assembling a final custom profile, ensuring every technique was confirmed to execute correctly before inclusion.

The Initial Access technique was also revised mid-project. The original approach delivered a PDF file via the CALDERA C2 channel and manually simulated payload execution, but this did not produce a reliable parent–child process relationship detectable by the corresponding Elastic Security rule. It was replaced with a malicious HTA (HTML Application) file delivered to WS01 and executed via mshta.exe. This change was driven by three factors: HTA-based phishing is consistent with documented APT29 tradecraft; mshta.exe is a native Windows binary, making the technique a genuine Living-off-the-Land (LOLBin) approach aligned with the project's focus; and the technique introduces Defense Evasion coverage (T1218.005) that had previously been absent, complementing the existing Initial Access (T1566.001) and Execution (T1059.001) coverage.

A Collection-phase gap (T1119) was also identified and closed by adding a dedicated ability and detection rule for file enumeration and staging activity performed via PowerShell against the dummy corporate data planted on WS01. A dedicated Exfiltration-over-C2 technique (T1041) was added once the collection stage was in place, transmitting the staged data to the CALDERA C2 listener. The resulting, final attack chain is summarized in Table IV.

TABLE IV
Final Executed Attack Chain (Nine Techniques, Eight Phases)

#	Phase	Technique	ATT&CK ID
1	Initial Access	HTA file → mshta.exe → PowerShell	T1566.001 / T1218.005
2	Execution	Obfuscated / encoded PowerShell	T1059.001 / T1027
3	Discovery	whoami, net user, systeminfo, ipconfig	T1082 / T1069
4	Persistence	Registry Run key	T1547.001
4	Persistence	Scheduled task	T1053.005

#	Phase	Technique	ATT&CK ID
5	Credential Access	DCSync (krbtgt hash)	T1003.006
6	Lateral Movement	WinRM to WS02	T1021.006
7	Persistence (WS02)	WMI event subscription	T1546.003
8	Collection	Staging of dummy corporate data on WS01	T1119
9	Exfiltration	Transmission to the CALDERA C2 server	T1041

To validate the Initial Access detection capability end-to-end, a custom CALDERA ability named "Safe HTA Spearphishing Attachment Simulation" was developed. The ability delivers a benign HTA file (Invoice.hta) to the victim host via the CALDERA C2 channel and executes it using mshta.exe, which in turn spawns a PowerShell child process — replicating the exact process chain produced by a real spearphishing attachment opened by an end user.

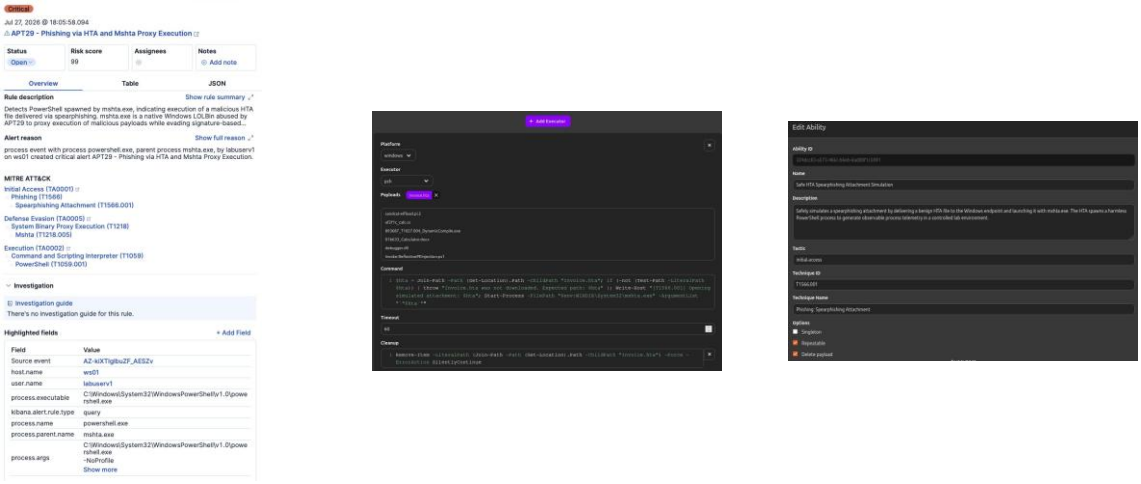


Fig. 21. Development of the custom "Safe HTA Spearphishing Attachment Simulation" CALDERA ability: payload/executor configuration, ability metadata, and live execution against WS01.

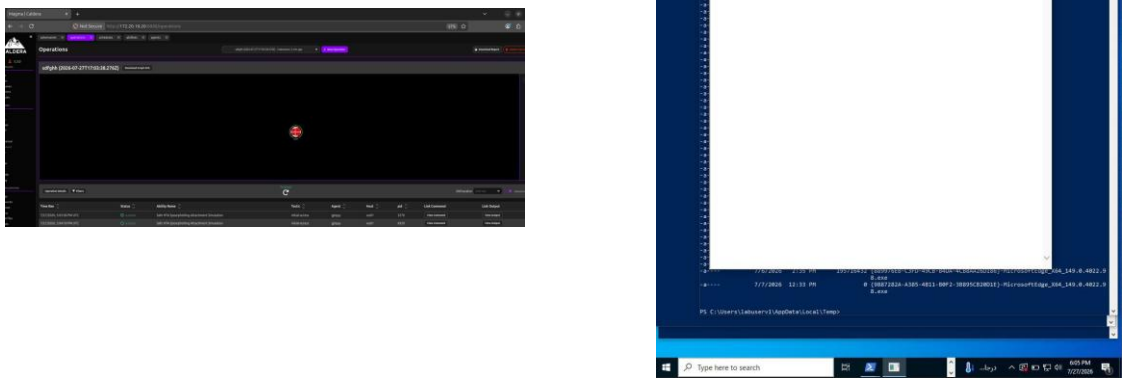


Fig. 22. The HTA payload observed opening on WS01 and spawning a PowerShell child process on screen, replicating a real spearphishing execution.

I. Detection Rule Engineering Methodology

Detection rules were engineered incrementally and technique-by-technique rather than all at once, following an explicit discipline requested by the project mentor: after each rule was authored, the corresponding CALDERA ability was executed immediately to confirm the rule fired correctly in Kibana before moving to the next rule. This caught several problems early that would otherwise have surfaced only during the final full-chain run.

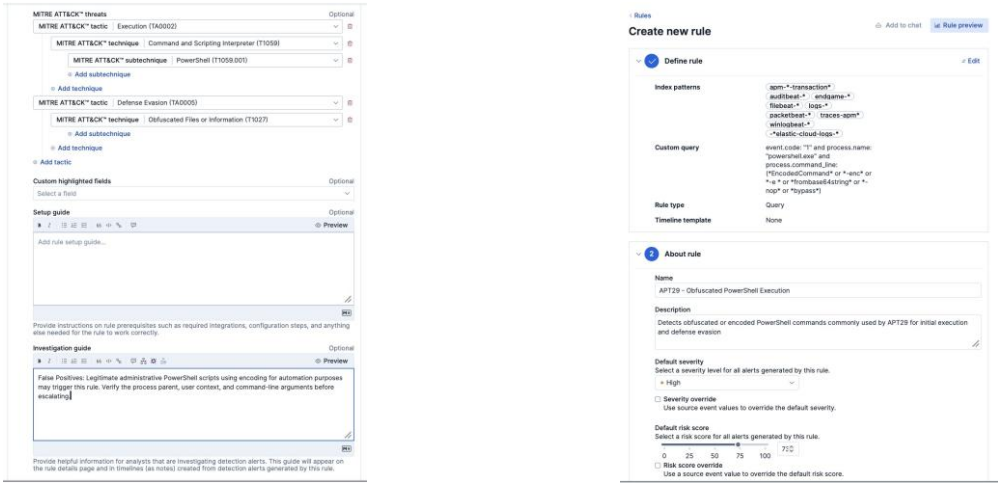


Fig. 23. Authoring the first detection rule (Obfuscated PowerShell Execution) in Elastic Security.

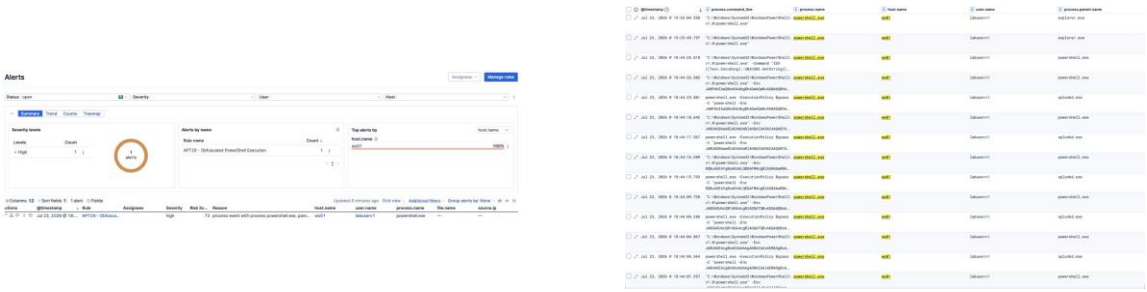


Fig. 24. Triggering the CALDERA ability associated with the Obfuscated PowerShell Execution rule to validate detection before proceeding to the next rule.

A critical, easy-to-miss discrepancy was uncovered during this process: PowerShell Script Block content is exposed in the Elastic data model as powershell.file.script_block_text, not winlog.event_data.ScriptBlockText as public documentation and initial assumptions suggested. Rules written against the documented field name would have silently failed to generate alerts. This reinforced a general validation discipline that was applied for the remainder of the project: never trust a rule until it has been fired against a known action and confirmed in the Alerts view.

Sixteen detection rules were ultimately engineered, spanning Process Behavior, Persistence and Identity, and Network and Collection categories, each configured with a one-minute execution interval and a five-minute look-back window using KQL and EQL. The complete rule catalogue — including the exact query,

MITRE ATT&CK mapping, known false-positive examples, and reference sources for every rule — is reproduced in Appendix B; a summarized version is presented in Section IV-D.

1) Iterative False-Positive Tuning

Detection rules were tuned against real false positives observed in the environment rather than written once and left static. The Registry Run Key Persistence rule, for example, initially fired against legitimate OneDrive and Windows update activity; it was tuned to exclude OneDriveSetup.exe, MsMpEng.exe, SppExtComObj.exe, msiexec.exe, TrustedInstaller.exe, and msedge.exe.

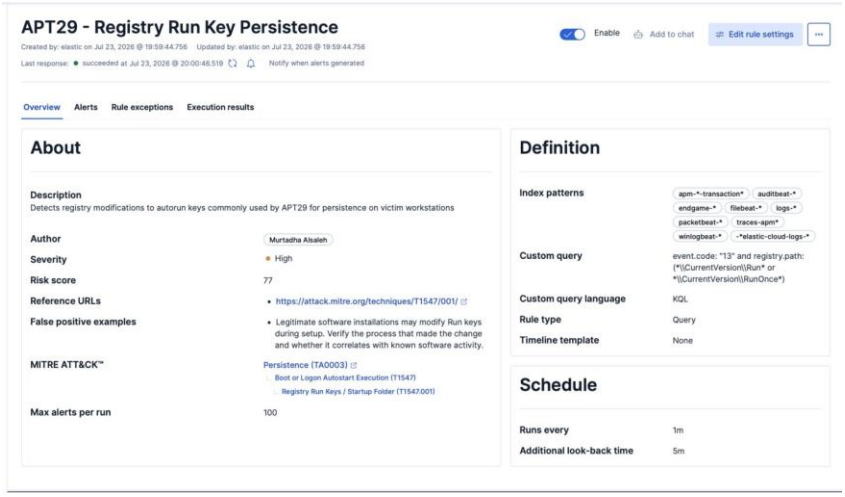


Fig. 25. Registry Run Key Persistence detection rule as originally authored.

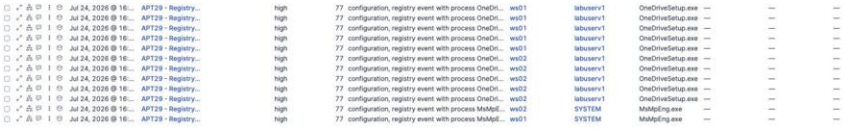


Fig. 26. A false-positive alert generated by a legitimate process against the Registry Run Key Persistence rule, motivating the exclusion list shown below.

```
event.code: "13" and
registry.path: (*\\CurrentVersion\\Run* or *\\CurrentVersion\\RunOnce*) and
not process.name: ("OneDriveSetup.exe" or "MsMpEng.exe" or
  "SppExtComObj.exe" or "msiexec.exe" or "TrustedInstaller.exe" or "msedge.exe")
```

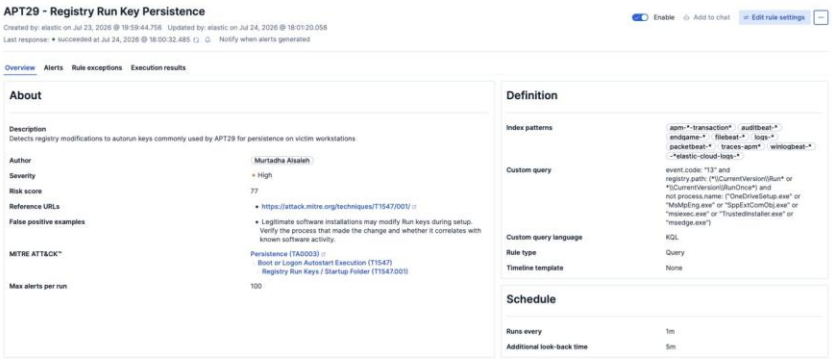


Fig. 27. The Registry Run Key Persistence rule after tuning, ready for re-validation against the next attack run.

Similarly, the Lateral Movement via WinRM rule initially produced a high volume of false positives generated by WS01's machine account (WS01$) authenticating to DC01 through routine domain operations such as Group Policy refresh, Kerberos ticket renewal, and DNS resolution — all of which generate Security Event ID 4624 (Logon Type 3). The rule was tuned to exclude computer accounts (identifiable by a trailing $) and anonymous logon events, so that it triggers only on human-account network logons originating from WS01. The File Collection via PowerShell rule likewise initially fired on benign path-handling commands (Get-Location, Test-Path, Join-Path) embedded in the HTA payload itself; it was tuned to require genuine enumeration or staging commands (Get-ChildItem, Copy-Item, Get-Content, Compress-Archive) instead.

J. Deception Technology — Honeypot Account

As an additional, non-signature detection layer, a decoy Active Directory account, svc-backup, was created under the IT OU on DC01 so that it would appear attractive to an adversary performing account discovery. The account has zero legitimate use, so any authentication attempt against it is treated as a high-confidence indicator of compromise. Directory Service auditing of failed and successful logon attempts was enabled for the account, and a dedicated Critical-severity detection rule was created to monitor Event IDs 4624, 4625, 4768, and 4769 for any reference to svc-backup.

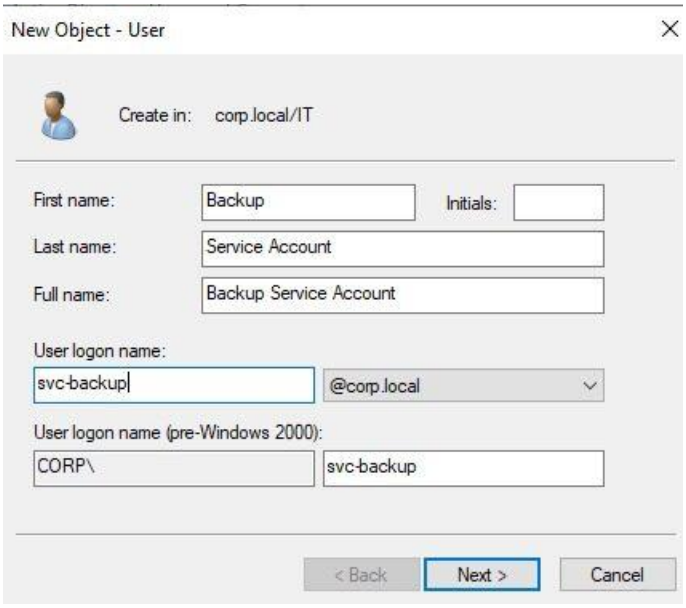


Fig. 28. Creation of the svc-backup honeypot account under the IT OU on DC01.

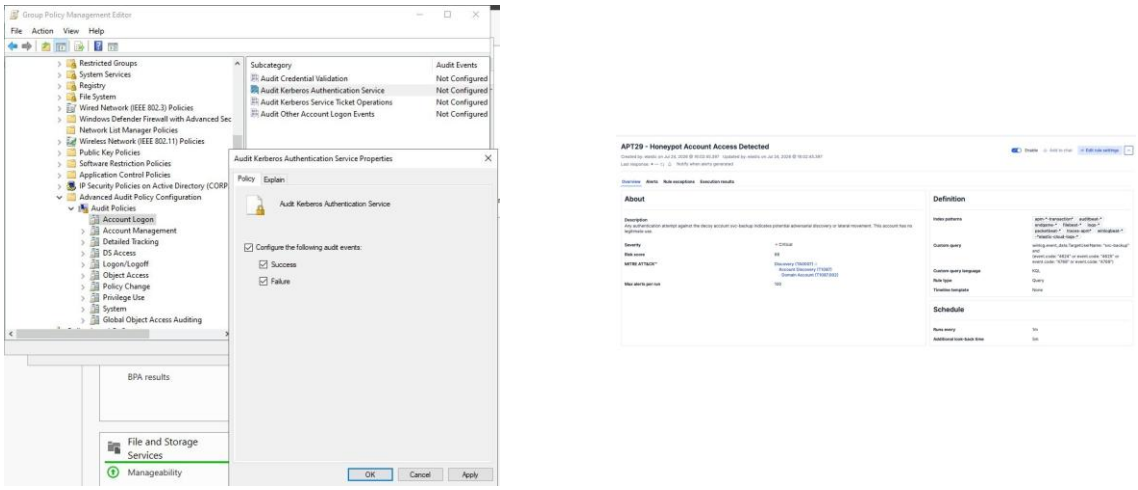


Fig. 29. (left) Directory Service audit policy enabled for logon events against the decoy account; (right) the Honeypot Account Access Detected rule configured in Elastic Security.



Fig. 30. Validation of the honeypot: a simulated authentication attempt against svc-backup (left) and the resulting Critical-severity alert triggered in Kibana (right).

K. Threat-Hunting Methodology

In addition to automated detection rules, six manual threat hunts were designed in Kibana Discover to independently correlate endpoint, identity, persistence, and network activity into a single attack timeline, complementing the alert-driven view with an analyst-driven one. The hunts and their underlying queries are presented together with their results in Section IV-G.

L. Incident-Response Methodology

The incident-response phase of the project was carried out as a structured exercise against the emulated intrusion, following the detection-and-triage, containment, eradication, recovery, and post-incident-review lifecycle defined in NIST SP 800-61 Rev. 2 [5]. Every containment, eradication, and recovery action executed against the lab environment was captured and formalized into a standalone playbook (Appendix A) so that the same steps can be replayed consistently by a SOC analyst outside the context of this specific exercise. Execution of the plan and its results are presented in Section IV-H.

IV. RESULTS

A. Environment Build-Out

By the end of Week 3, the lab environment described in Section III was fully built and validated:

- DC01, WS01, and WS02 were joined to the corp.local domain with static IP addressing (172.20.10.30/.40/.50) and DNS pointed at DC01.
- WS01 (Employee OU) and WS02 (IT OU) were correctly placed with their intended privilege levels.
- CALDERA01 was operational on Ubuntu Desktop (172.20.10.20), reachable over the lab LAN, with agents deployed to WS01, WS02, and later DC01.
- Elastic Agent was installed and reporting Healthy/Connected status on DC01, WS01, and WS02, with Fleet Server operational on ELASTIC01 (172.20.10.10).

B. Logging Pipeline Validation

Sysmon and PowerShell Script Block/Transcription logging were confirmed operational on all three Windows hosts, with events visible in Kibana on a per-host basis (Section III-F). A functional test — executing a simple KQL query cross-referenced against a known, benign action — confirmed that events generated on an endpoint were reliably searchable in the SIEM within the expected timeframe (Section III-G). This validated pipeline was the prerequisite that made every subsequent detection result in this section meaningful.

C. Detection Engineering Results

Sixteen MITRE ATT&CK-mapped detection rules were engineered in Elastic Security across three categories: Process Behavior, Persistence and Identity, and Network and Collection. Each rule runs on a one-minute execution interval with a five-minute look-back window, using KQL or EQL. Table V summarizes the rule catalogue; the full query, false-positive notes, and reference sources for every rule are reproduced in Appendix B.

TABLE V
Detection Rule Catalogue (Sixteen Rules)

Rule Name	ATT&CK Technique	Primary Telemetry
APT29 - Phishing via HTA and Mshta Proxy Execution	T1566.001 / T1218.005 / T1059.001	Sysmon Event 1 (process creation)
APT29 - Obfuscated PowerShell Execution	T1059.001 / T1027	Sysmon Event 1
APT29 - PowerShell Script Block Suspicious Content	T1059.001 / T1546.003 / T1021.006	Event 4104 (Script Block Logging)
APT29 - Process Execution from Suspicious Directory	T1036 / T1059	Sysmon Event 1
APT29 - System and AD Discovery	T1082 / T1069 / T1007	Sysmon Event 1
APT29 - Registry Run Key Persistence	T1547.001 / T1003	Sysmon Event 13
APT29 - Scheduled Task Persistence	T1053.005	Sysmon Event 1 / Security 4698
APT29 - WMI Event Subscription Persistence	T1546.003	Sysmon Events 19, 20, 21
APT29 - Honeypot Account Access Detected	T1087 (deception)	Security 4624 / 4625 / 4768 / 4769
APT29 - Mimikatz Process Execution	T1003.006	Sysmon Event 1

Rule Name	ATT&CK Technique	Primary Telemetry
APT29 - DCSync Credential Dumping	T1003.006	Security Event 4662
APT29 - Explicit Credential Object Creation	T1021.006 / T1550.002	Event 4104
APT29 - Lateral Movement via WinRM	T1021.006	Security Event 4624 (Logon Type 3)
APT29 - File Collection via PowerShell	T1119	Sysmon Event 1
APT29 - PowerShell Suspicious Network Outbound	T1071.001 / T1041	Event 4104
APT29 - Data Exfiltration Over C2 Channel	T1041	Sysmon Event 3 (network connection)



Fig. 31. Rule authoring in Elastic Security: (left) Lateral Movement via WinRM; (right) WMI Event Subscription Persistence.

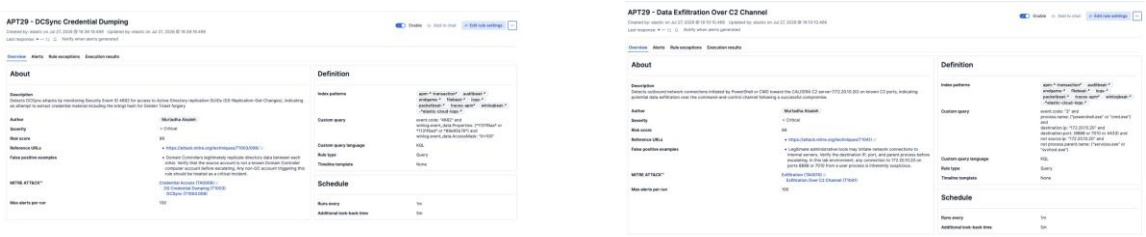


Fig. 32. Rule authoring in Elastic Security: (left) DCSync Credential Dumping; (right) Data Exfiltration Over C2 Channel.

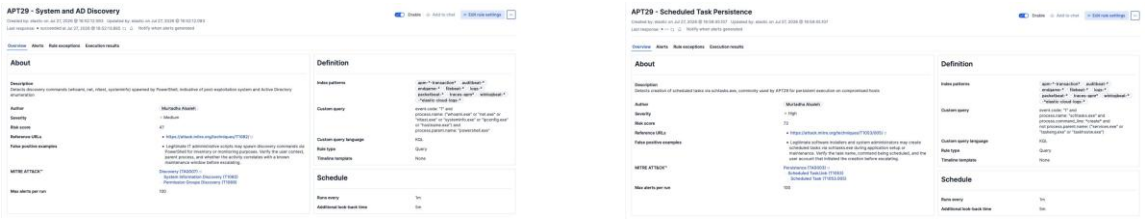


Fig. 33. Rule authoring in Elastic Security: (left) System and AD Discovery; (right) Scheduled Task Persistence.

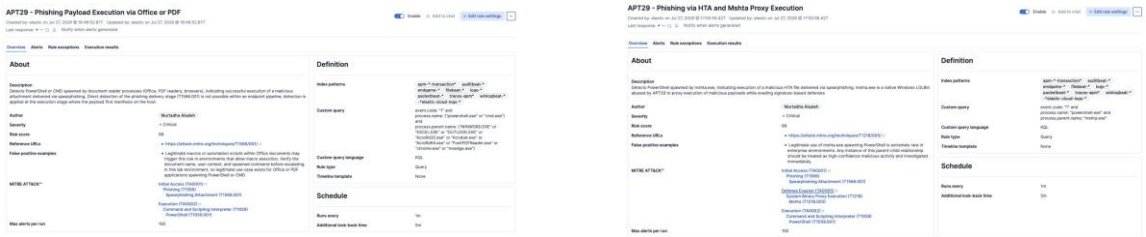


Fig. 34. (left) The original Phishing Payload Execution via Office or PDF rule, later retired; (right) its replacement, APT29 - Phishing via HTA and Mshta Proxy Execution, following the Initial Access technique revision described in Section III-H1.

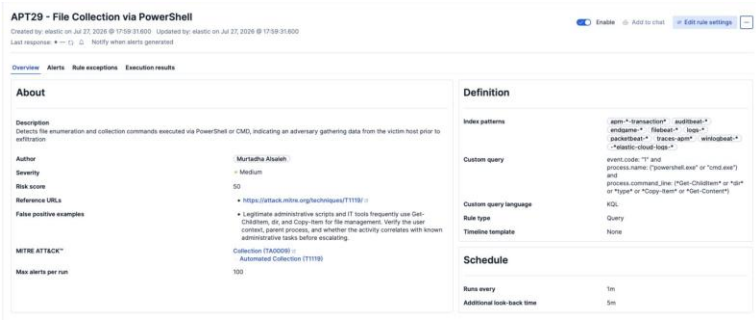


Fig. 35. APT29 - File Collection via PowerShell rule, added to close the Collection-phase (T1119) coverage gap.

D. Adversary Emulation Execution

The CALDERA agent was extended to DC01 during Week 4 so that credential-access techniques could be emulated directly against the domain controller.

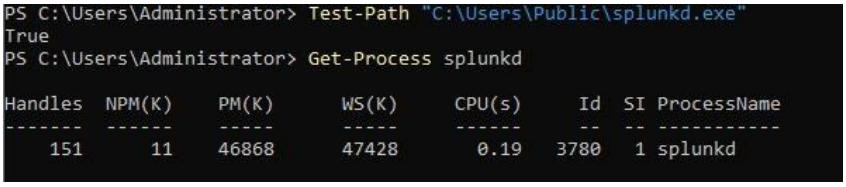


Fig. 36. CALDERA agent installation and enrollment on DC01, enabling credential-access emulation.

Dummy corporate data was staged on WS01 to give the Collection and Exfiltration phases a realistic target.

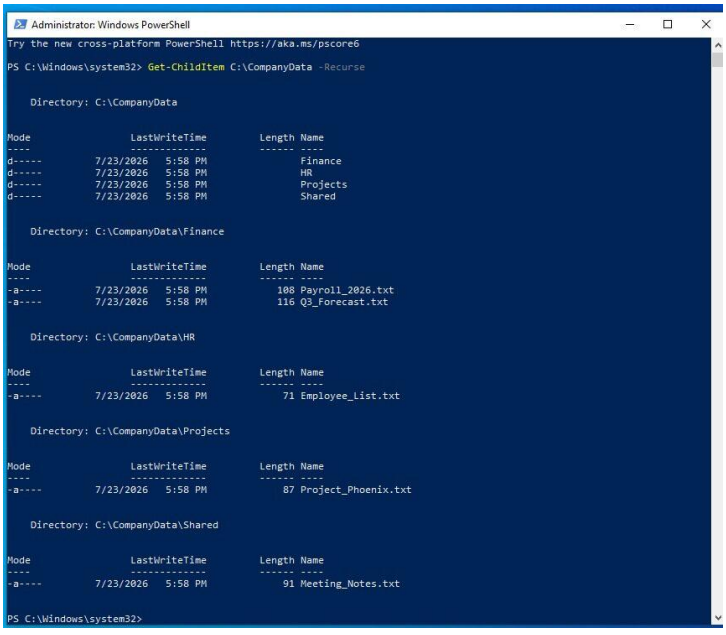


Fig. 37. Dummy corporate data (CompanyData) staged on WS01 to support the Collection and Exfiltration phases.

Running the assembled attack chain against the environment was not a single clean pass. During early full-chain runs, CALDERA reported a large number of ability-execution failures, predominantly syntax

mismatches between the ability's expected shell and the target host's actual command interpreter. These were resolved iteratively, one ability at a time, consistent with the validate-before-attack discipline described in Section VI-D.



Fig. 38. CALDERA operation log showing ability-execution errors encountered during an early full-chain run, subsequently resolved through per-ability syntax correction.

1) Final Validated Operation Run

The attack chain was ultimately executed and accepted as the final, reportable CALDERA operation, spanning three agents — uamwqj on WS01, hahuqk on WS02, and lajkar on DC01 — across all eleven techniques of the chain defined in Table IV. Table VI reproduces the operation log for this final run, in chronological order, exactly as recorded by CALDERA.

TABLE VI
Final CALDERA Operation Log — Complete Eleven-Technique Attack Chain

Time (UTC)	Ability	Tactic	Agent	Host	Status
7:46:50 PM	Safe HTA Spearphishing Simulation	initial-access	uamwqj	WS01	Success
7:48:32 PM	PowerShell Command Execution	execution	uamwqj	WS01	Success
7:48:47 PM	System Information Discovery	discovery	uamwqj	WS01	Success
7:49:47 PM	Elevated group enumeration (net group /domain)	discovery	uamwqj	WS01	Success
7:50:42 PM	Reg Key Run	persistence	uamwqj	WS01	Success
7:51:17 PM	Scheduled task (local)	persistence	uamwqj	WS01	Success
7:52:22 PM	Recon information for export with PowerShell	collection	uamwqj	WS01	Success — see note below
7:53:12 PM	C2 Data Exfiltration	exfiltration	uamwqj	WS01	Success — see note below
7:56:31 PM	DCSync (Active Directory)	credential-access	lajkar	DC01	Success
8:00:29 PM	Persistence via WMI Event Subscription (CommandLineEventConsumer)	persistence	hahuqk	WS02	Success
8:02:29 PM	Honeypot Account Access — svc-backup	discovery	uamwqj	WS01	Success

Two abilities — Recon Information for Export with PowerShell and C2 Data Exfiltration — were reported by the CALDERA operation table with a "failed" status field. This was confirmed to be a CALDERA-side reporting artifact rather than an actual execution failure: opening View Output for each ability showed the underlying PowerShell commands completing successfully, and the staged CompanyData contents were

independently confirmed to have arrived at the CALDERA C2 listener. Both abilities are therefore counted as successful in the eleven-technique coverage figure reported in Section IV-E, consistent with the output-level evidence rather than the summary status field alone.

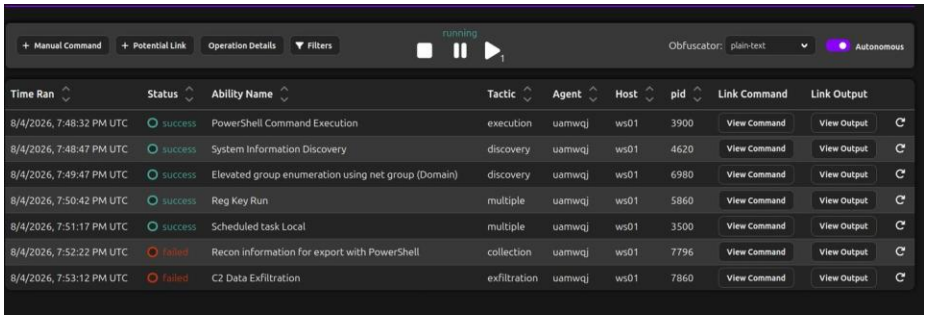


Fig. 39. Final operation log for the WS01 agent (uamwqj), showing seven sequential abilities. The Recon Information and C2 Data Exfiltration entries display a "failed" status field despite completing successfully, as confirmed via each ability's View Output.



Fig. 40. (left) Initial Access ability — Safe HTA Spearphishing Simulation — confirmed successful on WS01; (right) DCSync (Active Directory) credential-access ability confirmed successful against DC01 (agent lajkar).



Fig. 41. (left) WMI Event Subscription persistence confirmed successful on WS02 (agent hahuqk); (right) the honeypot-account discovery ability confirmed successful against svc-backup on WS01.

2) Lateral-Movement Rule Re-Validation

The Lateral Movement via WinRM rule was re-validated after tuning (Section III-I1) and confirmed to trigger only on genuine human-account network logons from WS01, with the machine-account noise eliminated.

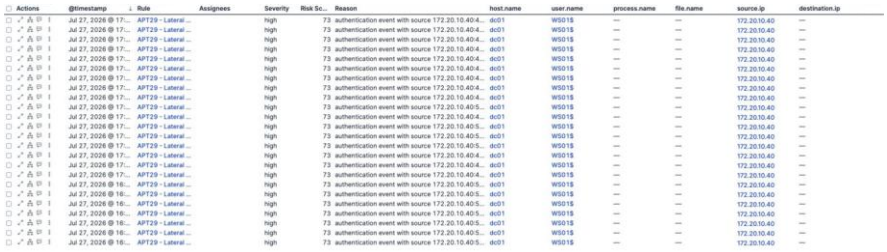


Fig. 42. Lateral Movement via WinRM rule after tuning to exclude computer-account (WS01$) and anonymous logon noise, re-validated against a live attack run.

E. Final Attack-Chain Run and Alert Summary

The optimized final attack-chain execution produced 39 alerts with technique coverage across all eleven emulated techniques (11/11, 100%) within the project's selected scope, drawing on the credential-access, persistence, lateral-movement, collection, and exfiltration phases described in Section III-H. Three detection rules required a query fix after this validation run (System/AD Discovery gained a cmd.exe parent-process condition, Scheduled Task Persistence gained a Register-ScheduledTask PowerShell cmdlet

condition, and the exfiltration rule gained port 9999 to its destination-port list), and five distinct false-positive sources were identified and excluded across the rule set, consistent with the tuning discipline described in Section III-I1.

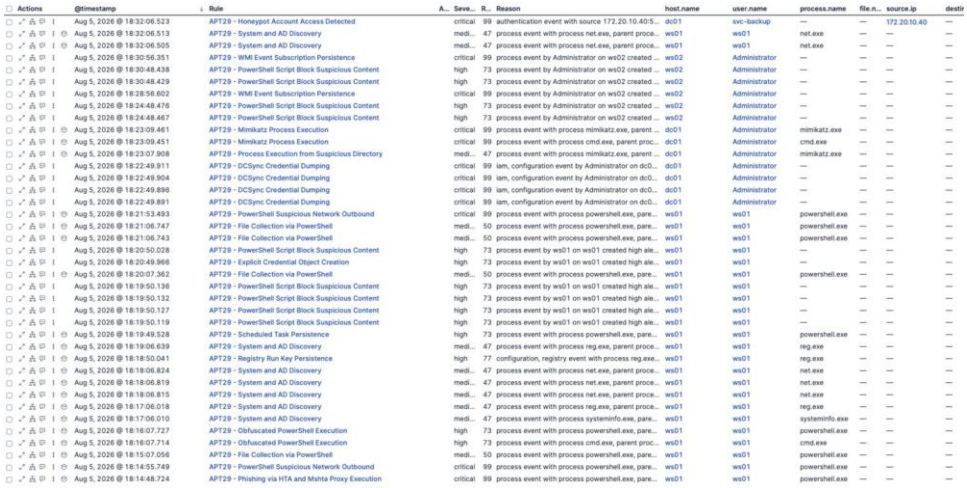


Fig. 43. Elastic Security alert feed from the final attack-chain execution, showing critical- and high-severity alerts spanning the honeypot, credential-dumping, persistence, and PowerShell-execution rule categories with full host and user attribution.

Reviewed alongside the CALDERA agent's own alert timeline, the honeypot-account rule, the DCSync/Mimikatz rules, and the PowerShell script-block content rule together accounted for the majority of Critical-severity alerts, confirming that the layered, defense-in-depth rule design described in Section VI-F produced redundant detection coverage at more than one point in the attack chain.

F. Threat Hunting Results

Six targeted threat hunts were executed in Kibana Discover after the final attack run, independently correlating endpoint, identity, persistence, and network activity across the environment into a single attack timeline, without relying on the automated alert feed. Table VII summarizes the hunts.

TABLE VII
Threat-Hunting Queries and Targets

Hunt	Investigation Focus	Query / Indicator	Target Host / Log
H1	CALDERA child processes	event.code:"1" and process.parent.name:"splunkd.exe"	Sysmon Event 1
H2	Registry persistence	event.code:"13" and registry.path:*CurrentVersion\Run	Sysmon Event 13
H3	Scheduled tasks	event.code:"1" and (process.name:"schtasks.exe" or process.command_line:*Register-ScheduledTask)	Security 4698 / Sysmon 1
H4	WMI persistence	event.code:"19" or event.code:"20" or event.code:"21"	Sysmon Events 19, 20, 21
H5	Command-and-control communications	event.code:"3" and destination.ip:"172.20.10.20"	Sysmon Event 3
H6	DCSync replication	event.code:"4662" and winlog.event_data.Properties:(*1131f6aa* or *1131f6ad*)	Security Event 4662 (DC01)



Fig. 44. Hunt H1: CALDERA Sandcat child processes (disguised as splunkd.exe) identified across the attack timeline in Kibana Discover.

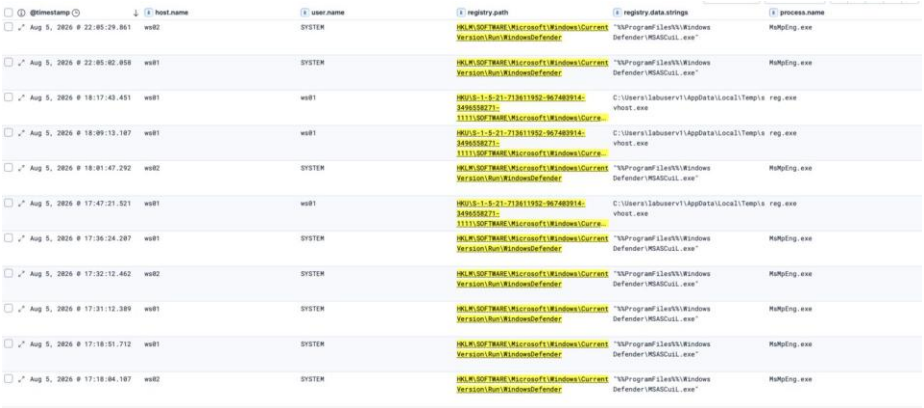


Fig. 45. Hunt H2: Registry Run-key creation and modification events correlated across the persistence timeline.



Fig. 46. Hunt H3: Scheduled-task creation events (schtasks.exe / Register-ScheduledTask) identified on WS01.

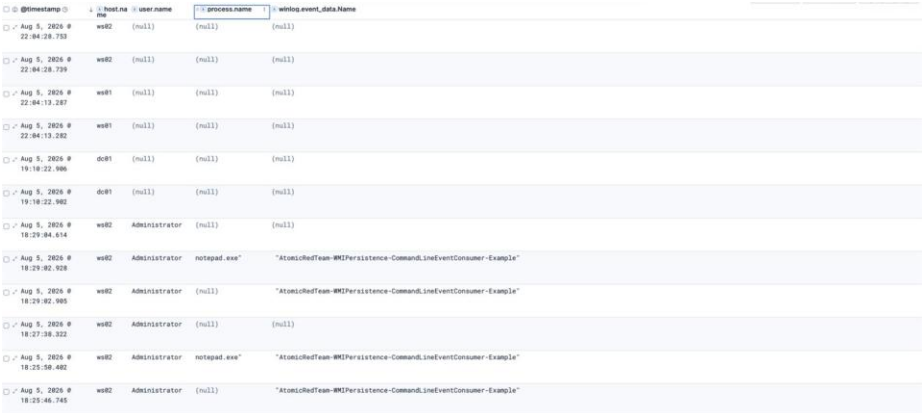
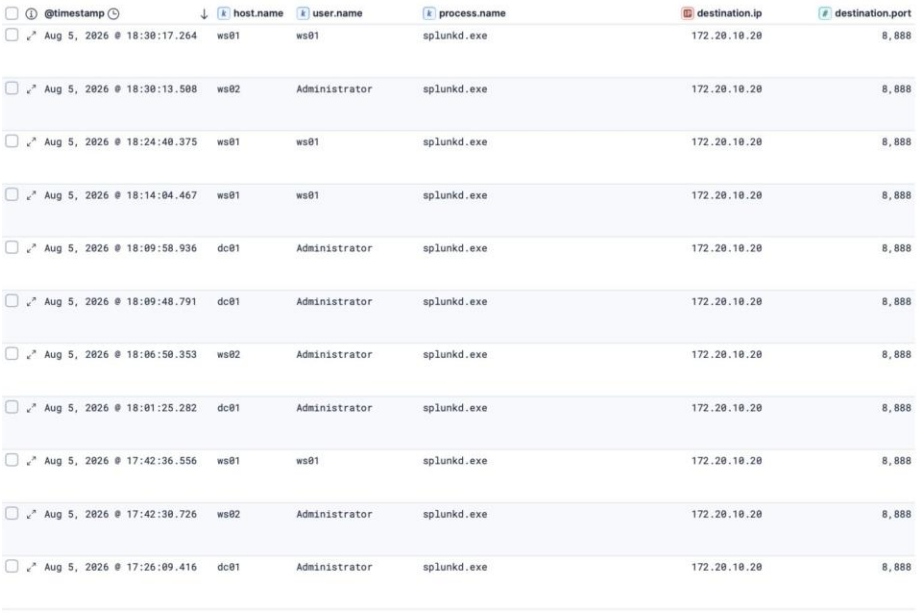
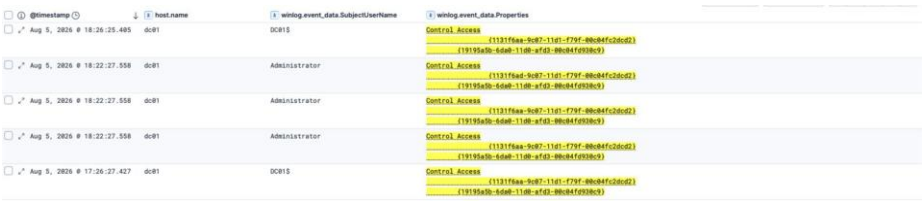


Fig. 47. Hunt H4: WMI Event Filter, Consumer, and FilterToConsumerBinding registration events (Sysmon 19/20/21) correlated on WS02.



@timestamp	host.name	user.name	process.name	destination.ip	destination.port
Aug 5, 2026 @ 18:30:17.264	ws01	ws01	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:30:13.508	ws02	Administrator	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:24:40.375	ws01	ws01	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:14:04.467	ws01	ws01	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:09:58.936	dc01	Administrator	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:09:48.791	dc01	Administrator	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:06:50.353	ws02	Administrator	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 18:01:25.282	dc01	Administrator	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 17:42:36.556	ws01	ws01	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 17:42:30.726	ws02	Administrator	splunkd.exe	172.20.10.20	8,888
Aug 5, 2026 @ 17:26:09.416	dc01	Administrator	splunkd.exe	172.20.10.20	8,888

Fig. 48. Hunt H5: Outbound network connections toward the CALDERA C2 server (172.20.10.20), tracing the command-and-control and exfiltration channel.



@timestamp	host.name	winlog.event_data.SubjectUserName	winlog.event_data.Properties
Aug 5, 2026 @ 18:26:25.405	dc01	DC01$	Control Access {1131f6aa-9c07-11d1-f79f-00c04fc2dcd2} {19195a5b-6da0-11d0-afd3-00c04fd930c9}
Aug 5, 2026 @ 18:22:27.558	dc01	Administrator	Control Access {1131f6ad-9c07-11d1-f79f-00c04fc2dcd2} {19195a5b-6da0-11d0-afd3-00c04fd930c9}
Aug 5, 2026 @ 18:22:27.558	dc01	Administrator	Control Access {1131f6aa-9c07-11d1-f79f-00c04fc2dcd2} {19195a5b-6da0-11d0-afd3-00c04fd930c9}
Aug 5, 2026 @ 18:22:27.558	dc01	Administrator	Control Access {1131f6aa-9c07-11d1-f79f-00c04fc2dcd2} {19195a5b-6da0-11d0-afd3-00c04fd930c9}
Aug 5, 2026 @ 17:26:27.427	dc01	DC01$	Control Access {1131f6aa-9c07-11d1-f79f-00c04fc2dcd2} {19195a5b-6da0-11d0-afd3-00c04fd930c9}

Fig. 49. Hunt H6: Security Event 4662 access to Active Directory replication GUIDs on DC01, confirming the DCSync credential-access step.

Two complete attack-chain runs were identified in the environment from Hunt H1 alone, confirming that the CALDERA agent's process footprint (splunkd.exe, staged in C:\Users\Public) is a reliable pivot point for reconstructing the full intrusion timeline independent of the rule-based alert feed. Taken together, the six hunts connected isolated events into a clear sequence from initial execution and persistence through to command-and-control and DCSync activity, closing the loop between automated detection and analyst-driven investigation.

G. Incident Response Execution

A full incident-response cycle was executed against the simulated intrusion, following the detection-and-triage, containment, eradication, recovery, and post-incident phases of NIST SP 800-61 Rev. 2. The complete, reusable playbook produced from this exercise is reproduced in Appendix A; this subsection presents the execution evidence.

1) Detection and Triage

Alerts were reviewed in Kibana under Security → Alerts, sorted by timestamp to reconstruct the attack timeline, and triaged by severity. During triage, the compromised account was confirmed to be ws01, not labuserv1 as originally assumed from the naming convention used elsewhere in the project documentation — a discrepancy that was corrected before containment began.



Fig. 50. Confirmation that the compromised Active Directory identity was "ws01", correcting an earlier assumption that the account was "labuserv1".

2) Containment

WS01, the primary compromised host, was isolated from the network by disabling its Ethernet adapter. The CALDERA agent process (staged as splunkd.exe) was stopped, and the compromised account was disabled in Active Directory.

```
# Isolate WS01
Disable-NetAdapter -Name "Ethernet" -Confirm:$false

# Stop CALDERA agent
Get-Process splunkd -ErrorAction SilentlyContinue | Stop-Process -Force

# Disable the compromised account
Disable-ADAccount -Identity "ws01"
```

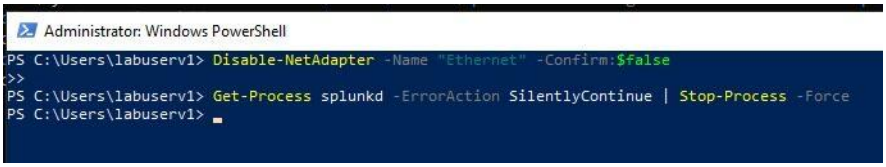
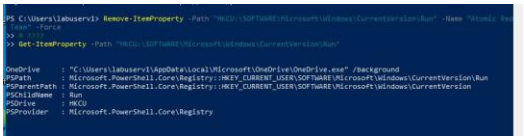


Fig. 51. WS01 isolated from the network and the CALDERA agent process terminated, executed on the compromised host.

3) Eradication

Adversary artifacts were removed from every affected host in a fixed order, with each removal verified before proceeding to the next: the Registry Run key on WS01, the scheduled task on WS01, the WMI Event Filter/Consumer/Binding chain on WS02, the Mimikatz toolkit on DC01, and the CALDERA agent binary on all three hosts.



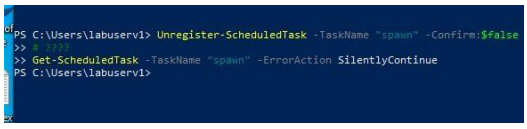


Fig. 52. Eradication commands executed and verified: the Registry Run-key persistence mechanism on WS01 (left) and the scheduled task on WS01 (right).

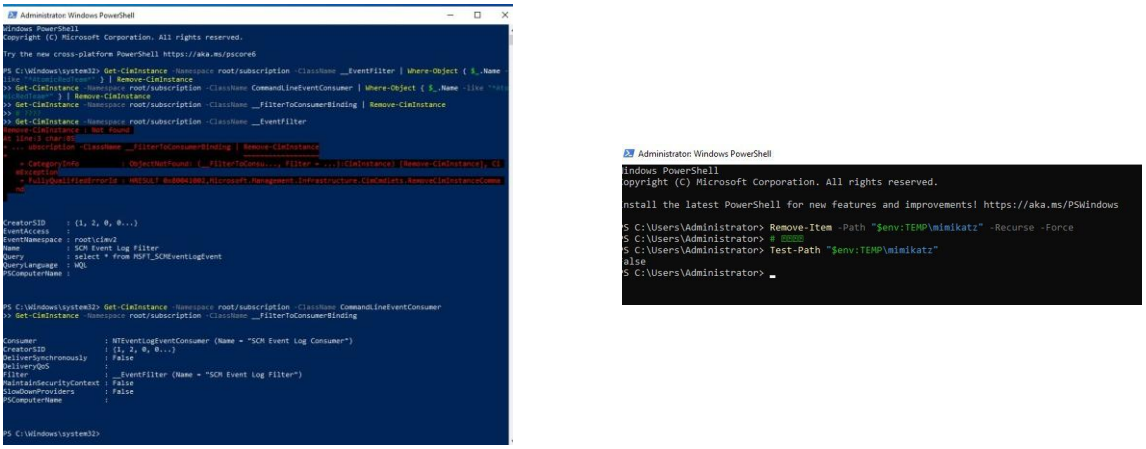


Fig. 53. Eradication commands executed and verified: the WMI Event Filter/Consumer/Binding chain on WS02 (left) and the Mimikatz toolkit on DC01 (right).

WMI persistence artifacts — the AtomicRedTeam __EventFilter, CommandLineEventConsumer, and their __FilterToConsumerBinding — were removed from WS02 via Remove-CimInstance, in the required order: binding, then consumer, then filter. Post-removal verification confirmed that only the default Windows SCM Event Log filter, consumer, and binding remained, with no trace of the adversary-planted subscription.



Fig. 54. Per-host eradication verification of the CALDERA agent binary: WS01 (left), WS02 (center), and DC01 (right), all confirmed clean of adversary artifacts.

4) Recovery

The compromised account's password was reset with a mandatory change at next logon, the account was re-enabled, and WS01's network adapter was re-enabled to restore normal operation. In this lab, the DCSync attack extracted the krbtgt hash but no Golden Ticket was forged; the double krbtgt password-reset procedure (with a minimum 72-hour interval between resets) is documented in the playbook for completeness and treated as a recovery step to rehearse rather than one that was required by an actual forged ticket.

```
Set-ADAccountPassword -Identity "ws01" -Reset `
  -NewPassword (ConvertTo-SecureString "NewSecure2026!" -AsPlainText -Force)
Set-ADUser ws01 -ChangePasswordAtLogon $true
Enable-ADAccount -Identity "ws01"
Enable-NetAdapter -Name "Ethernet" -Confirm:$false
```



Fig. 55. Recovery commands executed and verified: WS01's network adapter re-enabled (left) and the compromised account's password reset with the account re-enabled (right), closing the incident-response cycle.

H. Security Dashboards

A dedicated Kibana dashboard, "APT29 LOTL Adversary Emulation — Security Overview," was built with four panels covering alert volume by rule, alert distribution by host, alert severity breakdown, and a

MITRE ATT&CK technique-coverage view, giving a leadership-level summary of detection performance without requiring direct access to Discover.

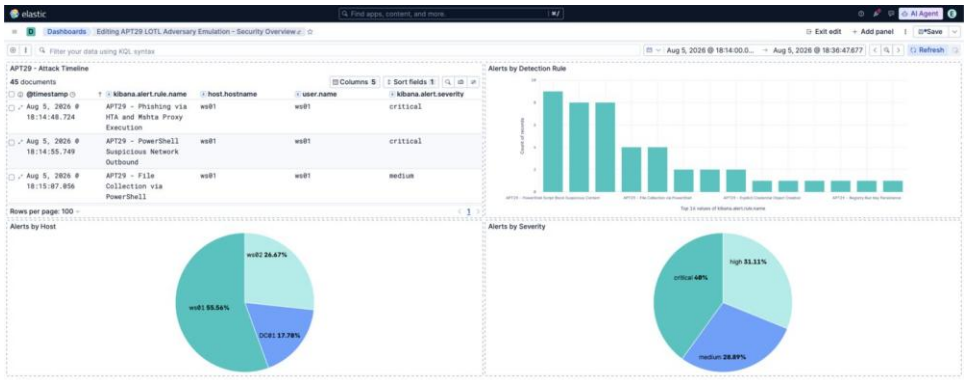


Fig. 56. Kibana security-overview dashboard summarizing alert volume, severity distribution, and technique coverage from the final attack-chain execution.

I. Evaluation Metrics

Five SOC performance metrics were evaluated to summarize detection-engineering performance quantitatively. Mean Time to Detect (MTTD) and MITRE ATT&CK technique coverage were measured directly from the final attack-chain run and the Elastic alert timestamps. Mean Time to Contain (MTTC), Mean Time to Remediate (MTTR), and the false-positive rate (FPR) are procedure-based estimates derived from the documented containment/eradication/recovery timings and the false-positive counts observed during rule tuning, rather than measurements from a live, unscripted incident; this distinction is stated explicitly in Table VIII rather than presented as a uniform level of evidence.

TABLE VIII
SOC Evaluation Metrics

Metric	Result	Evidence Status
Mean Time to Detect (MTTD)	18.4 seconds	Measured
Mean Time to Contain (MTTC)	~2.1 minutes	Procedure-based estimate
Mean Time to Remediate (MTTR)	~5.6 minutes	Procedure-based estimate
MITRE ATT&CK Technique Coverage	11 / 11 = 100%	Measured, within selected scope
False-Positive Rate (FPR)	~8.5%	Preliminary estimate

Overall, detection engineering performed strongly within the scope of the project: every emulated technique produced a corresponding, validated alert, and the measured detection speed (under twenty seconds on average) reflects the one-minute rule-execution interval and the high-fidelity telemetry sources — particularly PowerShell Script Block Logging — that the rules were built against. The estimated containment, remediation, and false-positive figures should be read as directional rather than statistically rigorous, and are revisited as a limitation in Section IX.

Producing these results, however, required manually correlating evidence across three separate tools — the CALDERA operation log, the Elastic Security alert feed, and the incident-response playbook execution —

since no single system in the environment tracked emulation, detection, and response outcomes together. This observation motivated a small prototype extension, Burhan, introduced in Section V.

V. BURHAN: A PROTOTYPE FOR UNIFIED SOC AND IR VALIDATION

A. Motivation

As noted at the close of Section IV, producing the evaluation metrics reported in this project required manually correlating evidence across three separate systems: the CALDERA operation log (Table VI), the Elastic Security alert feed and dashboards (Sections IV-C through IV-H), and the incident-response actions executed against the playbook in Appendix A. No single system in the environment tracked which techniques were emulated, which were detected, and how the response performed, as one connected record. After the original project concluded, this observation was explored further through a small proof-of-concept, Burhan, built specifically to examine whether emulation, detection, and response evidence could be brought together into a single validation and measurement layer.

Burhan was developed after, and separately from, the experimental work described in Sections III and IV. It was not used to design the attack chain, engineer the detection rules, or execute the incident-response cycle reported earlier in this document; it is introduced here strictly as a downstream extension motivated by that experience.

B. Extended Conceptual Lifecycle

The project as implemented and reported in Sections III and IV followed a four-stage workflow: an adversary profile was emulated against the environment, the resulting activity was detected by the Elastic Security rule set, the same evidence was independently re-examined through manual threat hunting, and a structured incident-response cycle was carried out against the confirmed intrusion.

Emulate → Detect → Threat Hunt → Respond

Burhan proposes a conceptual fifth stage, Validate, appended to this lifecycle purely as a forward-looking extension rather than a stage that was implemented or executed during the original experiment:

Emulate → Detect → Threat Hunt → Respond → ***Validate (proposed extension)***

In this conceptual extension, Validate refers to the automated, cross-tool scoring of how completely a given emulation was detected and how effectively it was responded to — the same questions this report answers manually in Sections IV and the Appendices, reframed as something a supporting tool could compute directly from the underlying evidence.

C. Prototype Demonstration

Burhan is a lightweight SOC-and-incident-response validation dashboard. Conceptually, it accepts three categories of evidence — adversary-emulation records (technique, host, timestamp, outcome), SIEM detection records (alert, rule, severity, timestamp), and incident-response records (containment/eradication/recovery actions and timing) — and computes a composite validation score from them, alongside per-technique MITRE ATT&CK coverage.

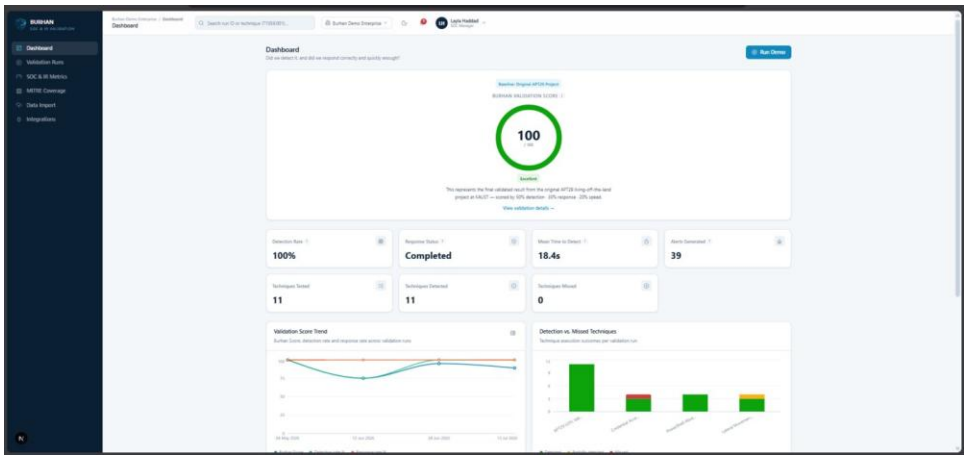


Fig. 57. Burhan dashboard, configured with this project's own results as its historical baseline. The 100/100 validation score, 100% detection rate, 39 alerts generated, 18.4-second mean time to detect, and 11/11 techniques detected reproduce the figures already reported in Table VI and Table VIII of this report; they are not independently recalculated values.

The dashboard's composite "Validation Score" is Burhan's own scoring construct — a weighted combination of detection rate (50%), response outcome (30%), and detection speed (20%) — and is shown here purely to demonstrate how such a composite metric could be derived from this project's own evidence. It is not an additional experimental metric of the original APT29 emulation and does not replace or revise the evaluation metrics reported in Table VIII.

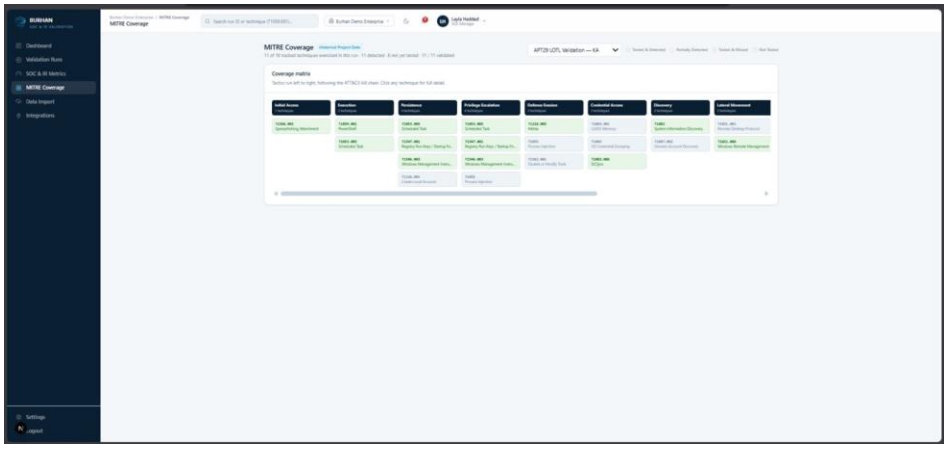


Fig. 58. Burhan's MITRE ATT&CK coverage matrix, explicitly labeled "Historical Project Data" in the interface, laying the eleven techniques executed and detected in this project (Table IV, Table VI) against the broader ATT&CK tactic categories.

Beyond this project's own results, Burhan tracks each adversary-emulation exercise as an individual "validation run." Alongside the historical APT29 LOTL run from this project — explicitly tagged Historical in the interface — several unrelated scenarios (lateral-movement, PowerShell-abuse, and credential-access validation runs, each tagged Demo) were exercised to demonstrate the tool's applicability beyond the single attack chain reported here. These demo runs use simulated data, are unrelated to the APT29 emulation, and are not presented as findings of this project; they illustrate only that the same validation approach could, in principle, extend to other emulated attack scenarios.

D. Scope and Future Work

Burhan remains an early-stage prototype rather than a production or validated component of the SOC pipeline described in this report. Its current data is imported or entered manually rather than pulled automatically from CALDERA or Elastic Security; live connectors to both platforms, as well as to Splunk and Microsoft Sentinel, exist in the interface only as prototype placeholders, with real API integration explicitly identified as future work. Section IX-B revisits this maturation path alongside the project's other planned enhancements.

VI. DISCUSSION

A. Distributed Lab Topology

Splitting the environment across five laptops (one to two VMs each) rather than hosting everything on a single hypervisor host traded some setup convenience for two benefits: it avoided resource contention on any one machine, and it forced the team to solve real LAN connectivity and addressing problems, which is closer to how a real organization's infrastructure is distributed than an all-in-one virtual lab would have been. The cost of this decision was coordination overhead — connectivity between machines had to be explicitly configured and verified via ping tests rather than assumed, and any one team member's laptop or network hiccup could stall the whole group, as happened during the Week 4 travel-related NTP and CALDERA connectivity disruption.

B. SIEM Choice

The Elastic Security vs. Wazuh comparison in Section III-B favored Elastic primarily on threat-hunting query language (KQL/EQL), dashboarding, and CALDERA integration. This is a defensible choice for a project centered on detection engineering and threat hunting specifically, though it comes with a trade-off worth naming: Elastic Security has a steeper operational footprint (Fleet Server, agent policies) than Wazuh, which was experienced directly when WS01 briefly reported an Unhealthy status during policy application. A project prioritizing simplicity or resource-constrained deployment might reasonably have chosen Wazuh instead.

C. Privilege Asymmetry by Design

Placing WS01 in the Employee OU with standard privileges and WS02 in the IT OU with administrative privileges was a deliberate design choice rather than an oversight, intended to create a realistic lateral-movement and privilege-escalation target. This is a common condition in real enterprises — an IT-support machine with broader rights than a typical user's — and is exactly the kind of asymmetry that identity-focused adversaries such as APT29 are known to exploit. It also gave the detection-engineering work two hosts with meaningfully different "normal" baselines to build detections against.

D. Validate-Before-Attack Discipline

Deliberately validating the logging pipeline with a benign, known action before executing any adversary emulation was a sound methodological choice: it isolates pipeline problems from detection-logic problems. Had this step been skipped in favor of going straight to TTP execution, a missing detection could have meant either "the SIEM did not see the event" or "the SIEM saw the event, but no rule matched it" — two very different problems that this validation step correctly separated throughout the project.

E. Detection Engineering as an Iterative Process

The rule-by-rule validation discipline described in Section III-I — author a rule, immediately trigger the corresponding technique, and confirm the alert before moving on — proved to be the single most valuable process decision in the detection-engineering phase. It surfaced the powershell.file.script_block_text field discrepancy, the WS01$ machine-account noise on the WinRM rule, and the benign-command false positives on the File Collection rule at the point each rule was introduced, rather than as an undifferentiated wave of failures during the final full-chain run. Tuning false positives is not a sign that a rule was poorly designed; in every case observed here it was the expected and necessary second half of writing a usable detection.

F. Defense in Depth in Detection

Several attack phases were detected by more than one rule simultaneously. The lateral-movement phase, for example, triggered both the Lateral Movement via WinRM rule (Event 4624, Logon Type 3) and the Explicit Credential Object Creation rule (Event 4104, PSCredential construction). The DCSync phase triggered both the DCSync Credential Dumping rule (Event 4662) and the Mimikatz Process Execution rule (Sysmon Event 1). This layered coverage means that an adversary evading one detection mechanism — for example, by suppressing the Event 4662 audit log — would still be caught by the process-layer rule, reducing the risk of a single point of detection failure and directly supporting the strong technique-coverage result reported in Section IV-I.

VII. ROOT CAUSE ANALYSIS

The adversary-emulation exercise revealed several underlying weaknesses in the environment that collectively enabled the attack chain to proceed from initial access through to exfiltration without automatic prevention at any stage. These weaknesses represent the root causes that a real incident-response team would be expected to identify and remediate following an intrusion of this nature.

A. Unrestricted LOLBin Execution at Initial Access

Initial access succeeded due to insufficient email-attachment controls and unrestricted LOLBin execution. The HTA-based phishing simulation succeeded because mshta.exe — a native Windows binary — faced no execution restrictions. In a hardened environment, application-control policies such as Windows Defender Application Control or AppLocker would prevent mshta.exe from executing remote or user-downloaded content, eliminating this initial-access vector entirely regardless of whether the user opens the attachment.

B. Absence of PowerShell Hardening Controls

Execution and persistence were unimpeded by PowerShell hardening controls. Throughout the attack chain, PowerShell was invoked with the Bypass execution policy, and no Constrained Language Mode was enforced. This permitted the execution of base64-encoded commands, unsigned scripts, and interactive remoting sessions without restriction. Enabling PowerShell Constrained Language Mode and requiring script signing would have significantly raised the cost of execution for the attacker, forcing reliance on noisier alternatives.

C. Excessive Privilege Assignment

Credential access via DCSync was possible due to excessive privilege assignment. The compromised account was elevated to Domain Admin during lab setup for operational convenience. In a production environment, granting Domain Admin rights to a standard user account represents a direct violation of the least-privilege principle. A properly scoped account would not possess the DS-Replication-Get-Changes rights required to perform DCSync, rendering the technique ineffective without a prior privilege-escalation step.

D. Weak Credential Hygiene and Absent Segmentation

Lateral movement was facilitated by weak credential hygiene and an unrestricted attack surface. The Administrator account on WS02 shared the same lab-wide password across the domain, enabling straightforward credential reuse following the initial compromise of WS01. Additionally, WinRM was enabled on WS02 with no network-level restrictions, and no segmentation policy existed between the Employee OU workstations and the IT OU workstations. Network segmentation enforced at the host firewall or VLAN level would have contained the breach to WS01 and prevented lateral movement to the more privileged WS02.

E. Detective but Not Preventive Persistence Controls

Persistence mechanisms went undetected in real time due to the absence of preventive controls. Registry Run-key modifications, scheduled-task creation, and WMI event-subscription registration all succeeded without triggering any preventive response. While the detective controls built in this project (Sysmon Events 13, 1, and 19/20/21) successfully generated alerts, the absence of endpoint protection capable of blocking these operations at execution time meant that persistence was established before any analyst could respond. This highlights the distinction between detective and preventive security controls, and the need for both layers in a mature security posture.

F. No Network-Level Exfiltration Controls

Data exfiltration completed without network-level intervention. The absence of Data Loss Prevention controls and outbound network filtering allowed PowerShell to establish a TCP connection from WS01 to the CALDERA C2 server at 172.20.10.20:9999 and transmit the contents of the staged CompanyData directory without interruption. A properly configured host-based firewall restricting outbound connections to approved destinations, combined with DLP controls monitoring for bulk file reads followed by network activity, would have provided an opportunity to block or alert on the exfiltration before data left the endpoint.

G. Synthesis

Taken together, these root causes reflect a pattern common in real enterprise environments: individual misconfigurations and policy gaps that are each defensible in isolation but collectively form an attack path that a patient, living-off-the-land adversary can traverse without triggering prevention controls at any stage. The detection-engineering work in this project demonstrates that behavioral detection can compensate for some of these gaps, but the root-cause analysis makes clear that detection alone is not a substitute for hardening the environment against initial access and lateral movement in the first place.

VIII. LESSONS LEARNED

The adversary-emulation exercise yielded several actionable insights across detection engineering, tooling, and operational practice.

A. Detection Relies on Context, Not Signatures

Every tool used in the attack chain — mshta.exe, powershell.exe, net.exe, schtasks.exe, reg.exe, and mimikatz.exe — is either a native Windows binary or a publicly available utility. Signature-based detection alone would have missed the majority of the techniques emulated. Effective detection required correlating parent–child process relationships, command-line argument patterns, registry paths, and network destinations rather than blocking tools by name. This validates the core premise of the project: living-off-the-land tradecraft demands behavioral detection engineering.

B. Script Block Logging Is the Highest-Fidelity Telemetry Source Available

PowerShell Script Block Logging (Event 4104) captured the decoded content of every PowerShell script before execution, providing visibility that persisted even when commands were base64-encoded or passed via encoded command-line arguments. In several cases, Script Block Logging provided deeper insight than Sysmon process-creation events alone, particularly for the lateral-movement and WMI-persistence phases. Organizations that have not enabled this logging source are operating with a significant blind spot against PowerShell-based living-off-the-land techniques.

C. Audit Policy Gaps Are a Common and Consequential Misconfiguration

Event 4662, required to detect DCSync attacks via directory replication access, did not appear in the SIEM until the "Directory Service Access" audit subcategory was explicitly enabled on DC01 using auditpol. This is a well-documented but frequently overlooked configuration gap in enterprise Active Directory environments, and its absence would have left DCSync — one of APT29's most impactful credential-access techniques — undetected despite a correctly written detection rule.

D. Validation Must Be Empirical, Not Assumed

During detection-rule development, the PowerShell Script Block field was assumed to be winlog.event_data.ScriptBlockText based on public documentation, but the field name in this Elastic stack was powershell.file.script_block_text. Rules written against the wrong field would have silently failed to generate alerts. This reinforced the discipline of validating every rule against a known benign action before relying on it for adversary detection — a principle applied consistently throughout the project, as described in Section III-I.

E. False-Positive Tuning Is Non-Trivial and Environment-Specific

Five distinct false-positive sources were identified and remediated during the exercise: OneDriveSetup.exe and OneDriveStandaloneUpdater.exe triggering the suspicious-directory rule during routine Microsoft update activity; FileCoAuth.exe (Office co-authoring) doing the same; and DismHost.exe spawning from cleanmgr.exe during disk cleanup. Each required investigation to distinguish legitimate system behavior from adversary activity, mirroring the analyst workload in a real SOC. The CALDERA Sandcat agent itself (splunkd.exe, staged in C:\Users\Public) was intentionally excluded from the rule with explicit

documentation, acknowledging that in a production environment the same indicator would warrant immediate investigation.

F. Defense in Depth in Detection Provides Resilience

As discussed in Section VI-F, several attack phases were detected by more than one rule simultaneously. This layered coverage means that an adversary evading one detection mechanism would still be caught by a second, independent rule, reducing the risk of a single point of detection failure.

IX. LIMITATIONS AND FUTURE WORK

A. Limitations

- Small-scale environment. The lab consists of a single domain controller and two workstations, which is sufficient to demonstrate the AD attack surface APT29 typically targets but does not capture the scale, noise, or heterogeneity — multiple OUs, trust relationships, cloud-hybrid identity — of a real enterprise network.
- CALDERA default agents and abilities are known to defenders. Because CALDERA's out-of-the-box agents and TTP library are publicly known, detections built against them may not generalize to a real APT29 actor using custom tooling; this is an inherent limitation of using an emulation platform rather than a claim specific to this project.
- Golden Ticket forgery (T1558.001) was scoped as a dedicated follow-up operation (Section III-H) but was not executed within the project timeline. The DCSync step successfully extracted the krbtgt hash, but no forged ticket was generated or tested against the environment, so the expected detection gap discussed in the project plan remains a predicted rather than an observed finding.
- MTTC, MTTR, and the false-positive rate are procedure-based estimates rather than measurements drawn from a live, unscripted incident (Section IV-I); only MTTD and MITRE ATT&CK technique coverage were measured directly from timestamped evidence.
- Planned enhancements — network-level visibility via Packetbeat and an AI-assisted alert-triage summary script — were scoped in the project plan but were not completed and validated within the project timeline, and are carried forward as future work below rather than reported as results.

B. Future Work

- Execute the Golden Ticket forgery operation (T1558.001) as a dedicated follow-up using the already-extracted krbtgt hash, and document the resulting detection outcome — expected to surface a genuine coverage gap given the absence of a dedicated Golden Ticket rule.
- Deploy Packetbeat on ELASTIC01 to add network-level visibility alongside the existing Sysmon endpoint coverage, particularly for WinRM and lateral-movement traffic at the network layer.
- Complete the AI-assisted threat-summary script (Elastic alert API → LLM analysis → automated triage note) and integrate it into the alert-response workflow.

- Perform a Sysmon configuration hardening pass — replacing the default configuration with a community baseline such as SwiftOnSecurity or Olaf Hartong's — and re-run the attack chain to quantify the before/after improvement in coverage and fidelity.
- Extend the environment with a second domain controller or a small trust relationship to better approximate a multi-site enterprise Active Directory deployment.
- Introduce a SOAR platform to automate the containment and eradication steps currently executed manually per the Appendix A playbook.
- Add formal risk-scoring and evaluation metrics aligned with the NIST Cybersecurity Framework to complement the SOC metrics reported in Section IV-I.
- Mature the Burhan prototype introduced in Section V from imported/simulated data toward live API integration with CALDERA and Elastic Security, enabling automated, real-time correlation of emulation, detection, and response evidence for future attack scenarios beyond APT29.

X. CONCLUSION

This project set out to build a safe, isolated environment for emulating APT29-style adversary behavior against an Active Directory network, evaluate the resulting SOC detection capability, and carry that evaluation through to a full incident-response cycle. Over five weeks, the team selected the APT29 scenario, evaluated and chose Elastic Security as the SIEM platform, and built a five-host distributed lab spanning a domain controller, two privilege-differentiated workstations, a CALDERA adversary-emulation server, and a fully validated Sysmon/PowerShell/Elastic Agent logging pipeline.

A nine-technique, eight-phase attack chain was designed, iteratively debugged, and executed end-to-end against the environment, and sixteen MITRE ATT&CK-mapped detection rules were engineered and tuned against real false positives to achieve complete technique coverage on the final validation run, with a measured mean time to detect of 18.4 seconds. Six independent threat hunts confirmed that the same attack timeline could be reconstructed analytically from raw telemetry, and a full NIST-aligned incident-response cycle — detection, containment, eradication, recovery, and post-incident review — was carried out and formalized into a reusable playbook.

The root-cause analysis and lessons learned presented in Sections VII and VIII make clear that the project's strongest result is not any single detection rule, but the demonstration that behavioral, telemetry-driven detection engineering — built on Script Block Logging, Sysmon, and disciplined empirical validation — can achieve reliable coverage against a living-off-the-land adversary that no signature-based control would catch, while also making explicit that detection alone cannot substitute for the preventive hardening measures identified in Section VII. Every stage of the environment, the attack chain, and the response was verified functionally rather than assumed to work, culminating in a body of evidence — from initial ping tests to final credential resets — that a real SOC team could use both as a training exercise and as a template for validating their own detection coverage against this adversary profile.

Beyond the completed exercise, this project also took an initial step toward addressing the manual-correlation observation described in Section IV-I: the Burhan prototype presented in Section V demonstrates, at a conceptual level, how the emulation, detection, and response evidence produced here could be brought together into a single measurable validation layer. Burhan remains an early-stage

extension rather than a validated component of the SOC pipeline, but it points toward a natural direction for future capstone or production work — a unified, evidence-driven way of asking not only "did we detect it," but "did we detect it, respond to it, and prove that we did."

REFERENCES

[1] MITRE ATT&CK, "APT29, Group G0016," MITRE Corporation. [Online]. Available: https://attack.mitre.org/groups/G0016/. Accessed: Jul. 2026.

[2] MITRE Corporation, "MITRE ATT&CK." [Online]. Available: https://attack.mitre.org/. Accessed: Jul. 2026.

[3] MITRE Corporation, "CALDERA: Automated Adversary Emulation Platform." [Online]. Available: https://caldera.readthedocs.io/. Accessed: Jul. 2026.

[4] Elastic N.V., "Elastic Security Solution Documentation." [Online]. Available: https://www.elastic.co/guide/en/security/current/index.html.

[5] National Institute of Standards and Technology, "Computer Security Incident Handling Guide," NIST Special Publication 800-61 Rev. 2, Aug. 2012.

[6] Microsoft, "about_Logging_Windows," Microsoft Learn — PowerShell documentation. [Online]. Available: https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_logging_windows.

[7] M. Russinovich and T. Garnier, "Sysmon — Windows Sysinternals," Microsoft. [Online]. Available: https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon.

[8] B. Delpy, "mimikatz," GitHub repository. [Online]. Available: https://github.com/gentilkiwi/mimikatz.

[9] S. Metcalf, "Mimikatz DCSync Usage, Exploitation, and Detection," Active Directory Security. [Online]. Available: https://adsecurity.org/?p=1729.

[10] MITRE ATT&CK, individual technique references: T1566.001, T1218.005, T1059.001, T1027, T1082, T1069, T1547.001, T1053.005, T1003.006, T1021.006, T1546.003, T1119, T1041, T1550.002. [Online]. Available: https://attack.mitre.org/techniques/.

APPENDIX A

INCIDENT RESPONSE PLAYBOOK — APT-29 LOTL, CORP.LOCAL LAB

This appendix reproduces, in full, the incident-response playbook produced as a deliverable of this project. It is written to be directly reusable by a SOC analyst responding to this adversary profile, independent of the narrative results presented in Section IV-G.

TABLE IX
Playbook Metadata

Field	Value
Organization	KAUST Academy Summer Cybersecurity Program
Classification	Internal — Isolated Lab Environment
Reference Framework	NIST SP 800-61 Rev. 2
Version	1.0 — August 2026
Author / IR Lead	Murtadha Alsaleh

A. Quick Reference — Artifacts and Hosts

TABLE X
Artifact Locations

Artifact	Location	Host
CALDERA agent	C:\Users\Public\splunkd.exe	WS01, WS02, DC01
Mimikatz	%TEMP%\mimikatz\x64\mimikatz.exe	DC01
Registry Run key	HKCU:\...\CurrentVersion\Run — "Atomic Red Team"	WS01
Scheduled task	Task Scheduler — "spawn" (cmd.exe, every 5 min)	WS01
WMI subscription	root\subscription — AtomicRedTeam-WMIPersistence-*	WS02
Staging files	%TEMP%\T1119_1.txt … T1119_5.txt	WS01
Honeypot account	svc-backup (IT OU, DC01) — zero legitimate use	DC01
Compromised account	CORP\ws01 (elevated during lab setup)	DC01

TABLE XI
Host Reference

Host	IP	Role
DC01	172.20.10.30	Domain Controller — Windows Server 2022
WS01	172.20.10.40	Primary compromised host — Windows 10
WS02	172.20.10.50	Lateral-movement target — Windows 10
CALDERA01	172.20.10.20	C2 server — ports 8888 and 9999

Host	IP	Role
ELASTIC01	172.20.10.10	SIEM — Elastic Security / Kibana

B. Phase 1 — Detection and Triage

1) Confirm Alerts in Kibana

- Open Kibana — http://172.20.10.10:5601 → Security → Alerts.
- Set the time range to the last 24 hours, or narrow to the suspected attack window.
- Sort by @timestamp ascending to build the attack timeline from the first alert.
- For each alert: open it, read the Reason field, and note host.name and user.name.

2) Triage Decision

TABLE XII
Triage Decision Matrix

Alert Severity	Immediate Action
Critical (red)	Escalate immediately — begin containment within 15 minutes.
High (orange)	Investigate within 1 hour — correlate with adjacent alerts.
Medium (yellow)	Investigate within 4 hours — check for false positive first.

Note: Before dismissing any alert as a false positive, verify the process name against the known false-positive list — OneDrive.exe, OneDriveStandaloneUpdater.exe, FileCoAuth.exe, DismHost.exe. Any other process executing from AppData, Windows\Temp, or Users\Public must be treated as suspicious.

C. Phase 2 — Containment

Execute the steps in this order. Do not proceed to Eradication until every containment step is verified.

1) Isolate WS01 (Primary Compromised Host)

```
# PowerShell — run on WS01 as Administrator
Disable-NetAdapter -Name "Ethernet" -Confirm:$false

# Verify
Get-NetAdapter | Select-Object Name, Status   # Expected: Disabled
```

2) Stop the CALDERA Agent on All Hosts

```
Get-Process splunkd -ErrorAction SilentlyContinue | Stop-Process -Force
```

3) Disable the Compromised Account on DC01

```
Disable-ADAccount -Identity "ws01"

# Verify
Get-ADUser ws01 | Select-Object Name, Enabled   # Expected: False
```

4) Containment Checklist

☐ WS01 network adapter disabled

☐ CALDERA agent stopped on WS01

☐ CALDERA agent stopped on WS02

☐ CALDERA agent stopped on DC01

☐ Compromised account disabled in Active Directory

D. Phase 3 — Eradication

Remove all adversary artifacts in the order below, verifying each removal before proceeding to the next.

1) Registry Run Key — WS01

```
Remove-ItemProperty -Path "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" `
  -Name "Atomic Red Team" -Force

# Verify — "Atomic Red Team" must not appear in output
Get-ItemProperty "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run"
```

2) Scheduled Task — WS01

```
Unregister-ScheduledTask -TaskName "spawn" -Confirm:$false

# Verify — must return no output
Get-ScheduledTask -TaskName "spawn" -ErrorAction SilentlyContinue
```

3) WMI Event Subscription — WS02

```
# Remove in order: binding -> consumer -> filter
Get-CimInstance -Namespace root/subscription -ClassName __FilterToConsumerBinding |
  Where-Object { $_.Filter.Name -like "*AtomicRedTeam*" } | Remove-CimInstance

Get-CimInstance -Namespace root/subscription -ClassName CommandLineEventConsumer |
  Where-Object { $_.Name -like "*AtomicRedTeam*" } | Remove-CimInstance

Get-CimInstance -Namespace root/subscription -ClassName __EventFilter |
  Where-Object { $_.Name -like "*AtomicRedTeam*" } | Remove-CimInstance

# Verify — must return no output
Get-CimInstance -Namespace root/subscription -ClassName __EventFilter
```

4) Mimikatz — DC01

```
Remove-Item -Path "$env:TEMP\mimikatz" -Recurse -Force

# Verify — must return False
Test-Path "$env:TEMP\mimikatz"
```

5) CALDERA Agent Binary — All Hosts

```
Remove-Item "C:\Users\Public\splunkd.exe" -Force -ErrorAction SilentlyContinue
Test-Path "C:\Users\Public\splunkd.exe"   # Must return False
```

6) Staging Files — WS01

```
Remove-Item "$env:TEMP\T1119_*.txt" -Force -ErrorAction SilentlyContinue
Get-ChildItem "$env:TEMP" -Filter "T1119_*"   # Must return no items
```

7) Eradication Checklist

☐ Registry Run key "Atomic Red Team" removed — WS01

☐ Scheduled task "spawn" removed — WS01

☐ WMI EventFilter, Consumer, and Binding removed — WS02

☐ mimikatz directory removed — DC01

☐ splunkd.exe removed — WS01, WS02, DC01

☐ T1119_*.txt staging files removed — WS01

E. Phase 4 — Recovery

1) Reset Credentials

```
Set-ADAccountPassword -Identity "ws01" -Reset `
  -NewPassword (ConvertTo-SecureString "NewSecure2026!" -AsPlainText -Force)
Set-ADUser ws01 -ChangePasswordAtLogon $true
Enable-ADAccount -Identity "ws01"

# Verify
Get-ADUser ws01 | Select-Object Name, Enabled   # Must show True
```

2) Reset the krbtgt Account Password

Reset the krbtgt account password twice, with a minimum 72-hour interval between resets, to invalidate any Kerberos ticket material that could have been forged from credential material extracted during the DCSync attack. In this lab, the DCSync attack extracted the krbtgt hash but no Golden Ticket was created (Section IX-A); the double-reset procedure is documented here for completeness and should be rehearsed as part of recovery training.

3) Restore WS01 Network

```
Enable-NetAdapter -Name "Ethernet" -Confirm:$false
Get-NetAdapter | Select-Object Name, Status   # Must show Up
```

4) Verify SIEM Visibility

```
event.code: "1" and host.name: "ws01"
event.code: "1" and host.name: "ws02"
event.code: "1" and host.name: "dc01"
```

5) Recovery Checklist

☐ Compromised account password reset and change-at-logon enforced

☐ Compromised account re-enabled

☐ krbtgt password reset (first reset complete)

☐ krbtgt password reset (second reset — 72h later)

☐ WS01 network adapter re-enabled

☐ SIEM receiving events from WS01, WS02, DC01

☐ All detection rules confirmed active in Elastic Security

F. Post-Incident Actions

TABLE XIII
Post-Incident Action Items

Action	Owner	Deadline
Complete IR report — timeline, root cause, findings	IR Lead (Murtadha)	Within 5 business days
Update detection rules based on gaps identified	SIEM Analyst (Hassan)	Within 5 business days
Enforce least privilege — remove Domain Admin from the compromised account	DC Admin (Abdullah)	Immediately post-recovery
Enable PowerShell Constrained Language Mode via GPO	DC Admin (Abdullah)	Within 2 weeks
Deploy Application Control (WDAC) restricting mshta.exe	DC Admin (Abdullah)	Within 1 month
Implement network segmentation — Employee OU vs. IT OU	Network / DC Admin	Within 1 month
Second krbtgt password reset (72h after first)	DC Admin (Abdullah)	Scheduled

APPENDIX B

COMPLETE DETECTION RULE CATALOGUE

This appendix reproduces the exact query, MITRE ATT&CK mapping, risk score, and known false-positive examples for all sixteen detection rules summarized in Table V (Section IV-C). All rules run on a one-minute execution interval with a five-minute look-back window in Elastic Security.

B.1. APT29 - Phishing via HTA and Mshta Proxy Execution

MITRE ATT&CK: T1566.001 / T1218.005 / T1059.001 **Risk Score:** 99

```
event.code: "1" and process.name: "powershell.exe" and process.parent.name: "mshta.exe"
```

Description: Detects PowerShell spawned by mshta.exe, indicating execution of a malicious HTA file delivered via spearphishing. mshta.exe is a native Windows LOLBin abused by APT29 to proxy execution while evading signature-based defenses.

Known False Positives: Legitimate use of mshta.exe spawning PowerShell is extremely rare in enterprise environments; any instance of this parent-child relationship should be treated as high-confidence malicious activity.

B.2. APT29 - Obfuscated PowerShell Execution

MITRE ATT&CK: T1059.001 / T1027 **Risk Score:** 73

```
event.code: "1" and (process.name: "powershell.exe" or process.command_line: *powershell*)
and (process.command_line: *-EncodedCommand* or *-enc* or *-e * or *frombase64string* or
*-nop* or *-bypass* or *-windowstyle hidden* or *-w hidden* or *invoke-expression*)
```

Description: Detects obfuscated or encoded PowerShell commands commonly used by APT29 for initial execution and defense evasion.

Known False Positives: Legitimate administrative PowerShell scripts using encoding for automation purposes.

B.3. APT29 - PowerShell Script Block Suspicious Content

MITRE ATT&CK: T1059.001 (Execution) / T1546.003, T1021.006 (further stages) **Risk Score:** 73

```
event.code: "4104" and (powershell.file.script_block_text: *Invoke-Command* or
*New-CimInstance* or *lsadump* or *ConvertTo-SecureString* or *Register-ScheduledTask* or
*Net.Sockets.TcpClient*)
```

Description: Detects suspicious PowerShell script content captured via Script Block Logging (Event 4104), including lateral-movement commands, WMI persistence registration, credential dumping, scheduled-task creation, and C2 socket connections. Captures the actual decoded script content before execution, making it significantly harder for adversaries to evade through encoding or obfuscation.

Known False Positives: Legitimate administrative automation may use Invoke-Command for remote management, Register-ScheduledTask for maintenance jobs, or New-CimInstance for WMI-based monitoring.

B.4. APT29 - Process Execution from Suspicious Directory

MITRE ATT&CK: T1036 / T1059 **Risk Score:** 47

```
event.code: "1" and (process.executable: *AppData* or *Windows\Temp* or *Users\Public*) and
not process.name: ("MicrosoftEdgeUpdate.exe" or "setup.exe" or "install.exe" or
"FileCoAuth.exe" or "splunkd.exe" or "DismHost.exe" or "OneDrive.exe" or
"OneDriveStandaloneUpdater.exe")
```

Description: Detects execution of processes from user-writable directories commonly abused by adversaries for staging and executing malicious payloads. In this environment, both Mimikatz and the CALDERA Sandcat agent are staged in these paths, making this rule effective at catching both the initial foothold and post-exploitation tooling.

Known False Positives: Legitimate software installers, browser updates, and temporary application files may execute from AppData\Local\Temp or Windows\Temp during normal installation or update processes.

B.5. APT29 - Honeypot Account Access Detected

MITRE ATT&CK: T1087 (Discovery, deception control) **Risk Score:** 99

```
(winlog.event_data.TargetUserName: "svc-backup" or user.name: "svc-backup") and (event.code:
"4624" or "4625" or "4768" or "4769")
```

Description: Any authentication attempt against the decoy account svc-backup indicates potential adversarial discovery or lateral movement. This account has no legitimate use.

Known False Positives: None — the account is a zero-legitimate-use honeypot; any hit is treated as a genuine indicator.

B.6. APT29 - Registry Run Key Persistence

MITRE ATT&CK: T1547.001 **Risk Score:** 77

```
event.code: "13" and registry.path: (*CurrentVersion\Run* or *CurrentVersion\RunOnce*) and
not process.name: ("OneDriveSetup.exe" or "MsMpEng.exe" or "SppExtComObj.exe" or
"msiexec.exe" or "TrustedInstaller.exe" or "msedge.exe") and (registry.data.strings: *Temp*
or *Public* or *AppData*)
```

Description: Detects registry modifications to autorun keys commonly used by APT29 for persistence on victim workstations.

Known False Positives: Legitimate software installations may modify Run keys during setup; excluded process names cover the observed cases.

B.7. APT29 - Mimikatz Process Execution

MITRE ATT&CK: T1003.006 **Risk Score:** 99

```
event.code: "1" and (process.name: "mimikatz.exe" or process.command_line: *mimikatz* or
*lsadump* or *dcsync* or *sekurlsa*)
```

Description: Detects execution of Mimikatz or Mimikatz-like credential-dumping tools based on process name and command-line arguments. Complements the DCSync Event 4662 detection by catching the offensive tool at the process-execution layer, providing defense-in-depth coverage for credential-access attempts including DCSync, LSA dumping, and Kerberos ticket extraction.

Known False Positives: Security researchers or penetration testers running Mimikatz in authorized red-team exercises. In this lab environment, any execution outside of planned emulation windows should be treated as a critical incident.

B.8. APT29 - PowerShell Suspicious Network Outbound

MITRE ATT&CK: T1071.001 / T1041 **Risk Score:** 99

```
event.code: "1" and process.name: "powershell.exe" and process.command_line:
(*Invoke-WebRequest* or *WebClient* or *Net.Sockets.TcpClient*) and process.command_line:
*172.20.10.20*
```

Description: Detects PowerShell processes initiating outbound network connections to the CALDERA C2 server using web-request or socket cmdlets in the command line. Provides process-layer visibility into data-exfiltration attempts, complementing the network-layer Exfiltration rule (T1041) by detecting the offensive command before the connection is established.

Known False Positives: Legitimate administrative scripts may use Invoke-WebRequest to contact internal servers. In this lab, any PowerShell process referencing 172.20.10.20 with web or socket cmdlets outside planned emulation windows should be treated as a critical incident.

B.9. APT29 - Data Exfiltration Over C2 Channel

MITRE ATT&CK: T1041 **Risk Score:** 99

```
event.code: "3" and process.name: ("powershell.exe" or "cmd.exe") and destination.ip:
"172.20.10.20" and destination.port: (8888 or 7010 or 4433 or 9999) and not source.ip:
"172.20.10.20" and not process.parent.name: ("services.exe" or "svchost.exe")
```

Description: Detects outbound network connections initiated by PowerShell or CMD toward the CALDERA C2 server on known C2 ports, indicating potential data exfiltration over the command-and-control channel following a successful compromise.

Known False Positives: Legitimate administrative tools may initiate network connections to internal servers; excluded parent processes and destination-port scoping minimize this risk.

B.10. APT29 - Scheduled Task Persistence

MITRE ATT&CK: T1053.005 **Risk Score:** 73

```
((process.name: "schtasks.exe" and process.command_line: *create*) or (process.name:
"powershell.exe" and process.command_line: *Register-ScheduledTask*)) and not
process.parent.name: ("services.exe" or "taskeng.exe" or "taskhostw.exe")
```

Description: Detects creation of scheduled tasks via schtasks.exe or the Register-ScheduledTask PowerShell cmdlet, commonly used by APT29 for persistent execution on compromised hosts.

Known False Positives: Legitimate software installers and system administrators may create scheduled tasks during application setup or maintenance.

B.11. APT29 - DCSync Credential Dumping

MITRE ATT&CK: T1003.006 **Risk Score:** 99

```
event.code: "4662" and winlog.event_data.Properties: (*1131f6aa* or *1131f6ad* or
*89e95b76*) and winlog.event_data.AccessMask: "0x100" and not
winlog.event_data.SubjectUserName: (*$)
```

Description: Detects DCSync attacks by monitoring Security Event ID 4662 for access to Active Directory replication GUIDs (DS-Replication-Get-Changes), indicating an attempt to extract credential material including the krbtgt hash for Golden Ticket forgery.

Known False Positives: Domain Controllers legitimately replicate directory data between each other; the rule excludes computer accounts (trailing $). Any non-DC account triggering this rule should be treated as a critical incident.

B.12. APT29 - Explicit Credential Object Creation

MITRE ATT&CK: T1550.002 / T1021.006 **Risk Score:** 73

```
event.code: "4104" and (powershell.file.script_block_text: *New-Object
System.Management.Automation.PSCredential* or *ConvertTo-SecureString*) and
powershell.file.script_block_text: *Invoke-Command*
```

Description: Detects PowerShell scripts that explicitly construct credential objects using PSCredential and ConvertTo-SecureString in combination with Invoke-Command, indicating credential-based lateral movement over WinRM. This pattern is characteristic of APT29 tradecraft where stolen or known credentials are used programmatically to authenticate to remote systems without interactive logon.

Known False Positives: Legitimate administrative automation scripts may construct PSCredential objects for remote-management tasks such as software deployment or configuration management.

B.13. APT29 - Lateral Movement via WinRM

MITRE ATT&CK: T1021.006 **Risk Score:** 73

```
event.code: "4624" and winlog.event_data.LogonType: "3" and host.name: "ws02" and
winlog.event_data.AuthenticationPackageName: "Kerberos" and not
winlog.event_data.TargetUserName: (*$ or "ANONYMOUS LOGON")
```

Description: Detects network logon sessions (Logon Type 3) originating from WS01 toward WS02, indicating credential-based lateral movement over Windows Remote Management. Tuned to exclude machine-account (trailing $) and anonymous logons after false-positive analysis identified WS01$ domain-operation noise (Section III-I1).

Known False Positives: Legitimate administrative remote sessions from IT workstations may trigger this rule; verify the source host, destination, and user account before escalating.

B.14. APT29 - File Collection via PowerShell

MITRE ATT&CK: T1119 **Risk Score:** 50

```
event.code: "1" and process.name: ("powershell.exe" or "cmd.exe") and (process.command_line:
*Get-ChildItem* or *Copy-Item* or *Get-Content* or *Compress-Archive*) and not
(process.command_line: *Get-Location* or *Test-Path* or *Join-Path* or *Invoice*) and
(process.command_line: *CompanyData* or *DepartmentData* or *T1119*)
```

Description: Detects file enumeration and collection commands executed via PowerShell or CMD, indicating an adversary gathering data from the victim host prior to exfiltration. Tuned to exclude benign

path-handling commands (Get-Location, Test-Path, Join-Path) that partially matched during initial validation against the HTA spearphishing payload (Section III-I1).

Known False Positives: Legitimate administrative scripts and IT tools frequently use Get-ChildItem, dir, and Copy-Item for file management; verify user context, parent process, and whether the activity correlates with known administrative tasks.

B.15. APT29 - WMI Event Subscription Persistence

MITRE ATT&CK: T1546.003 **Risk Score:** 99

```
sequence by host.name [any where event.code == "19"] [any where event.code == "20"] [any
where event.code == "21"]
```

Description: Detects WMI event-subscription persistence by correlating Sysmon Event IDs 19 (WmiEventFilter), 20 (WmiEventConsumer), and 21 (WmiEventConsumerToFilter binding), indicating registration of a persistent WMI execution trigger.

Known False Positives: Legitimate software may register WMI subscriptions for monitoring purposes; verify the process name, subscription name, and query content before escalating.

B.16. APT29 - System and AD Discovery

MITRE ATT&CK: T1082 / T1069 / T1007 **Risk Score:** 47

```
event.code: "1" and (process.name: ("whoami.exe" or "net.exe" or "nltest.exe" or
"systeminfo.exe" or "ipconfig.exe" or "hostname.exe" or "reg.exe") and process.parent.name:
("powershell.exe" or "cmd.exe")) and not process.command_line: (*localgroup* and
*administrators*)
```

Description: Detects discovery commands (whoami, net, nltest, systeminfo) spawned by PowerShell or CMD, indicative of post-exploitation system and Active Directory enumeration.

Known False Positives: Legitimate IT administrative scripts may spawn discovery commands via PowerShell for inventory or monitoring purposes; verify the user context, parent process, and whether the activity correlates with a known maintenance window.