DETECTION LAB ARCHITECTURE

Technical Design Document

MITRE ATT&CK-Aligned Detection Engineering Laboratory

Version 1.0

January 2026

UNCLASSIFIED // FOR OFFICIAL USE ONLY

Table of Contents

Executive Summary

This document defines the technical architecture for a comprehensive Detection Engineering Laboratory designed to validate security detection capabilities against the MITRE ATT&CK framework. The lab provides an isolated, instrumented environment where security teams can safely execute adversary techniques, generate telemetry, develop detection rules, and train analysts.

The architecture supports the full detection engineering lifecycle: technique execution, telemetry collection, rule development, validation testing, and analyst training. All components are designed for rapid deployment, reproducibility, and alignment with enterprise security architectures.

Design Objectives

- Provide safe, isolated execution environment for ATT&CK technique testing
- Generate authentic telemetry matching production security tooling
- Enable rapid iteration on detection rule development and validation
- Support automated testing via Atomic Red Team and custom test harnesses
- Facilitate analyst training with realistic attack scenarios
- Maintain full MITRE ATT&CK coverage tracking and gap analysis

Scope

This architecture covers Windows and Linux endpoint detection, Active Directory attack paths, network-based detection, and cloud integration points. It does not cover mobile platforms, OT/ICS environments, or classified systems, though the patterns are extensible to those domains.

Architecture Overview

Network Topology

The detection lab implements a segmented network architecture with strict isolation from production environments. Traffic flows through monitored chokepoints to ensure complete visibility while preventing lateral movement to corporate infrastructure.

Zone	VLAN	Subnet	Purpose
Management	10	10.10.10.0/24	Lab management, orchestration, jumpbox access
Victim Domain	20	10.10.20.0/24	Windows AD forest, domain-joined endpoints
Victim Linux	25	10.10.25.0/24	Linux servers, containers, cloud simulation
Attacker	30	10.10.30.0/24	Red team systems, C2 infrastructure
Logging	40	10.10.40.0/24	SIEM, log collectors, analysis platforms
Services	50	10.10.50.0/24	DNS, DHCP, NTP, update servers

Traffic Flow Design

All inter-VLAN traffic routes through a monitored firewall with full packet capture capability. The firewall implements default-deny with explicit allow rules for required lab communications. A network TAP feeds traffic to Zeek/Suricata sensors for protocol analysis.

- Attacker VLAN can reach Victim VLANs only through defined attack paths
- Victim VLANs can reach Services VLAN for legitimate infrastructure
- All VLANs forward logs to Logging VLAN via dedicated syslog path
- Management VLAN has out-of-band access to all systems for orchestration
- No direct path exists between lab networks and production/corporate

Isolation Architecture

The lab implements defense-in-depth isolation to prevent escape of malware or lateral movement to production systems. Multiple layers ensure containment even if individual controls fail.

Layer	Control	Implementation
Physical	Air gap option	Dedicated hardware with no physical uplinks to corporate
Network	Firewall isolation	pfSense/OPNsense with explicit deny to RFC1918 corporate ranges
Hypervisor	Separate cluster	Dedicated Proxmox/ESXi cluster, no vMotion to production
DNS	Internal resolution	Lab DNS servers with no forwarding to corporate resolvers

Internet	Controlled egress	Optional breakout through dedicated ISP or cellular for C2 testing
Snapshots	Rapid recovery	Automated snapshots before each test, instant rollback capability

Infrastructure Components

Virtualization Platform

The lab runs on a dedicated virtualization cluster providing compute, storage, and networking for all lab components. Proxmox VE is recommended for its open-source nature, snapshot capabilities, and API-driven automation.

Component	Specification	Notes
Hypervisor	Proxmox VE 8.x or VMware ESXi 8.x	API access for automation required
Compute Nodes	3x servers, 64+ cores total, 512GB+ RAM	Enables parallel test execution
Storage	NVMe SSD, 10TB+ usable	Fast snapshots critical for rapid iteration
Networking	10GbE between nodes, 1GbE to lab	VxLAN or VLAN trunking for isolation
Backup	Dedicated NAS, 50TB+	Full VM backups, detection rule versioning

Hardware Requirements

Minimum hardware for a functional detection lab supporting 20+ concurrent VMs with acceptable performance for interactive testing and automated pipelines.

Tier	Hosts	CPU	RAM	Storage	Use Case
Minimal	1	16 cores	128GB	2TB NVMe	Individual analyst, limited parallelism
Standard	3	64 cores total	384GB total	10TB NVMe	Team use, CI/CD integration
Enterprise	5+	160+ cores	1TB+ total	50TB+ NVMe	Full ATT&CK coverage, parallel testing

Network Infrastructure

- Core switch: Managed switch with VLAN support, port mirroring, 10GbE uplinks
- Firewall: pfSense/OPNsense VM or dedicated appliance with 10Gbps throughput
- Network TAP: Passive TAP or SPAN port feeding Zeek/Suricata sensors
- Wireless (optional): Isolated SSID for mobile device testing

Victim Environment

Windows Domain Architecture

The victim environment replicates a realistic enterprise Windows domain with multiple trust levels, GPO configurations, and typical misconfigurations that enable common attack paths.

System	OS	Role	Configuration
DC01	Server 2022	Primary Domain Controller	AD DS, DNS, DHCP, GPO, Certificate Services
DC02	Server 2019	Secondary DC	Replication partner, backup authentication
FS01	Server 2022	File Server	SMB shares, DFS namespace, sensitive data
SQL01	Server 2019	Database Server	SQL Server 2019, sample databases
WEB01	Server 2022	IIS Web Server	Vulnerable web apps (DVWA, WebGoat)
WS01-WS05	Windows 11 Pro	Workstations	Domain-joined, Office 365, various software
WS10	Windows 10 LTSC	Legacy Workstation	Older configs, reduced security

Active Directory Configuration

The AD forest includes intentional misconfigurations representing common enterprise vulnerabilities. These enable testing of credential-based attacks, privilege escalation, and lateral movement techniques.

- Kerberoastable service accounts with weak passwords
- AS-REP roastable accounts (no pre-authentication required)
- Unconstrained delegation on select servers
- Privileged group nesting (Domain Users -> Backup Operators chain)
- Sensitive data in SYSVOL (GPP passwords, scripts with credentials)
- ACL misconfigurations (WriteDACL, GenericAll on sensitive objects)
- Tiered admin model violations (T0 creds used on T1/T2 systems)

Linux Environment

System	OS	Role	Configuration
LIN01	Ubuntu 22.04 LTS	General Server	SSH, web services, development tools
LIN02	RHEL 9	Enterprise Linux	SSSD domain-joined, enterprise apps
LIN03	Debian 12	Database Server	PostgreSQL, MySQL, exposed services

DOCK01	Ubuntu 22.04	Container Host	Docker, Kubernetes (k3s), container escapes
KALI01	Kali Linux	Attack Platform	Attacker VLAN, full toolkit installed

Cloud Simulation

For testing cloud-based attacks, the lab includes simulated cloud services or connections to dedicated cloud tenants used exclusively for security testing.

- LocalStack for AWS API simulation (S3, IAM, Lambda, EC2 metadata)
- Azurite for Azure Blob/Queue/Table storage simulation
- Dedicated Azure AD tenant for identity attack testing (separate from production)
- GCP Cloud Shell simulation for credential harvesting scenarios

Attacker Environment

Red Team Systems

The attacker environment provides tooling for executing ATT&CK techniques in a controlled manner. Systems are isolated in a dedicated VLAN with monitored egress paths.

System	OS	Purpose	Key Tools
KALI01	Kali Linux 2024.x	Primary attack platform	Metasploit, Impacket, BloodHound, CrackMapExec
COMMANDO	Windows 11	Windows-based attacks	Rubeus, Mimikatz, SharpHound, Covenant
C2-HTTP	Ubuntu 22.04	HTTP/S C2 Server	Cobalt Strike, Mythic, Sliver
C2-DNS	Ubuntu 22.04	DNS Tunneling	dnscat2, Iodine, custom resolvers
PHISH01	Ubuntu 22.04	Phishing Infrastructure	Gophish, Evilginx2, mail server

C2 Framework Matrix

Multiple C2 frameworks provide diverse implant signatures for detection rule testing. Each framework generates unique network and host artifacts.

Framework	Protocols	Key Artifacts	Detection Focus
Cobalt Strike	HTTP/S, DNS, SMB	Named pipes, malleable C2 profiles	JA3/JA4, beacon timing, named pipes
Mythic	HTTP/S, websockets	Agent callbacks, custom payloads	HTTP patterns, process injection
Sliver	HTTP/S, mTLS, DNS, WireGuard	Implant staging, pivoting	Certificate anomalies, DNS patterns
Havoc	HTTP/S, SMB	Demon agent, BOF execution	Memory artifacts, API sequences
Metasploit	HTTP/S, TCP reverse	Meterpreter, shellcode	Classic signatures, staging patterns

Atomic Red Team Integration

Atomic Red Team provides a library of discrete, atomic tests mapped to ATT&CK techniques. The lab integrates ART for automated, reproducible technique execution.

```
# Install Atomic Red Team

IEX (IWR
'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/i
nstall-atomicredteam.ps1' -UseBasicParsing)
```

```
Install-AtomicRedTeam -getAtomics

# Execute specific technique
Invoke-AtomicTest T1003.001 -TestNumbers 1  # LSASS dump via procdump
Invoke-AtomicTest T1059.001 -TestNumbers 1,2,3  # PowerShell execution

# Execute all techniques for a tactic
Invoke-AtomicTest T1053 -ShowDetailsBrief  # List all Scheduled Task tests
```

Logging Infrastructure

Log Collection Architecture

Comprehensive logging is the foundation of detection engineering. The lab implements enterprise-grade log collection with multiple transport mechanisms and centralized aggregation.

Source	Transport	Collector	Destination
Windows Events	WEF (WinRM)	Windows Event Collector	SIEM via Beats/NXLog
Sysmon	Windows Event Log	WEF forwarding	SIEM (dedicated index)
PowerShell	Script Block Logging (4104)	WEF forwarding	SIEM (dedicated index)
Linux auditd	Syslog (TCP/TLS)	rsyslog aggregator	SIEM via Filebeat
Network (Zeek)	JSON logs	Filebeat	SIEM (network index)
Firewall	Syslog	rsyslog	SIEM (firewall index)
EDR Telemetry	Agent -> Cloud/On-prem	EDR console	SIEM via API/Syslog
DNS Queries	Passive DNS tap	dnstap/packetbeat	SIEM (dns index)

SIEM Platform

The detection lab SIEM serves as the central analysis platform for all collected telemetry. Elastic Stack is recommended for its flexibility, community detection rules, and cost-effectiveness in lab environments.

Component	Specification	Purpose
Elasticsearch	3-node cluster, 64GB heap each	Log storage, indexing, search
Kibana	Single instance, 8GB RAM	Visualization, dashboards, detection rules
Logstash	2 instances, 16GB RAM each	Log parsing, enrichment, normalization
Elastic Agent/Beats	On each endpoint	Log shipping, endpoint telemetry
Fleet Server	Single instance	Centralized agent management

Windows Event Collection

Windows Event Forwarding (WEF) centralizes event collection with minimal agent footprint. The lab implements subscription-based forwarding for critical security events.

- Security events: 4624, 4625, 4648, 4672, 4688, 4697, 4698, 4720, 4728, 4732, 4756

- Sysmon events: 1 (Process), 3 (Network), 7 (Image Load), 8 (CreateRemoteThread), 10 (Process Access), 11 (File Create), 12/13/14 (Registry), 22 (DNS Query), 23 (File Delete)
- PowerShell: 4103 (Module Logging), 4104 (Script Block), 4105/4106 (Start/Stop)
- Windows Defender: 1116, 1117 (Malware Detection), 5001 (Tamper)

Sysmon Configuration

Sysmon provides enhanced endpoint visibility beyond native Windows logging. The lab uses SwiftOnSecurity's baseline configuration with additional rules for attack detection.

```
# Deploy Sysmon with detection-focused config
sysmon64.exe -accepteula -i sysmonconfig-export.xml

# Key config sections for attack detection:
# - ProcessCreate: Log all, filter noise (chrome, edge, etc.)
# - NetworkConnect: Log suspicious ports, exclude browsers
# - CreateRemoteThread: Log ALL (critical for injection detection)
# - ProcessAccess: Log LSASS access (TargetImage contains lsass.exe)
# - FileCreate: Log exe/dll/ps1 creation in temp paths
```

Detection Tooling

Endpoint Detection

The lab deploys multiple EDR solutions to compare detection capabilities and understand how different tools surface the same attack techniques.

Tool	Type	Deployment	Key Capabilities
Microsoft Defender for Endpoint	Commercial EDR	All Windows endpoints	Behavioral, cloud analytics, ASR rules
CrowdStrike Falcon	Commercial EDR	Sample endpoints	Kernel telemetry, IOA detection
Elastic Defend	Open Source EDR	All endpoints	Behavioral rules, memory protection
Velociraptor	DFIR/Hunting	Server deployment	Live forensics, VQL hunting, artifact collection
osquery	Endpoint Visibility	All endpoints	SQL-based queries, compliance, inventory

Network Detection

Network-based detection provides visibility into C2 communications, lateral movement, and exfiltration that may bypass endpoint controls.

Tool	Purpose	Deployment	Key Outputs
Zeek (Bro)	Protocol analysis	Network TAP/SPAN	conn.log, dns.log, http.log, ssl.log, files.log
Suricata	Signature IDS/IPS	Inline or TAP	Alerts, EVE JSON, protocol logs
RITA	Beacon detection	Processes Zeek logs	Beaconing scores, DNS tunneling, long connections
JA3/JA4	TLS fingerprinting	Zeek module	Client/server fingerprints for known malware
Arkime (Moloch)	Full PCAP	TAP with large storage	Searchable packet capture, session reconstruction

Detection Rule Frameworks

Standardized detection rule formats enable portability across SIEM platforms and community sharing.

Format	Purpose	Tooling

Sigma	Generic SIEM rules	sigmac, pySigma, Sigma CLI for conversion to platform-specific queries
YARA	File/memory scanning	yara, yara-python, integrated with EDR and sandbox
Snort/Suricata	Network signatures	Direct deployment to Suricata, ET Open/Pro rulesets
KQL	Microsoft Sentinel	Native Azure Sentinel detection rules
SPL	Splunk	Splunk Enterprise Security correlation searches

Detection Engineering Workflow Tools

- Sigma: Universal detection rule format with converters for all major SIEMs
- Detection Lab (Chris Long): Automated lab deployment with pre-configured logging
- Atomic Red Team: Technique execution library for validation testing
- MITRE ATT&CK Navigator: Visual coverage mapping and gap analysis
- DeTT&CT: ATT&CK data source and detection coverage tracking
- Hayabusa: Fast Windows event log analysis and Sigma rule scanning
- Chainsaw: Rapid forensic triage using Sigma and custom rules

Automation Framework

CI/CD Pipeline for Detections

Detection rules should follow software development best practices with version control, automated testing, and deployment pipelines.

Stage	Tool	Purpose
Version Control	Git (GitLab/GitHub)	Track rule changes, review process, branching
Validation	Sigma CLI, yamllint	Syntax checking, schema validation
Testing	Atomic Red Team	Execute technique, verify rule fires
Conversion	pySigma	Convert Sigma to platform-specific format
Deployment	Ansible/API	Push rules to SIEM, enable in production
Monitoring	Detection metrics	Track true/false positive rates

Automated Testing Pipeline

Each detection rule should have associated test cases that automatically validate detection coverage. The pipeline executes techniques and verifies alerts fire within expected timeframes.

```
# Example: Automated detection test workflow


1. Reset lab VMs to clean snapshot
2. Deploy latest detection rules to SIEM
3. Execute Atomic Red Team test (e.g., T1003.001)
4. Wait for log ingestion (30-60 seconds)
5. Query SIEM for expected alert
6. Record result: PASS (alert fired) or FAIL (no alert)
7. Generate coverage report
8. Restore VM snapshot for next test
```

Test Harness Architecture

The automation framework uses Invoke-AtomicTest for technique execution and REST API calls to validate SIEM alerts. Results feed into coverage dashboards.

Component	Implementation	Purpose
Orchestrator	Python/PowerShell scripts	Coordinate test execution across lab
Technique Library	Atomic Red Team	Standardized technique execution
SIEM API Client	Elasticsearch/Splunk SDK	Query for alerts, validate detections

Snapshot Manager	Proxmox/vSphere API	Reset VMs between tests
Results Database	PostgreSQL/SQLite	Track test results over time
Dashboard	Grafana/Custom	Visualize coverage, trends, gaps

Sample Automation Script

```
# PowerShell: Automated detection validation
$technique = 'T1003.001'
$testNumber = 1

# Execute atomic test
Invoke-AtomicTest $technique -TestNumbers $testNumber

# Wait for log ingestion
Start-Sleep -Seconds 45

# Query Elasticsearch for alert
$query = @{
    query = @{
        bool = @{
            must = @(
                @{ match = @{ 'rule.name' = 'LSASS Memory Access' } }
                @{ range = @{ '@timestamp' = @{ gte = 'now-2m' } } }
            )
        }
    }
} | ConvertTo-Json -Depth 10

$result = Invoke-RestMethod -Uri 'http://elasticsearch:9200/alerts/_search'
-Method POST -Body $query -ContentType 'application/json'

if ($result.hits.total.value -gt 0) {
    Write-Host '[PASS] Detection fired for $technique' -ForegroundColor
Green
} else {
    Write-Host '[FAIL] No detection for $technique' -ForegroundColor Red
}
```

Validation Workflows

Detection Rule Lifecycle

Each detection rule progresses through defined stages from initial development to production deployment with continuous validation.

1. Hypothesis: Identify technique, define expected telemetry, draft detection logic
2. Development: Write Sigma rule, validate syntax, test in isolated lab
3. Validation: Execute technique via Atomic Red Team, confirm alert fires
4. Tuning: Adjust thresholds, add exclusions for false positives
5. Documentation: Record coverage, data sources, response playbook
6. Deployment: Push to production SIEM via CI/CD pipeline
7. Monitoring: Track alert volume, false positive rate, analyst feedback
8. Iteration: Refine based on production data, adversary evolution

Coverage Validation Matrix

Systematic validation ensures detection coverage across all ATT&CK techniques in scope. The matrix tracks validation status for each technique.

Status	Definition	Action Required
Not Started	No detection rule exists	Develop hypothesis and rule
In Development	Rule drafted, not validated	Execute validation test
Validated - Lab	Rule fires in lab environment	Deploy to production, monitor
Validated - Production	Rule fires in production, tuned	Ongoing monitoring
Gap - No Data Source	Required telemetry not available	Instrument data source
Gap - Infeasible	Detection not reliable/practical	Document, accept risk

Analyst Training Scenarios

The lab supports structured training scenarios that walk analysts through attack chains with realistic telemetry. Scenarios range from single-technique exercises to full red team simulations.

Scenario	Techniques	Duration	Learning Objectives
Credential Theft 101	T1003, T1558	2 hours	Identify LSASS access, Kerberos attacks in logs
Lateral Movement	T1021, T1550, T1570	4 hours	Trace attacker movement across systems

Ransomware Precursors	T1486, T1490, T1489	4 hours	Detect pre-encryption indicators
Full Kill Chain	All tactics	8 hours	End-to-end attack investigation
Purple Team Exercise	Custom	2 days	Collaborative attack/detect iteration

Implementation Roadmap

Phase 1: Foundation (Weeks 1-4)

Establish core infrastructure, basic networking, and minimal viable logging stack.

- Deploy virtualization platform (Proxmox/ESXi cluster)
- Configure network segmentation (VLANs, firewall rules)
- Build Windows domain (2 DCs, 3 workstations)
- Deploy Elastic Stack (single-node for initial testing)
- Configure Windows Event Forwarding to central collector
- Install Sysmon on all Windows endpoints
- Validate basic log flow end-to-end

Phase 2: Detection Tooling (Weeks 5-8)

Add detection capabilities, network monitoring, and initial rule development.

- Deploy Zeek/Suricata on network TAP
- Install Velociraptor server and agents
- Configure Elastic detection rules (Elastic prebuilt rules)
- Import Sigma rules for common techniques
- Build initial ATT&CK Navigator coverage map
- Test 10 priority techniques with Atomic Red Team

Phase 3: Automation (Weeks 9-12)

Implement CI/CD pipeline, automated testing, and coverage tracking.

- Set up GitLab/GitHub for detection rule version control
- Build automated test harness (PowerShell/Python)
- Integrate Atomic Red Team with CI pipeline
- Create Grafana dashboard for coverage metrics
- Document validation procedures and runbooks
- Train team on lab usage and workflows

Phase 4: Expansion (Ongoing)

Expand coverage, add advanced capabilities, integrate with production workflows.

- Add Linux endpoint telemetry (auditd, osquery)
- Deploy cloud simulation (LocalStack, Azure AD tenant)
- Integrate additional C2 frameworks for diversity

- Build analyst training curriculum
- Establish purple team exercise cadence
- Continuous ATT&CK coverage expansion

Appendix A: Recommended Tool Versions

Tool	Version	Source
Proxmox VE	8.1+	proxmox.com
Windows Server	2022 (21H2)	Microsoft Volume Licensing
Windows 11	23H2	Microsoft Volume Licensing
Ubuntu Server	22.04 LTS	ubuntu.com
Kali Linux	2024.1+	kali.org
Elasticsearch	8.12+	elastic.co
Sysmon	15.0+	Microsoft Sysinternals
Zeek	6.0+	zeek.org
Suricata	7.0+	suricata.io
Velociraptor	0.72+	docs.velociraptor.app
Atomic Red Team	Latest	github.com/redcanaryco/atomic-red-team
Sigma	Latest	github.com/SigmaHQ/sigma

Appendix B: Critical Sysmon Events for Detection

Event ID	Name	Detection Use Cases
1	Process Create	Execution, suspicious processes, command lines
3	Network Connection	C2 callbacks, lateral movement, exfiltration
6	Driver Loaded	Rootkits, vulnerable drivers (BYOVD)
7	Image Loaded	DLL side-loading, unsigned modules
8	CreateRemoteThread	Process injection, shellcode execution
10	Process Access	Credential dumping (LSASS), process tampering
11	File Create	Malware drops, staging, persistence
12/13/14	Registry Events	Persistence, configuration changes
15	FileCreateStreamHash	Alternate data streams, hiding
17/18	Pipe Events	Named pipe C2 (Cobalt Strike), lateral movement
22	DNS Query	DNS C2, domain generation algorithms
23	File Delete	Anti-forensics, ransomware encryption markers
25	Process Tampering	Process hollowing, herpaderping
26	File Delete Logged	Deleted file content capture (with config)

Appendix C: Sigma Rule Template

Standard template for detection rules following SigmaHQ conventions.

```
title: Descriptive Title for Detection
id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx  # Generate UUID
status: experimental  # experimental, test, stable
description: |
    Detailed description of what this rule detects,
    why it matters, and expected adversary behavior.
references:
    - https://attack.mitre.org/techniques/TXXXX/
    - https://relevant-blog-or-research.com
author: Your Name
date: 2026/01/25
modified: 2026/01/25
tags:
    - attack.tactic_name
    - attack.tXXXX.XXX
logsource:
    product: windows
    category: process_creation  # or service: sysmon
detection:
    selection:
        Image|endswith: '\suspicious.exe'
        CommandLine|contains:
            - '-encoded'
            - '-bypass'
    filter_legitimate:
        ParentImage|endswith: '\legitimate_parent.exe'
    condition: selection and not filter_legitimate
falsepositives:
    - Legitimate admin tools
    - Specific software that triggers benign matches
level: high  # informational, low, medium, high, critical
```

Appendix D: Network Architecture Diagram Reference

ASCII representation of detection lab network topology for documentation and quick reference.