Enhancing Cybersecurity Detection Quality: Advanced SOC Techniques and Best Practices

The Rapid Evolution of Cybersecurity: Key Events, Challenges, and Lessons for Today

Major Milestones in Cybersecurity | 2010 - 2017

Cybersecurity grew rapidly and often reactively. Key events during 2010-2017 shaped both perception and practice.

- Stuxnet (2010): Cyber was mostly invisible to the public, used by governments as a precise, sophisticated tool; feared but understood by very few.
- Snowden Leaks (2013): Cyber capabilities exist in governments, with global reach; impact is potentially huge, but execution remains in expert hands.
- WannaCry Ransomware (2017): Cyber attacks hit enterprises worldwide; threat is now professional, financially motivated, and no longer limited to state actors.

This period marks the shift from experimental cyberattacks to real-world business impact, showing why cybersecurity became urgent for organizations worldwide.

Cybersecurity Costs vs. Benefits | Investment Dilemma

Cybersecurity is a significant investment for organizations, often without direct financial return.

- **High Costs**: Tools, platforms, SOC teams, training, insurance
- Limited Direct ROI: Security spends rarely generate revenue
- Marketing & Reputation: Cybersecurity often becomes a **selling point** ("we are secure")
- Strategic Challenge: Balancing risk reduction with budget constraints

Understanding costs and benefits helps organizations make informed security decisions and avoid superficial implementations.

Legal & Regulatory Pressure | When Compliance Creates New Risks

Regulations push companies to "do cybersecurity", but the way compliance is applied today often creates new risks, unnecessary complexity, and even technical weaknesses.

- Misaligned Requirements: Generic controls imposed by compliance may not fit the company's real architecture, sometimes forcing **weak or risky implementations**.
- New Vulnerabilities Introduced: To meet legal expectations, organizations sometimes deploy solutions **without or even against the advice of technical teams**, increasing the attack surface.
- Complexity Explosion: More tools, more processes, more audits → infrastructures become **heavier and harder to secure**, especially for SOC teams.
- Security Degradation: Compliance obligations can **reduce real defensive quality**, diverting

experts from technical priorities to administrative tasks.

- Cost Without Protection: Spending rises, but actual protection doesn't and may even deteriorate.

Compliance helps companies acknowledge cybersecurity, but the **current compliance-driven approach can increase risks**. Defenders must manage these consequences every day.

Consequences of Rapid Cyber Development | Partial & Imperfect Security

Because cybersecurity grew too fast and under pressure, many organizations ended up with **incomplete, inconsistent, or poorly integrated** security measures.

- Partial Implementations: Tools deployed without full configuration or not used to their full potential.
- Strategic Gaps: Security choices made quickly, without long-term vision or alignment with real threats.
- Fragmented Architectures: Layers of solutions added over time → inconsistent coverage and blind spots.
- Operational Overload: SOC teams must defend environments that are complex, noisy, and full of gaps.
- Quality Issues: Many controls exist "on paper" but provide little real protection in practice.

The fast, unstructured development of cybersecurity created fragile foundations today's defenders must navigate these weaknesses while building something stronger for the future.

The Path to Quality Cybersecurity | Learning From the Last 10 Years

Cybersecurity has grown fast, sometimes too fast. But we are now entering a phase where quality, prioritization, and technical mastery become essential.

- Only ~8 years of real investment: Modern cybersecurity budgets started around 2017 → systems are still young, fragmented, and evolving.
- Shift in Mindset: Organizations begin to move from "adding more tools" to identifying what truly matters and improving it properly.
- Effective Defense With Lower Operational Cost: The goal is not to remove tools, but to implement defenses correctly, which naturally reduces complexity, effort, and therefore costs.

- Better Foundations: Architecture, detection, and processes are finally stabilizing and becoming more reliable and sustainable.
- Your Opportunity: You will work in cybersecurity for the next 30-40 years. You'll help build the next generation of defenses, based on quality instead of urgency.
- Course Objective: Go beyond today's SOC practices and give you the technical understanding and defensive mindset required for tomorrow's cyber challenges.

The strength of tomorrow's cybersecurity will come from **skilled people** who know how to build and operate defenses **effectively.**

Inside Real Incidents: How Companies Create Their Own Vulnerabilities

Compliance-Driven Design That Broke Security

1. Large enterprise preparing for ISO 27001 certification.
2. Decision: deploy a "security layer" to control server access management.
3. Technical reality: solution relied on virtual segmentation only (VLAN isolation for a few machine groups).

- The new component could not support encryption in the environment's architecture.
- As a result, all authentication flows, including domain and domain admin passwords, were transmitted in cleartext on the network.
- For more than 3 years, this system ran in production without being integrated into security monitoring.

- Created a massive blind spot: server access invisible to SOC tools.
- Introduced a perfect vector for stealthy persistence if an attacker entered the network.
- Increased operational burden: more patching, more maintenance, more procedures, more attack surface.
- All of this added for compliance, yet it reduced real security.

Even once governance was aware of the vulnerability, no one was tasked with fixing it. The main concern of the ISMS: updating TLS certificates from 1.2 to 1.3 for compliance, not mitigating the actual security risk.

EDR Implementation Misaligned with Actual Needs

1. Year 2020: EDR solutions become standard and widely promoted in the industry.
2. The CISO decides to deploy one based solely on sales numbers and price, without consulting the technical teams.

- By 2023, it became clear that the chosen EDR was not effective for the organization's environment.
- The solution did not meet actual security needs, missing coverage compared to other available options.
- The organization had poor visibility into the SI, and the EDR gave a false sense of security, with detections often misleading or incorrect.

- 2024: after a POC with a better EDR, the organization switches, recovering more than 6

months of work and training that could have been avoided.
- Demonstrates the cost of decision-making without technical input: time, effort, and potential gaps in security during the interim.

Decisions must include technical assessment and alignment with real security needs to avoid wasted effort, false confidence, and increased risk. Choosing tools based on market hype or price alone can be costly.

Core Principles and Structure of a Security Operations Center

What Is a Security Operations Center (SOC)? | Cybersecurity Monitoring & Defense Overview

A Security Operations Center (SOC) is the centralized team responsible for monitoring, detecting, and responding to cybersecurity threats across an organization.

Mission:

- Ensure continuous protection of IT assets
- Identify malicious activity early
- Coordinate incident response and improve security posture

Key Points:

- Acts as the nerve center for cyber defense
- Evolves from basic monitoring to intelligence-driven operations
- Integrates people, processes, and technology to manage risk effectively

SOC Organizational Models | Centralized, Distributed & Outsourced SOC

A SOC can be organized in several ways depending on the organization's size, resources, and security strategy.

Common SOC Models:

- Centralized SOC: Single location monitoring all systems; strong control and coordination
- Distributed SOC: Multiple locations, often across regions; better local coverage
- Virtual SOC: Remote teams leveraging cloud tools; flexible and cost-efficient
- Outsourced SOC (MSSP/MDR): Managed by third-party providers; complements internal capabilities

Key Considerations:

- Trade-offs between cost, coverage, and control
- Organizational fit depends on risk tolerance and regulatory requirements

- Hybrid models often combine internal and external resources for optimal performance

SOC Roles and Responsibilities | Key Cybersecurity Functions

A SOC relies on specialized roles to detect, analyze, and respond to threats efficiently.

- Tier 1 Analyst: Monitors alerts, triages incidents
- Tier 2 Analyst: Investigates and escalates complex incidents
- Tier 3 Analyst / Threat Hunter: Performs deep analysis, proactive threat hunting
- Incident Responder / DFIR Specialist: Leads containment, eradication, recovery
- SOC Manager & Engineers: Coordinate operations, maintain tools and processes

Clear role definition ensures fast, accurate response and reduces **alert fatigue.**

SOC Processes & Workflows | Monitoring, Detection & Response

SOC processes structure how security events are handled from detection to resolution.

Core Workflow Steps:

- Monitoring & Alerting: Continuous surveillance of networks, endpoints, and cloud
- Triage & Investigation: Validate and analyze alerts
- Containment & Eradication: Stop threats and remove them from the environment
- Recovery & Lessons Learned: Restore systems and improve defenses

Well-defined processes and playbooks ensure fast, consistent, and effective incident handling.

Logging & Telemetry Fundamentals | Essential SOC Data Sources

Effective SOC operations depend on high-quality logs and telemetry from all parts of the IT environment.

Key Log Sources:

- Network: Firewalls, IDS/IPS, routers
- Endpoints: EDR/XDR agents, OS logs
- Cloud & Applications: Cloud workloads, SaaS apps, APIs
- Identity & Access: Authentication systems, IAM platforms

Best Practices:

- Ensure log quality, consistency, and retention
- Normalize and enrich logs for better detection
- Prioritize logs that support incident response and compliance

Visibility is the foundation of SOC effectiveness. You can't secure what you can't see.

Detection Engineering Essentials | SOC Threat Detection & Analytics

Detection engineering is the art of turning logs and telemetry into actionable alerts for the SOC.

- Rule-based detection: SIEM correlation rules for known threats
- Anomaly detection & analytics: Identify unusual behavior patterns
- MITRE ATT&CK alignment: Map detections to attacker techniques
- False positives management: Tune alerts to reduce noise

Effective detection requires balance between automation and human insight to spot threats before they cause damage.

SOC Tooling & Architecture Overview | Key Security Technologies

A SOC relies on integrated tools and platforms to detect, investigate, and respond to threats efficiently.

1. EDR/XDR (Endpoint Detection & Response / Extended Detection & Response): Endpoint and cross-layer visibility
2. Network Security Tools: IDS/IPS, NDR, firewalls for real-time network monitoring
3. Threat Intelligence Platforms (TIP): Enrich alerts with actionable threat data
4. SOAR (Security Orchestration, Automation, and Response): Automates repetitive tasks and workflows

Integrated tooling ensures faster detection, coordinated response, and continuous improvement of security operations.

Threat Intelligence in the SOC | Using TI for Proactive Defense

Threat intelligence (TI) helps the SOC anticipate attacks and enrich alerts with context.

- Tactical TI: Indicators of compromise (IOCs) for immediate response
- Operational TI: Understanding attacker methods and campaigns
- Strategic TI: Trends, risks, and long-term threat planning
- Integration: Feed TI into SIEM, SOAR, and detection rules

Actionable threat intelligence turns raw data into informed decisions, improving detection and response efficiency.

Incident Response Foundations | SOC Incident Management

Incident response ensures organized, fast, and effective handling of security events.

- Identification & Classification: Detect and categorize incidents by severity
- Containment & Eradication: Stop the threat and remove its impact
- Recovery: Restore systems and data safely
- Lessons Learned: Update playbooks, rules, and processes

A structured incident response reduces damage, accelerates recovery, and strengthens future SOC defenses.

SOC Challenges & Best Practices | Improving Security Operations

SOC teams face common challenges that impact efficiency and effectiveness.

Key Challenges:

- Alert fatigue: High volume of false positives
- Coverage gaps: Missing visibility across systems or cloud services
- Resource constraints: Limited skilled personnel and time

Best Practices:

- Automation & Orchestration: Reduce manual work and response time
- Continuous Improvement: Use metrics, purple teaming, and lessons learned
- Prioritization & Intelligence: Focus on high-risk threats and actionable alerts

A well-optimized SOC balances technology, processes, and people to stay ahead of evolving cyber threats.

Core Security Tools: Capabilities, Limits & SOC Impact

Firewalls in Cybersecurity: Benefits, Limits, and Common Misconceptions

Strengths

- Centralized control of network traffic (inbound and outbound).
- Can filter based on IP, port, protocol, and application rules.
- Helps protect against unauthorized external access.
- Useful for network segmentation to reduce attack surface.

Weaknesses

- Cannot detect malicious activity already inside allowed traffic.
- Misconfigurations can create false positives or block legitimate traffic.
- Managing complex rules can be challenging in dynamic or cloud environments.

Firewalls in Cybersecurity: Benefits, Limits, and Common Misconceptions

Limits

- Only sees traffic that passes through it; cannot see endpoint behavior or encrypted payloads.
- Cannot prevent attacks coming from inside the network.
- Security depends on rule quality and timely updates.

Incorrect Assumption

- Believing that a firewall can stop all attacks, including internal or insider threats.
- Instead of relying on a firewall to detect or block malware uploads.

Antivirus & Endpoint Protection (EPP): Benefits, Limits, and Common Misconceptions

Strengths

- Protects endpoints from known malware and viruses.
- Uses heuristic and behavioral analysis to detect unknown threats.
- Centralized management for updates and policy enforcement.
- Can block malicious files before they execute and provide endpoint visibility.
- Relatively low false positives, reducing unnecessary alerts.

Weaknesses

- Limited effectiveness against advanced persistent threats (APT) or targeted attacks.
- Can impact endpoint performance if misconfigured.
- Requires constant updates and tuning to maintain high detection rates.
- Limited granularity in configuration, restricting fine-tuned control.

Limits

- Cannot guarantee detection of sophisticated attacks or in-memory exploits.
- Cannot prevent lateral movement across the network on its own.
- Effectiveness depends on correct configuration and policy enforcement.
- Struggles to detect malicious activity when attackers use legitimate admin tools or system binaries (LOLBins).

Intended Use

- Basic malware protection for workstations and servers.

Incorrect Assumption

- Believing it can stop all attacks, including sophisticated, targeted, or fileless malware.

IDS & IPS in Cybersecurity: Detection, Prevention, Limits and SOC Essentials

Strengths

- Detects suspicious or malicious network patterns (signatures, anomalies).
- Provides visibility on attacks such as scans, exploits, protocol abuses.
- IPS can actively block known malicious traffic.
- Helps detect attacks bypassing basic firewall filtering.

Weaknesses

- Signature-based detection misses unknown or obfuscated attacks.
- Encrypted traffic drastically reduces visibility.

- Limited efficiency against advanced or lateral-movement techniques.

IDS & IPS in Cybersecurity: Detection, Prevention, Limits and SOC Essentials

Limits

- Cannot see internal traffic or workstation-to-workstation communications.

Incorrect Assumption

- Considering IDS a source of complete visibility, including internal traffic.
- Treating IPS as a "set and forget" universal protection.

Intended Use

- Detecting exploit attempts, scans, brute-force attacks, protocol anomalies.
- Monitoring exposed zones (DMZ, internet-facing applications).
- Adding deep-inspection capabilities on top of firewall rules.

- Tracking repeated malicious connections or external probing.

Endpoint Detection & Response (EDR): Advanced Threat Detection, Response & SOC Integration

Strengths

- Continuous endpoint monitoring: processes, files, registry, network connections.
- Detects suspicious behavior, including fileless malware, LOLBins, and lateral movement.
- Provides detailed forensic data for incident investigation and root-cause analysis.
- Supports automated or semi-automated containment (quarantine, kill process, isolate device).

Weaknesses

- Requires agent deployment on endpoints; coverage gaps if agents are missing or disabled.
- Resource-intensive: may impact performance
- Advanced attackers can bypass EDR

- Dependent on proper tuning: too loose → missed threats, too strict → false positives.
- High alert volume

Endpoint Detection & Response (EDR): Advanced Threat Detection, Response & SOC Integration

Limits

- Limited visibility on network attacks
- Detects suspicious activity after it occurs, so potential impact may have already happened.

Incorrect Assumption

- Detecting unknown malware, ransomware, and lateral movement within the network.
- Investigating alerts and providing detailed forensic evidence for incidents.
- Containing compromised endpoints quickly to prevent further spread.

Network Detection & Response (NDR): Monitoring, Threat Detection & SOC Insights

Strengths

- Provides network-wide visibility, including east-west traffic and encrypted flows.
- Detects anomalies, lateral movement, command-and-control activity, and unusual protocol behavior.
- Complements EDR by detecting threats that don't execute on endpoints.
- Supports investigation with detailed network telemetry.

Limits

- Limited detection of attacks inside endpoints before network activity occurs.
- Encrypted traffic may reduce full visibility without TLS inspection.

- May not provide actionable response beyond alerting or network quarantine.

Network Detection & Response (NDR): Monitoring, Threat Detection & SOC Insights

Weaknesses

- Blind spots exist if NDR sensors are not deployed on all critical network segments.
- Provides post-factum detection: alerts often after malicious network activity is underway.

Intended Use

- Detecting lateral movement, suspicious beaconing, and command-and-control traffic.
- Monitoring network segments where endpoint coverage is limited or missing.
- Providing additional context for SOC analysts during threat hunts.

Security Information & Event Management (SIEM): Centralized Monitoring, Correlation & Threat Detection

Strengths

- Centralizes and normalizes logs from all IT and security systems.
- Supports dashboards, alerting, and reporting for SOC analysts.
- Enables historical analysis and forensic investigation of security incidents.
- Supports event correlation rules to identify patterns across multiple sources.

Weaknesses

- Typically not real-time; alerting depends on log collection interval and processing.
- High alert volume can overwhelm SOC analysts if correlation rules are poorly tuned.
- Cannot block or prevent attacks; purely detective.

- Complex deployment and maintenance, especially in dynamic, complex or large environments.

Security Information & Event Management (SIEM): Centralized Monitoring, Correlation & Threat Detection

Limits

- Alerts are only as good as the logs collected; missing logs → blind spots.
- Limited by latency: attacks can progress before detection due to log collection intervals.
- SOC must implement correlation rules manually; achieving meaningful detection often takes years of tuning and refinement.
- Must manage alert fatigue and group related alerts when the SIEM produces high volumes or low-quality signals.

Incorrect Assumption

- Thinking SIEM replaces endpoint or network monitoring; it only correlates the data already collected.
- Expecting SIEM to block attacks, even when integrated with SOAR.

Intended Use

- Centralized logging and reporting for SOC visibility.
- Identifying patterns and trends across multiple sources over time.
- Investigating security incidents and reconstructing timelines.

Threat Intelligence Platform (TIP): Collect, Analyze & Operationalize Threat Data

Strengths

- Centralizes threat intelligence from multiple sources: feeds, open source, commercial, internal indicators.
- Normalizes and enriches threat data for SOC consumption.
- Provides context for alerts: IOCs (IPs, domains, URLs, file hashes, TTPs).
- Facilitates sharing intelligence across teams and trusted partners.

Weaknesses

- Only as good as the quality and relevance of threat feeds.
- Risk of irrelevant or outdated indicators being acted on if not managed carefully.

- Unknown IOCs or threats that change indicators will not be detected.

Threat Intelligence Platform (TIP): Collect, Analyze & Operationalize Threat Data

Intended Use

- IOCs based detections
- Providing contextual information for alerts.
- Conducting threat hunting based on known attacker TTPs.

WAF (Web Application Firewall): Protecting Web Applications from Attacks

Strengths

- Filters and blocks HTTP/HTTPS attacks.
- Protects exposed web applications from common exploitation attempts.
- Useful for virtual patching when the application cannot be fixed immediately.

Weaknesses

- Ineffective against logic flaws.
- Can be bypassed with obfuscation or non-standard payloads.
- Not useful for detecting attacks on internal services

WAF (Web Application Firewall): Protecting Web Applications from Attacks

Incorrect Assumption

- Deploying it as a replacement for secure development or code reviews.

Intended Use

- Protecting web applications from common attack vectors (OWASP)
- Providing additional telemetry for SOC investigations.

Data Loss Prevention (DLP): Protecting Sensitive Data on Endpoint & Network

Strengths

- Monitors and controls sensitive data movement on endpoints, network, and cloud.
- Detects and block unauthorized data transfers: emails, USB devices, file uploads, cloud storage.
- Provides audit logs for investigations and reporting.

Weaknesses

- May generate false positives if policies are too broad.
- Limited effectiveness against encrypted traffic or obfuscated data.
- Can be bypassed by sophisticated users.

Data Loss Prevention (DLP): Protecting Sensitive Data on Endpoint & Network

Limits

- Only detects actions defined in policies; unknown data types or formats may pass undetected.
- Endpoint DLP relies on agents; if agents are disabled or circumvented, protection is lost.
- Network DLP requires proper placement to monitor traffic; blind spots can exist.

Incorrect Assumption

- Assuming DLP replaces user awareness and good practices; it's a technical control, not a human behavior fix.

Intended Use

- Preventing accidental or malicious exfiltration of sensitive data.

- Supporting SOC investigations into data exfiltration events.

Extended Detection & Response (XDR): One Tool to Rule Them All ?

- XDR aims to unify EDR, NDR, DLP, SIEM, TIP, and more under a single platform.
- Designed to simplify SOC operations and provide centralized visibility.
- Reality: most XDR solutions today attempt to do everything but do each component partially.
- Can lead to superficial detection, alert fatigue, and gaps in coverage if relied on as a standalone solution.
- Effective use requires still relying on specialized tools and skilled analysts.

UEBA (User & Entity Behavior Analytics): Detecting Anomalies Through Behavior

- UEBA analyzes user and entity behavior patterns to detect unusual activity.
- Useful for spotting insider threats, compromised accounts, and lateral movement.
- Reality check: UEBA can produce false positives, needs time to learn "normal" behavior, and is not effective on its own.

Additional Detection Tools: Specialized Sensors Across Hosts, Network & Cloud

- HIDS (Host Intrusion Detection System): monitors host-level events: file integrity, logs, configuration changes.
- NIDS (Network Intrusion Detection System): inspects network packets to detect known signatures or suspicious patterns.
- NTA (Network Traffic Analysis): focuses on behavioral and statistical analysis of network flows.
- Wireless IDS (WIDS): monitors Wi-Fi activity for rogue access points, suspicious clients, and RF-based attacks.
- Email Security Gateways: filters phishing, malware, spoofed domains, malicious attachments.
- Secure Web Gateway (SWG) / Proxy Monitoring: inspects outbound web traffic: URL filtering, file inspection, SSL/TLS inspection.

SOAR Platforms for SOC Operations: Orchestration & Automated Response

Strengths

- Designed to automate SOC workflows, enrich alerts, and execute response playbooks.
- Helps SOC teams reduce manual workload, accelerate triage, and standardize responses.
- Extremely useful for repetitive tasks: enrichment, blocking IPs, disabling accounts, notifications.
- Reduces MTTR (Mean Time To Respond) when properly implemented.
- Automates routine SOC tasks → frees analyst time for real investigations.

Weaknesses

- Requires high-quality detection upstream → no good input, no good output.

- Complex to integrate across heterogeneous environments.
- Requires constant maintenance of playbooks, connectors, API keys, and workflows, the SOC must manually integrate every input and output tool.
- Risk of over-automation → blocking legitimate users or systems if rules are too aggressive.

SOAR Platforms for SOC Operations: Orchestration & Automated Response

Incorrect Assumption

- Deploying SOAR as a "magic fix" for alert fatigue instead of fixing root causes.
- Thinking SOAR will improve detection quality.
- Believing SOAR can correlate alerts or improve detection logic, it does not analyze or correlate signals; it only automates actions based on existing alerts.

Intended Use

- Alert enrichment (WHOIS, TI lookup, sandboxing).
- Automated containment (isolate host, revoke tokens, disable user).

- Workflow automation (ticket creation, notifications, triage templates).

Cybersecurity Vulnerability Scanning: Tools for Exposure and Risk Identification

Strengths

- Scan systems, applications, configurations, and software dependenciesfor known vulnerabilities.
- Provide risk scoring (CVSS, EPSS), remediation guidance, and asset visibility.
- Essential for continuous hardening, hygiene, and reducing attack surface.
- Primarily used by internal SOC / internal security teams.
- Scanners identify potential weaknesses, not ongoing exploitation.
- Helps prioritize patching based on severity and business impact.

Cybersecurity Vulnerability Scanning: Tools for Exposure and Risk Identification

Limits

- Limited against zero-days, logic flaws, or application-specific bugs.

Intended Use

- Maintaining a vulnerability management program.
- Tracking remediation progress across teams.

Incorrect Assumption

- Assuming scanning alone improves security, without patching, scans mean nothing.

Detection Tool Internals: Exploring Mechanisms and Techniques

Hashing and Fuzzy Hashes: Core File-Based Detection

- Cryptographic Hashes: MD5, SHA-1, SHA-256 for identifying known files
- Format-Specific Hashes: imphash (import hash) for PE files to detect variants: bypass by adding Win32 API functions usages
- Fuzzy Hashes: SSDEEP, TLSH to identify similar or slightly modified files: bypass by adding data or padding (length based)
- Use Case: Quickly verify file integrity and detect known malware

Hashing is a foundational detection mechanism, but alone it cannot catch modern malware variants, complementary heuristics and behavioral monitoring are required.

Heuristic Analysis: Detecting Suspicious File Patterns

- File Metadata Analysis: Inspect PE (Windows) and ELF (Linux) executables, timestamps, imports/exports, embedded macros: potential bypass with headers modifications
- Text/File Parsing: Analyze PDFs (JavaScript), Office files (VBA), scripts for suspicious patterns: potential bypass with encoding
- Entropy Analysis: Detect compressed or encrypted content that may hide malware: bypass with encoding
- Use Case: Identify previously unknown malware or modified files that evade signature-based detection
- Limitations: May generate false positives; cannot always pinpoint precise threat origin

Heuristic analysis strengthens detection by examining file structure, content, and entropy, allowing the discovery of hidden or modified malware beyond simple hash checks.

Signature Languages: Regex & Yara for Malware Detection

- Regex: Pattern matching and text parsing for suspicious strings, macros, or script content
- Yara: Advanced pattern matching tool; can scan files, memory, and processes for complex malware indicators
- Power of Yara: Supports modular rules, conditional logic, and metadata; widely used in EDR and threat hunting
- Limitation: Purely signature-based

Yara extends simple regex with powerful, flexible rules that allow detection of sophisticated malware, including in-memory threats, but still relies on pre-defined signatures.

File Monitoring: fanotify, inotify, and Minifilter Drivers

- inotify (Linux):
  - Monitors filesystem events on specific files or directories
  - Non-blocking, low overhead, no root required
  - Based on VFS, ideal for general file monitoring
- fanotify (Linux):
  - Can block or allow access to files, requires root privileges
  - Higher overhead, monitors entire mounts
  - Suitable for security enforcement where blocking is needed
- Minifilter Drivers (Windows):
  - Kernel-level monitoring of file operations
  - Requires signed kernel module

    - Tracks disk actions like create, read, write, delete

File monitoring provides granular visibility into filesystem-level operations, differentiating between non-blocking observers (inotify/ETW) and blocking enforcement (fanotify/Minifilter).

Linux System Interfaces: ProcFS, SysFS, Netlink, and eBPF for Kernel-Level Monitoring

- ProcFS: Provides access to process information and system statistics from user space
- SysFS: Exposes kernel objects and hardware info for monitoring and configuration
- Netlink: Kernel-to-user communication for networking events and monitoring
- eBPF: Executes custom programs safely in the kernel for event tracing, monitoring, and security enforcement

Linux system interfaces enable deep visibility into processes, hardware, and kernel events, forming the foundation for advanced detection tools without modifying the kernel.

Windows Event Monitoring: ETW, ETW-TI, and WMI for SOC Visibility

- ETW (Event Tracing for Windows): Kernel and application-level event logging for detailed system activity
- ETW-TI: Threat Intelligence integration with ETW events for enriched detection
- WMI (Windows Management Instrumentation):
  - Queries system and application data; relies on ETW underneath
  - Very simple to attach callbacks
  - Deeply integrated with Windows, .NET, and PowerShell
- Key Usage: Collects detailed telemetry for monitoring processes, user actions, and system changes

Windows event monitoring provides SOC teams with granular visibility into system and application behavior, forming the backbone of detection and correlation in EDR and SIEM solutions.

AMSI: Antimalware Scan Interface for Runtime Malware Detection

- What is AMSI: Windows API that allows applications and scripts to request malware scanning from the system's antivirus engine
- Language Agnostic: Any language can call AMSI (PowerShell, VBScript, .NET, etc.), its userland API (bypass by memory patching)
- How to use:

1. The runtime tokenizes, parses, builds AST, and resolves aliases
2. After preprocessing, the language requests a scan from AMSI
3. Scans can occur on execution, on new block definitions, or on-demand

- Integration: Provides runtime protection without modifying the underlying antivirus engine

AMSI enables runtime malware scanning by allowing languages to submit preprocessed code for detection, bridging scripts and antivirus engines for enhanced SOC visibility.

Logging Techniques for Detection: /var/log, ETW, Sysmon, and auditd

- Linux Logging (/var/log):
  - Most logs are line-by-line text files, read incrementally by process
  - Binary logs exist (journalctl, wtmp/utmp/btmp) and are harder to parse
  - Tools like auditd provide structured, configurable logging for security monitoring
- Windows Logging (ETW):
  - Event Tracing for Windows enables real-time event registration, more reliable than parsing .evtx files
  - Allows detection of system, application, and security events
  - Sysmon enhances ETW with detailed process, network, and file activity logs, easier to parse and configure than raw ETW

- Sysmon (Windows): Extended system monitoring, logs process creation, network connections, and file changes

These logging mechanisms provide the raw event streams that detection tools consume; understanding their format, access method, and incremental update behavior is essential to implement detection at the system level.

Sigma & Rule Languages for Detection

- Core Idea: Standardized, human-readable detection rules across platforms
- Purpose: Translate detection logic into a universal format usable on multiple SIEMs and log sources
- Mechanism:
    - Convert log events into structured patterns
    - Translate platform-specific log fields into a normalized schema
    - Rules can be shared and reused across environments
- Limitations:
    - Requires mapping log fields to a standardized schema
    - Effectiveness limited by the quality and richness of log data available

Sigma enables structured detection across diverse log sources, but mapping and log completeness are critical for accurate detection.

Kernel Modules & Syscall Tracing (Linux) & ELAM (Windows)

- Core Idea: Deep visibility by intercepting system activity directly in the kernel
- Linux Kernel Modules for Detection:
  - Implement syscall hooking or syscall table redirection to monitor process, file, and network actions
  - Very difficult for malware to kill or bypass when properly protected
- Windows ELAM (Early Launch Anti-Malware):
  - Kernel-level anti-malware driver loaded before all other drivers
  - Ensures malicious kernel modules are blocked before initialization
  - ELAM drivers must be signed by Microsoft
- Windows Drivers:

    - Kernel callbacks (process, image, thread)
    - Registry callbacks
    - Minifilters
    - Object Callbacks (HANDLE)
    - ETW
    - WFP Callouts (Windows Filtering Platform, network)

- Kernel faults → system crashes
- Requires correct signing (Windows)

Kernel-level monitoring (Linux modules, Windows ELAM) provides the strongest and earliest detection capability, resistant to userland bypass techniques, but comes with high complexity and strict signing requirements.

Advanced Threat Intelligence Formats (MISP / STIX) for SOC Automation

- Role: Structured representation of IOCs, TTPs, relationships, campaigns, and infrastructures; seamless integration into TIP/SIEM/EDR/SOAR/CTI workflows.
- MISP: Object-based model (Events/Attributes), optimized for high-volume IOC ingestion, normalization, scoring, and operational distribution.
- STIX 2.x: JSON cyber-object model, optimized for graph-based context, linking indicators, behaviors, and threat activity, preferred by hunters for relational visibility.
- Limits: Highly perishable IOCs (hash/IP/domain churn), limited coverage of unknown/emerging threats, and significant maintenance effort (cleaning, deduplication, validation).

MISP optimizes IOC ingestion, while STIX provides a relationship-driven graph model ideal for hunting and contextual analysis.

Advanced Email Authentication Mechanisms (SPF / DKIM / DMARC)

- SPF: DNS-based validation of authorized sending IPs; prevents basic envelope spoofing but fragile with forwarding.
- DKIM: Cryptographic signing of headers/body; ensures integrity and domain-level authenticity, unaffected by relays.
- DMARC: Alignment check (SPF/DKIM ↔ From header) + policy enforcement (none/quarantine/reject) + feedback loops.
- Limits: No protection against internal phishing, compromised accounts, or malicious content (attachments/links) bypassing authentication controls.

SPF and DKIM provide authentication signals, while DMARC enforces policy and alignment, but none mitigate internal phishing or payload-based attacks.

SOC Detection Engineering Methodology: Building High-Quality Detection Rules

Understanding the Attack Scenario

Why this step exists

- Detection priorities come from Governance (risk, policy, compliance).
- Detection topics come from CTI (what attackers use).
- Both define **what must be covered**, not **how the attack actually occurs**.

SOC challenges

- CTI knows *what attackers use* (TTP, tools), but not *how attackers actually implement the attack*.
- Governance imposes requirements without technical context.
- SOC often writes rules without a complete scenario → weak, noisy, or blind detections.

Methodology: Read available documentation, Answer core questions: *Where does this scenario occur in the attack flow ? Why would the attacker perform this action ? At which stage of the attack does it appear ?* Use this understanding to build a clear mental model before writing any detection rule.

Selecting the Right Detection Tool

Why this step exists

Each detection approach has different constraints:

- Some do not work for certain scenarios
- Some are too limited
- Others lack efficiency or responsiveness
- Some take significantly longer to implement

SOC challenges

- Many SOC teams choose a method by habit or urgency rather than by suitability.
- This leads to using approaches that don't match the scenario's needs, causing delays, noise, or blind spots.

Methodology: Choose the detection method that makes the most sense for the scenario: the one that offers the right visibility, the best long-term efficiency, and the lowest operational cost.

Writing the Initial Detection Rule

Why this step exists

- The first version of a rule is not meant to be perfect, it's meant to translate the scenario into an initial detection hypothesis.
- This draft reveals what is missing, unclear, or unrealistic before deeper refinement.

SOC challenges

- Many SOC teams write rules without a clear scenario, leading to noisy, incomplete, or irrelevant detections.
- Others copy existing rules without understanding their assumptions, creating misaligned or fragile detections.

Methodology: Start with a simple, scenario-based rule to capture the core behavior you aim to detect. As a first step, you can rely on public rules or AI-generated rules to identify key elements and outline the scenario before writing the final detection rule.

Identifying Potential False Positives

Why this step exists

- A detection rule is useless if it triggers constantly on legitimate activity.
- Identifying potential false positives early prevents writing rules that will overwhelm analysts and never reach production.

SOC challenges

- Lack of business context leads SOC teams to misinterpret normal behaviors as malicious.
- Time pressure often prevents analysts from deeply understanding how a TTP works at a low level, and from imagining potential false positives before creating or testing the rule.

Methodology: Understand the draft rule thoroughly to identify all known, legitimate system behaviors that could match it. Avoid "tuning" or adjusting a rule that is still conceptually flawed, the draft stage is meant to challenge assumptions, not to prematurely optimize.

Distinguishing True Positives from False Positives

Why this step exists

- To design an accurate detection rule, you must clearly understand what makes the malicious behavior different from legitimate activity.
- Without this step, the final rule remains either too broad.

SOC challenges

- SOC teams often observe matches without fully understanding the execution logic behind them.
- Analysts may assume "it matches = it's good detection"

Methodology: Break down the malicious scenario into its fundamental behaviors and conditions. Break down the legitimate behaviors that also trigger the draft rule. Compare both sides to extract clear discriminators.

Identifying Bypass Techniques

Why this step exists

- A detection rule must account for the ways an attacker could modify their behavior to avoid being detected.
- Some detection concepts have structural limitations, and documenting them is essential for Incident Response and Governance to properly assess risk and prioritize future improvements.

SOC challenges

- SOC teams often evaluate rules only against the "expected" attack, not against variants or stealthier executions.
- Many assumptions are made about how an attacker "should" behave, leading to blind spots.

Methodology: Break the draft rule into all its individual parameters and conditions. For each parameter and condition, determine whether an attacker could alter it while still performing a malicious action.

Identifying Missing Scenarios

Why this step exists

- A solid detection must cover all realistic ways an attacker could achieve the targeted behavior
- Missing scenarios create blind spots that attackers can exploit without triggering any alert.

SOC challenges

- Analysts often rely only on previously seen attacks or existing rules, leading to a fixed and narrow view of the scenario.
- Lack of creativity and limited technical exploration prevent identification of alternative paths.
- Without actively searching for missing variants, many angles remain undetected.

Methodology: Start from the target behavior, identify all possible ways to reach that same behavior. Break down the underlying technology, protocol, or method to understand every valid interaction or workflow it allows.

Rewriting the Optimized Rule

Why this step exists

- After understanding false positives, true positives, bypasses, and missing scenarios, the initial draft rule must be redesigned from scratch.
- The goal is to build a clean, reliable, and conceptually sound detection, not a tuned version of the flawed draft.

SOC challenges

- Many SOC teams adjust the original rule instead of rebuilding a proper detection logic, keeping its weaknesses.
- Without integrating all previous findings, the final rule remains incomplete, fragile, or limited.

- Pressure to "deploy fast" often results in rules that only partially cover the scenario.

Methodology

- Start a new detection design, based on the discriminators identified earlier (true vs. false positives).
- Integrate all insights from the bypass analysis: ensure the rule covers attacker behaviors that cannot be easily altered.
- Incorporate missing scenarios or create a rule for all valid paths leading to the malicious outcome.

Testing and Validating the Detection Rule

Why this step exists

- A rule is only reliable if it consistently works over time.
- Testing ensures correctness, stability, and maintainability, even when the SI or the rule evolves.

SOC challenges

- Many SOC teams deploy rules without proper testing, leading to silent failures.
- Manual testing is incomplete, non-repeatable, and quickly abandoned.
- Changes in the infrastructure often break rules without anyone noticing.

Methodology: Create unit tests for every part of the logic: test each condition individually (each AND, each OR, and all combinations) with intentionally crafted logs. Create functional tests to validate the full scenario: replay or simulate the malicious behavior in the real SI, verify end-to-end detection, from log ingestion to alert generation, run these functional tests every 3 months to ensure the rule still works.

Documentation Requirements for High-Quality Detection Rules

4. **Identifying Potential False Positives:** Document all legitimate behaviors that may trigger the draft rule, it's required for rule redesign, IR awareness, and operational risk assessment.
5. **Distinguishing True vs. False Positives:** Document the discriminators, it's required to understand the final rule logic.
6. **Identifying Bypass Techniques:** Document all bypass possibilities and conceptual limitations it's critical for Governance and IR to understand residual risks and plan future improvements.
7. **Identifying Missing Scenarios:** Document all alternative paths leading to the same malicious outcome to ensures the final rule covers all relevant variants and avoids blind spots.
9. **Testing the Rule** Document unit tests, functional tests, and validation results to ensures repeatability, long-term reliability, and operational trust in the rule.

Summary of the Detection Engineering Methodology

1. Understand the Attack Scenario: Build a clear picture of the attacker's goal, context, and behavior.
2. Select the Appropriate Detection Method: Choose the approach that offers the right visibility and long-term efficiency.
3. Write the Initial Draft Rule: Create a hypothesis-based rule to explore the scenario.
4. Identify Potential False Positives: Map all legitimate behaviors that may incorrectly trigger detection.
5. Distinguish True vs. False Positives: Extract the discriminators that differentiate malicious from legitimate activity.
6. Identify Bypass Techniques: Analyze how attackers could alter their behavior to avoid detection.

7. Identify Missing Scenarios: Explore all possible paths leading to the same malicious outcome.
8. Rewrite the Optimized Rule: Rebuild a clean, accurate rule using all insights gathered.
9. Test and Validate the Rule: Use automated unit and functional tests to ensure reliability over time.

Practical Example: How a Poorly Designed Detection Rule Fails

From Attack Scenario to a Faulty Draft Rule (Typical SOC Mistake)

SOC teams are asked to detect "PowerShell executing malware on the network."

Typical SOC workflow (too fast, too shallow):

- Immediate choice: use the SIEM.
- Immediate action: write a rule.
- No deep analysis of the scenario or PowerShell behavior.

```
powershell.exe AND (
(Invoke-WebRequest AND Start-Process)
OR
-EncodedCommand
)
```

Many SOCs produce weak rules because they rush into writing them without understanding the scenario or its technical diversity.

Why This Detection Rule Completely Fails (FP, Bypass, Missing Scenarios)

False Positives (Operational Noise): Hypervisors using PowerShell with -EncodedCommand to query WMI. **Bypass Techniques (Attacker Variations Within the Same Behavior)**

- Using PowerShell aliases like iwr.
- Using LoLBins inside PowerShell (bitsadmin, certutil)
- Re-implementing HTTP through a socket in PowerShell.

Missing Scenarios (Different Attack Paths Completely Ignored)

- Malware format without file written: Shellcode (execution in memory), PowerShell (IEX), C# (Add-Type)
- Using another protocol such as an SMB mount.

- Injecting the payload through PowerShell STDIN, leaving no command-line trace.

Evaluation Criteria for Detection Rule Assignments

Core Evaluation

Tool Choice:

- Justify why the selected detection tool or approach was used.
- Consider efficiency, visibility, long-term maintainability.

Draft Rule Creation:

- Correctly implement a basic detection rule.
- Show understanding of the scenario.

Identification of False Positives:

- Detect legitimate activities that trigger the rule.
- Explain reasoning and provide examples.

Advanced Evaluation

Differentiating True Positives vs. False Positives (if applicable): Demonstrate ability to distinguish between legitimate matches and malicious matches.

Identification of Potential Bypasses or Alternative Scenarios (if applicable): Document how attackers could evade the rule.

Quality of Final Rule:

- Rule minimizes or eliminates false positives and false negatives.
- Consider coverage of all identified scenarios and correct design logic.

Optional / Bonus Points

Implementation of Automated Tests:

- Unit tests for each condition (AND, OR, keywords).
- Functional tests simulating the attack scenario.

Extra Bonus

- How to block this type of attack in a SOC context.
- How to perform an incident response for the detected attack.

Exam Logistics and Evaluation Process

Exam Workflow Overview

Before the Exam

- Students know the required setup in advance.
- Each group must prepare one VM (Linux or Windows depending on the scenario assigned).

Day 1 - Scenario Delivery & Rule Engineering

1. Morning: each group receives its attack scenario.
2. All day:
   - Deploy and configure their detection tool.
   - Write the initial detection rule.
   - Identify false positives and false negatives.
   - Produce the final corrected rule.

3. Evening: submit the report (one per group) with final rule and bullet-point reasoning for each methodological step

Evaluation & Oral Defense

Day 2 - Instructor Testing (Group Score Base)

- Instructor runs tests on each group's infrastructure: common false positives and crafted false negatives
- These tests establish the baseline score for the group.
- This evaluates the quality and robustness of the final rule.

Day 3 - Oral Examination (Individual Evaluation)

- Each student is examined individually, even though rule creation was done in groups.
- Objectives of the oral evaluation:
    - Verify understanding of every step of the 9-stage methodology.

- Assess ability to justify decisions: why this tool ? why this logic ? why this improvement ?
- Confirm that each member contributed and understood the final rule.