INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-14 18:30 UTC

BYOVD Ecosystem Grows: EDR Killers Expand Tooling and Accessibility, Demanding Kernel-Level Defense Upgrades

SECURITY ANALYSIS | HIGH | CVSS 7.5

SCC Item ID	SCC-STY-2026-0060
Type	Security Analysis
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Endpoint Detection and Response (EDR) solutions broadly, no specific vendor versions identified in available source excerpt; Windows kernel-dependent deployments most directly at risk
Published	2026-04-14T16:20:34
Discovery Source	Rss

Executive Summary

The bring-your-own-vulnerable-driver (BYOVD) ecosystem is maturing rapidly, with EDR-killing tooling becoming more accessible and commoditized across both ransomware groups and advanced persistent threat actors. By loading legitimately signed but vulnerable Windows kernel drivers, attackers can blind or terminate endpoint detection and response agents before any malicious activity is flagged, effectively neutralizing a core layer of enterprise defense. This trend signals that EDR coverage alone is no longer sufficient; organizations that have not implemented kernel-level integrity controls face a growing capability gap against a widening pool of adversaries.

Technical Analysis

BYOVD attacks exploit a fundamental trust asymmetry in the Windows driver model: legitimately signed drivers are permitted to operate in kernel space, and attackers have learned to weaponize this trust by loading known-vulnerable drivers to execute privileged code outside the visibility of user-mode security agents. Once a vulnerable driver is loaded, an attacker can issue direct kernel object manipulation calls to terminate EDR processes, unload kernel callbacks the EDR relies on for telemetry, or blind the agent entirely, clearing the path for ransomware deployment, lateral movement, or data exfiltration.

What distinguishes the current landscape from earlier BYOVD activity is scale and accessibility. Tooling that once required custom development or access to closed threat actor communities is now increasingly available in

commoditized form, lowering the skill threshold required to execute a successful EDR-kill operation. Ransomware operators in particular have incorporated BYOVD-based EDR killers as a pre-encryption step, ensuring defenses are disabled before payload delivery. APT actors have similarly adopted the technique for stealth and persistence.

The relevant MITRE ATT&CK techniques map the full attack chain: T1068 (Exploitation for Privilege Escalation) and T1562.001 (Impair Defenses: Disable or Modify Tools) describe the core capability; T1014 (Rootkit) and T1543.003 (Create or Modify System Process: Windows Service) reflect persistence and stealth mechanisms used alongside the driver abuse; T1211 (Exploitation for Defense Evasion) captures the evasion goal; and T1195 (Supply Chain Compromise) surfaces in cases where vulnerable drivers enter environments through legitimate software supply chains rather than direct attacker delivery.

Defensive gaps being exploited include: over-reliance on signature-based driver blocklists that do not account for newly surfaced vulnerable drivers; incomplete enforcement of Microsoft's Vulnerable Driver Blocklist, which requires active policy configuration rather than default enablement on many systems; absence of HVCI (Hypervisor-Protected Code Integrity), which prevents unsigned or vulnerable code from executing in kernel memory; and EDR products lacking robust self-protection mechanisms against kernel-level tampering. Organizations running EDR without these complementary kernel integrity controls are exposed regardless of agent version or vendor.

Action Checklist

1. Assess exposure: Inventory all endpoints for EDR agent deployment and confirm kernel-level self-protection features are enabled; verify whether your EDR vendor supports tamper-protection mechanisms that survive kernel-mode attacks.
2. Enforce Microsoft Vulnerable Driver Blocklist: Confirm the blocklist policy is actively enforced via Windows Defender Application Control (WDAC) or equivalent, default system state does not guarantee enforcement. Cross-reference the current blocklist at Microsoft's Vulnerable Driver Blocklist documentation.
3. Enable HVCI where supported: Audit hardware compatibility for Hypervisor-Protected Code Integrity and enable it on all eligible endpoints; HVCI prevents unsigned or vulnerable kernel code from executing and is the most effective technical countermeasure against BYOVD loading.
4. Update threat model: Add BYOVD-based EDR impairment (T1562.001, T1068) as an explicit pre-ransomware technique in your threat register and red team or tabletop against the scenario of an attacker operating blind after EDR termination.
5. Hunt for vulnerable driver presence: Query endpoint telemetry for loaded drivers matching known-vulnerable hashes from the loldrivers.io (community-maintained vulnerable driver database) or your EDR vendor's blocklist feed; flag any drivers loaded from non-standard paths or processes.
6. Brief leadership: Frame the risk as a gap in an assumed-reliable control layer, boards and executives who approved EDR investment need to understand that EDR alone does not close this exposure without kernel integrity enforcement alongside it.
7. Monitor for tooling evolution: Track threat intelligence feeds, CISA advisories, and the Microsoft Security Response Center for new vulnerable driver disclosures and corresponding blocklist updates; treat blocklist maintenance as an ongoing operational task, not a one-time configuration.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and IR leadership if the vulnerable driver hunt (Step 5) identifies any loldrivers.io hash matches on endpoints that run EDR without HVCI enabled, if Sysmon Event ID 6 or Windows Event ID 7045 shows a kernel driver loaded from a non-standard path within the prior 30 days, or if EDR tamper-protection telemetry shows unexpected agent stops not correlated with authorized maintenance — any of these conditions indicates active or recent BYOVD staging rather than theoretical exposure.
Recovery Notes	After deploying WDAC enforcement and HVCI, validate recovery by confirming Windows Event ID 3077 (CodeIntegrity enforce-mode block) is generating events for any subsequent attempts to load blocklisted drivers, and re-run the full vulnerable driver hunt from Step 5 to verify no previously identified hash matches persist as loaded drivers. Monitor Sysmon Event ID 6 and Windows System Event Log for new kernel driver load events for a minimum of 30 days post-deployment, as threat actors who had already staged a vulnerable driver before WDAC enforcement may attempt re-loading via a different signed driver not yet on the blocklist. Verify EDR agent health and tamper-protection status daily for the first two weeks post-recovery, since BYOVD campaigns frequently involve re-entry attempts after defenders respond.
Forensic Artifacts	Sysmon Event ID 6 (Driver Loaded) logs with SHA256 hashes — primary artifact for matching loaded drivers against loldrivers.io database; BYOVD attacks always produce a driver load event immediately before EDR service termination, creating a forensic sequence of Event ID 6 followed within seconds by Event ID 7036 (service stop) for the EDR agent process | Windows System Event Log Event ID 7045 (New Service Installed) with ServiceType=0x1 (kernel driver) — BYOVD tooling creates a new kernel-mode service entry to load the vulnerable driver; this event captures the service name, binary path (often a temp or user-writable directory), and account context, and will appear in the timeline immediately before EDR agent termination | Windows Security Event Log Event ID 4688 (Process Creation) filtered for processes invoking sc.exe, fltMC.exe, or rundll32.exe with driver-loading arguments — EDR-killer tools such as Terminator (Spyboy) use these execution paths to register and load the BYOVD driver; parent-child process chains involving these tools from unexpected parent processes are high-fidelity BYOVD indicators | File system artifacts in user-writable and temp directories — .sys files dropped to C:\Windows\Temp\, C:\Users\[user]\AppData\Local\Temp\, or C:\ProgramData\ with creation timestamps correlated to the attack window; legitimate kernel drivers are not staged in these paths, making any .sys file in these locations a high-confidence BYOVD staging artifact regardless of hash status | Windows CodeIntegrity Event Log (Microsoft-Windows-CodeIntegrity/Operational) Event IDs 3076 and 3077 — if WDAC was in audit mode during the attack, Event ID 3076 entries will record attempted loads of blocklisted or unsigned drivers that would have been blocked in enforce mode, providing a retrospective view of BYOVD driver load attempts that succeeded due to policy not being in enforce mode

Per-Action IR Details

Assess exposure: Inventory all endpoints for EDR agent deployment and confirm kernel-level self-protection features are enabled; verify whether your EDR vendor supports tamper-protection mechanisms that survive kernel-mode attacks.

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Establishing IR capability and validating defensive tool posture before an incident occurs

Controls: NIST IR-4 (Incident Handling) — validate that the incident handling capability itself (EDR) is not susceptible to pre-incident neutralization, NIST SI-4 (System Monitoring) — confirm monitoring tools operate at a protection level that survives kernel-mode tampering, NIST CM-7 (Least Functionality) — identify endpoints where unnecessary kernel-mode drivers may be loaded that expand BYOVD attack surface, CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory) — asset inventory is the prerequisite for knowing which endpoints lack EDR or run EDR without tamper protection enabled, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — the BYOVD exposure gap must be formally entered into the vulnerability management process, not treated as a vendor configuration detail

Compensating: For teams without centralized MDM or EDR console reporting: run 'sc query type= kernel state= all' on each Windows endpoint to enumerate loaded kernel-mode services, then cross-reference against expected baseline. Use osquery with 'SELECT name, path, sha256 FROM drivers WHERE type = "kernel";' to enumerate driver hashes across the fleet. Pipe output to a CSV and compare hashes against the loldrivers.io CSV export (downloadable at loldrivers.io). For EDR tamper-protection status without a console, query the registry key HKLM\SOFTWARE\[VendorName]\[AgentName]\TamperProtection and validate the expected enabled value per vendor documentation.

Evidence: Before assessing exposure, capture a driver baseline snapshot: (1) run 'driverquery /v /fo csv > driver_baseline.csv' on each endpoint to record all currently loaded drivers with paths and status; (2) collect registry hive HKLM\SYSTEM\CurrentControlSet\Services filtered to kernel-mode drivers (Type=1) using 'reg export HKLM\SYSTEM\CurrentControlSet\Services services_snapshot.reg'; (3) record Windows Security Event Log Event ID 7045 (New Service Installed) and Event ID 7036 (Service State Change) for the 30 days prior, as BYOVD attacks load the vulnerable driver as a new service immediately before EDR termination.

Enforce Microsoft Vulnerable Driver Blocklist: Confirm the blocklist policy is actively enforced via Windows Defender Application Control (WDAC) or equivalent — default system state does not guarantee enforcement. Cross-reference the current blocklist at aka.ms/VulnerableDriverBlockList.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: Applying controls that prevent the attacker technique (vulnerable driver loading) from succeeding while systems remain operational

Controls: NIST SI-3 (Malicious Code Protection) — WDAC enforcement of the vulnerable driver blocklist is a kernel-level malicious code prevention control directly blocking BYOVD driver load attempts, NIST CM-7 (Least Functionality) — restricting which kernel drivers may execute is a least-functionality enforcement action specific to the BYOVD technique, NIST SI-7 (Software, Firmware, and Information Integrity) — WDAC policy verification ensures only integrity-verified, non-blocklisted drivers are permitted to load, CIS 2.2 (Ensure Authorized Software is Currently Supported) — blocklisted drivers are by definition unsupported from a security posture standpoint; their presence must be flagged as unauthorized, CIS 4.4 (Implement and Manage a Firewall on Servers) — analogous kernel-enforcement principle: just as host firewall denies unauthorized network paths, WDAC denies unauthorized kernel code paths

Compensating: For teams without Intune or GPO-based WDAC deployment: manually deploy a WDAC base policy using 'New-CIPolicy -FilePath C:\BasePolicy.xml -Level Publisher -Fallback Hash' and merge Microsoft's recommended block rules from the MSRC GitHub repository (github.com/MicrosoftSecurityResponse/IOC — verify this URL before use). Apply via 'ConvertFrom-CIPolicy' and 'CiTool --update-policy'. For immediate enforcement without full WDAC, enable Windows Defender Credential Guard as a partial mitigation and use the free tool DriverView (NirSoft) to manually audit loaded drivers against the blocklist CSV. Set a weekly scheduled task to re-validate the WDAC policy is in enforce mode (not audit mode) by checking 'Get-CimInstance -ClassName Win32_DeviceGuard' and confirming CodeIntegrityPolicyEnforcementStatus = 2.

Evidence: Before enforcing the blocklist, capture: (1) Windows Event Log — CodeIntegrity channel (Microsoft-Windows-CodeIntegrity/Operational) for Event ID 3076 (audit mode block) and 3077 (enforce mode block) — these reveal which drivers on existing endpoints would have been blocked, indicating prior or current BYOVD staging; (2) WDAC policy deployment status via 'CiTool --list-policies' output saved to file; (3) any existing WDAC policy XML files under C:\Windows\System32\CodeIntegrity\CiPolicies\Active\ to establish whether policy is in audit vs. enforce mode; (4) Windows Security Event Log Event ID 4688 (Process Creation) for processes that previously loaded

known-vulnerable drivers such as mhyprot2.sys, gdrv.sys, or dbutil_2_3.sys.

Enable HVCI where supported: Audit hardware compatibility for Hypervisor-Protected Code Integrity and enable it on all eligible endpoints; HVCI prevents unsigned or vulnerable kernel code from executing and is the most effective technical countermeasure against BYOVD loading.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: Implementing the highest-efficacy technical control that structurally prevents the BYOVD kernel driver loading mechanism

Controls: NIST SI-7 (Software, Firmware, and Information Integrity) — HVCI enforces kernel code integrity at the hypervisor layer, making it cryptographically impossible to load unsigned or blocklisted drivers regardless of attacker privileges, NIST SC-39 (Process Isolation) — HVCI uses VBS to isolate kernel code integrity verification in a separate hypervisor-protected context, directly countering kernel-mode privilege abuse, NIST CM-6 (Configuration Settings) — enabling HVCI is a security configuration baseline change that must be standardized across eligible endpoints, CIS 4.6 (Securely Manage Enterprise Assets and Software) — HVCI enablement is a hardening configuration that must be tracked and enforced through configuration management, CIS 7.3 (Perform Automated Operating System Patch Management) — HVCI effectiveness depends on current Windows kernel patches; unpatched kernels may retain exploitable paths even with HVCI enabled

Compensating: For hardware that does not meet HVCI requirements (no VT-x/VT-d, no TPM 2.0, or legacy firmware): use the Microsoft Hardware Readiness Tool ('HardwareReadiness.ps1' from Microsoft) to categorize incompatible endpoints, then prioritize those endpoints for accelerated hardware refresh or network segmentation as a compensating control. On HVCI-incompatible endpoints, deploy Sysmon with a configuration that captures Event ID 6 (Driver Loaded) with hash logging enabled — this provides detection visibility for vulnerable driver loads even without prevention. Use the Sigma rule 'proc_driver_load_susp_paths.yml' (available in the SigmaHQ repository) to alert on drivers loaded from temp directories, user profile paths, or non-standard locations characteristic of BYOVD staging.

Evidence: Before enabling HVCI, capture: (1) current VBS/HVCI status via 'Get-CimInstance -ClassName Win32_DeviceGuard | Select-Object VirtualizationBasedSecurityStatus, CodeIntegrityPolicyEnforcementStatus' — document the pre-change baseline; (2) System Event Log Event ID 12 (Kernel boot) and Event ID 13 (Kernel shutdown) to establish reboot timing for the configuration change; (3) hardware inventory including BIOS/UEFI version, TPM version, and Secure Boot status using 'Confirm-SecureBootUEFI' and 'Get-Tpm' — HVCI activation failures on incompatible hardware generate Event ID 3099 in the Microsoft-Windows-Kernel-Boot/Operational channel; (4) driver compatibility pre-check via 'dxdiag /t dxdiag_output.txt' and Device Manager export to identify drivers that will fail under HVCI (notably older hardware drivers that are not HVCI-compatible).

Update threat model: Add BYOVD-based EDR impairment (T1562.001, T1068) as an explicit pre-ransomware technique in your threat register and red team or tabletop against the scenario of an attacker operating blind after EDR termination.

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Updating the organizational threat model and testing IR capability against realistic adversary scenarios where primary detection tooling has been neutralized

Controls: NIST IR-3 (Incident Response Testing) — tabletop or red team exercise specifically against the BYOVD pre-ransomware kill chain (driver load → EDR termination → ransomware deployment) tests IR capability under realistic degraded-detection conditions, NIST IR-8 (Incident Response Plan) — the IR plan must be updated to include decision trees for scenarios where EDR telemetry is absent or unreliable, which is the direct consequence of a successful BYOVD attack, NIST RA-3 (Risk Assessment) — BYOVD-based EDR impairment must be formally assessed as a risk scenario given its documented use by ransomware groups; threat register update is the formal risk documentation action, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — the threat model update formalizes BYOVD as a vulnerability class requiring ongoing tracking, not a one-time advisory response

Compensating: For a 2-person team without a red team function: run a structured tabletop using the MITRE ATT&CK Navigator (attack.mitre.org/resources/attack-navigator/ — verify before use) to map the BYOVD kill chain: Initial Access → Privilege Escalation (T1068) → Defense Evasion (T1562.001) → Impact. Walk through each phase asking: 'What detects this if EDR is offline?' Document detection gaps in a simple risk register spreadsheet. Use CISA's

Ransomware Vulnerability Warning Pilot (RVWP) notifications as inputs to the threat register update. The tabletop scenario anchor: 'An attacker has loaded gdrv.sys via a signed installer, killed your EDR agent at 2am, and spent 4 hours in your environment — what do you see in non-EDR telemetry?'

Evidence: Before updating the threat model, capture the current state of detection coverage for MITRE T1562.001 and T1068: (1) pull existing SIEM or EDR alert rules mapped to these technique IDs to document current detection posture; (2) review Windows Security Event Log for historical Event ID 7045 (New Service Installed) events where ServiceType = kernel driver, particularly those followed within minutes by EDR service stop events (Event ID 7036 with ServiceName matching your EDR vendor's agent); (3) collect any prior incident records or near-miss events where EDR telemetry showed gaps — these are evidence that BYOVD or similar tampering may have been attempted; (4) document the current threat register entry (or absence of one) for EDR impairment as the pre-change baseline for the threat model update.

Hunt for vulnerable driver presence: Query endpoint telemetry for loaded drivers matching known-vulnerable hashes from the loldrivers.io community database or your EDR vendor's blocklist feed; flag any drivers loaded from non-standard paths or processes.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Proactive hunting for indicators of BYOVD staging before EDR termination occurs, using driver hash telemetry as the primary signal

Controls: NIST SI-4 (System Monitoring) — endpoint telemetry must be configured to capture driver load events with hashes; without this, BYOVD staging is invisible until the EDR agent is already terminated, NIST AU-2 (Event Logging) — driver load events (Sysmon Event ID 6, Windows Event ID 7045) must be explicitly included in the audit event set; they are not logged by default in standard Windows configurations, NIST AU-6 (Audit Record Review, Analysis, and Reporting) — the hunt query against loldrivers.io hashes is an audit record analysis action that must be performed proactively on a defined frequency, CIS 8.2 (Collect Audit Logs) — driver load telemetry is an audit log type that must be explicitly enabled; default Windows logging does not capture driver hashes without Sysmon or equivalent, CIS 7.2 (Establish and Maintain a Remediation Process) — any vulnerable driver hash match must feed directly into a documented remediation workflow with defined SLAs

Compensating: For teams without EDR telemetry: deploy Sysmon with SwiftOnSecurity's config (github.com/SwiftOnSecurity/sysmon-config — verify before use) which enables Event ID 6 (Driver Loaded) with SHA256 hash capture. Export Sysmon Event ID 6 logs via: 'Get-WinEvent -LogName "Microsoft-Windows-Sysmon/Operational" | Where-Object {$_.Id -eq 6} | Select-Object TimeCreated, @{N="Hash";E={$_.Properties[7].Value}}, @{N="Driver";E={$_.Properties[0].Value}} | Export-Csv drivers_loaded.csv'. Download the loldrivers.io CSV (loldrivers.io/drivers.csv — verify before use), then use PowerShell: 'Import-Csv drivers_loaded.csv | Where-Object { (Import-Csv loldrivers.csv).SHA256 -contains $_.Hash }' to identify matches. Flag any driver loaded from %TEMP%, %APPDATA%, or a user-writable path as high-priority regardless of hash match.

Evidence: This step IS the evidence collection action — capture before and during the hunt: (1) Sysmon Event ID 6 (Driver Loaded) logs with ImageLoaded path and SHA256 hash for all endpoints for the prior 30 days; (2) Windows System Event Log Event ID 7045 (New Service Installed) with ServiceType=1 (kernel driver) — BYOVD loads always create a new service entry immediately before exploitation; (3) file system artifacts — check C:\Windows\Temp\, C:\Users\[username]\AppData\Local\Temp\, and C:\ProgramData\ for .sys files dropped within the investigation window using 'Get-ChildItem -Recurse -Filter *.sys -Path C:\' with creation timestamps; (4) Prefetch files (C:\Windows\Prefetch\) for execution evidence of tools known to weaponize BYOVD drivers such as Terminator, GhostDriver, or vendor-specific EDR killers; (5) Windows Security Event Log Event ID 4688 (Process Creation) for processes that invoked 'sc create', 'NtLoadDriver', or 'fltMC' as these are the primary driver-loading mechanisms.

Brief leadership: Frame the risk as a gap in an assumed-reliable control layer — boards and executives who approved EDR investment need to understand that EDR alone does not close this exposure without kernel integrity enforcement alongside it.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Communicating control layer gaps to leadership to drive resource allocation for HVCI and WDAC enforcement as structural remediation, not optional hardening

Controls: NIST IR-6 (Incident Reporting) — leadership briefing on a systematic control gap (EDR killability via BYOVD) meets the threshold for organizational-level incident reporting even in the absence of a confirmed breach, NIST IR-8 (Incident Response Plan) — the briefing should result in a documented decision: either fund HVCI/WDAC enforcement or formally accept the residual risk of EDR impairment in writing, NIST RA-3 (Risk Assessment) — the risk framing must quantify blast radius: if BYOVD neutralizes EDR across all endpoints, what is the organization's detection and response capability during a subsequent ransomware deployment?, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — the leadership briefing is the escalation step that moves BYOVD from a technical finding to a formally tracked organizational vulnerability with ownership and remediation funding

Compensating: For teams without a formal risk committee or GRC platform: prepare a one-page executive brief using this structure — (1) What the attacker does: loads a legitimately signed Windows driver (example: mhyprot2.sys from a video game anti-cheat, or dbutil_2_3.sys from a Dell update tool) to terminate the EDR agent before deploying ransomware; (2) What this means for our investment: the EDR product we purchased cannot protect itself against this technique without additional kernel-level configuration (HVCI + WDAC); (3) What remediation costs: HVCI is free on Windows 10/11 on compatible hardware — the cost is IT time for deployment and a hardware refresh plan for incompatible endpoints; (4) What accepting the risk means: document in a simple risk acceptance form signed by the CISO or equivalent, with a review date no more than 90 days out.

Evidence: For the leadership briefing, compile the following as supporting evidence: (1) output from the vulnerable driver hunt (Step 5) — if any loldrivers.io hash matches were found, lead with those as proof of existing exposure; (2) EDR coverage gap report showing percentage of endpoints without tamper protection enabled or without HVCI; (3) threat intelligence citations — reference specific ransomware groups documented as using BYOVD (e.g., groups using Terminator/Spyboy tool, BlackCat/ALPHV affiliates, Scattered Spider) to establish that this is active adversary tradecraft, not theoretical; (4) CISA advisories or MSRC bulletins on new vulnerable driver disclosures from the prior 90 days as evidence of the ongoing and escalating nature of the threat.

Monitor for tooling evolution: Track threat intelligence feeds, CISA advisories, and the Microsoft Security Response Center for new vulnerable driver disclosures and corresponding blocklist updates; treat blocklist maintenance as an ongoing operational task, not a one-time configuration.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Establishing continuous intelligence-driven processes to maintain defensive posture as the BYOVD tooling ecosystem evolves

Controls: NIST SI-5 (Security Alerts, Advisories, and Directives) — CISA advisories and MSRC driver disclosures are the primary authoritative alert feeds for new BYOVD-capable drivers and must be formally integrated into the operational workflow, NIST IR-5 (Incident Monitoring) — the loldrivers.io database and Microsoft blocklist are ongoing monitoring inputs for the driver-load threat category; changes to these feeds must trigger re-evaluation of endpoint posture, NIST AU-6 (Audit Record Review, Analysis, and Reporting) — the vulnerable driver hunt (Step 5) must be operationalized as a recurring scheduled hunt, not a one-time query, with frequency tied to blocklist update cadence, CIS 7.2 (Establish and Maintain a Remediation Process) — each new vulnerable driver added to the Microsoft blocklist or loldrivers.io represents a new remediation item that must enter the documented remediation process with a defined SLA, CIS 7.4 (Perform Automated Application Patch Management) — WDAC policy updates incorporating new blocklist entries must be treated as a patch management action with the same cadence discipline as OS and application patching

Compensating: For teams without a threat intelligence platform: set up free RSS or email monitoring for (1) CISA KEV catalog updates (cisa.gov/known-exploited-vulnerabilities-catalog — verify before use) filtering on driver-related entries; (2) Microsoft Security Update Guide filtered on 'driver' and 'Windows kernel'; (3) loldrivers.io GitHub repository watch for new driver additions (github.com/magicsword-io/LOLDrivers — verify before use). Operationalize the hunt from Step 5 as a monthly scheduled PowerShell task that pulls the latest loldrivers.io CSV, compares it against the local Sysmon Event ID 6 log export, and emails results to the security team. Set a calendar reminder on Patch Tuesday +2 days to download the updated Microsoft vulnerable driver blocklist and redeploy the WDAC policy if the blocklist has changed.

Evidence: Before establishing the monitoring workflow, capture baseline intelligence state: (1) current version/date of the Microsoft Vulnerable Driver Blocklist in use (check the WDAC policy XML for embedded blocklist version or the DriverSiPolicy.p7b file timestamp at C:\Windows\System32\CodeIntegrity\); (2) current loldrivers.io database snapshot

(download the full CSV and hash it with 'Get-FileHash loldrivers.csv -Algorithm SHA256' to enable future change detection); (3) historical CISA advisories referencing BYOVD or EDR-killing tooling from the prior 6 months as a baseline for tracking new advisory frequency; (4) a record of which MITRE ATT&CK techniques (T1562.001, T1068) currently have detection rules in your SIEM or Sysmon config so that new sub-techniques or technique updates can be matched against existing coverage gaps.

Detection Guidance

Detection for BYOVD activity must occur at multiple layers because by the time an EDR agent is terminated, the agent itself can no longer report what happened.

Kernel driver load events: Enable and monitor Windows Event ID 6 (kernel driver loaded) via Sysmon or equivalent. Alert on drivers loaded from non-standard paths (user profile directories, temp folders, or paths outside System32/Drivers), and cross-reference loaded driver hashes against the Microsoft Vulnerable Driver Blocklist and loldrivers.io (community-maintained vulnerable driver database).

Service creation anomalies: T1543.003 activity surfaces in Windows Event ID 7045 (new service installed). Flag services created by non-administrative users or from unusual parent processes, particularly those creating kernel-mode services outside of expected software installation windows.

EDR process termination: If your EDR platform supports it, alert on unexpected agent process termination or loss of telemetry from previously reporting endpoints. Gaps in endpoint telemetry, especially endpoints that stop reporting during a suspicious activity window, are a high-fidelity signal of impairment attempts.

Privilege escalation chains: Hunt for T1068 patterns, processes escalating from user-mode to kernel-mode privileges via driver interaction, particularly where the escalating process is not a recognized system binary or installer.

Direct kernel object manipulation (DKOM): Some EDR and XDR platforms expose kernel object access telemetry. Hunt for process handle operations against EPROCESS structures or attempts to modify kernel callback lists from unexpected processes.

Log sources to prioritize: Sysmon (Event IDs 6, 19), Windows Security Event Log (4697 service installation, 7045), kernel audit logs if HVCI or Credential Guard telemetry is available, and EDR self-health or heartbeat logs.

Indicators of Compromise

Type	Value	Context	Confidence
TOOL	`Pending — refer to Dark Reading source article and loldrivers.io for published vulnerable driver hashes`	Vulnerable signed Windows kernel drivers used to terminate or blind EDR agents via direct kernel object manipulation; specific driver hashes and names referenced in source reporting but not available in the provided excerpt	LOW

Framework Mappings

MITRE-ATTACK

- **T1562.001** — Disable or Modify Tools
- **T1211** — Exploitation for Defense Evasion
- **T1014** — Rootkit
- **T1543.003** — Windows Service
- **T1195** — Supply Chain Compromise
- **T1068** — Exploitation for Privilege Escalation

NIST-800-53R5

- **SA-9** — External System Services
- **SR-2** — Supply Chain Risk Management Plan
- **SR-3** — Supply Chain Controls and Processes
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-6** — Least Privilege
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **AC-3** — Access Enforcement
- **CP-9** — System Backup
- **IR-4** — Incident Handling
- **AT-2** — Literacy Training and Awareness
- **SI-4** — System Monitoring

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control

CIS-V8

- **5.4** — Restrict Administrator Privileges to Dedicated Administrator Accounts
- **6.8** — Define and Maintain Role-Based Access Control
- **3.3** — Configure Data Access Control Lists
- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process
- **14.2** — Train Workforce Members to Recognize Social Engineering Attacks
- **8.2** — Collect Audit Logs

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control
- **164.308(a)(7)(ii)(A)** — Data Backup Plan

NIST-CSF-2

- **RS.MI-01** — Incidents are contained
- **DE.CM-01** — Networks and network services are monitored

ISO-27001-2022

- **A.5.29** — Information security during disruption
- **A.5.34** — Privacy and protection of personal information

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1562.001	Disable or Modify Tools	Defense-Evasion
T1211	Exploitation for Defense Evasion	Defense-Evasion
T1014	Rootkit	Defense-Evasion
T1543.003	Windows Service	Persistence
T1195	Supply Chain Compromise	Initial-Access
T1068	Exploitation for Privilege Escalation	Privilege-Escalation

Sources

Source	URL	Tier
Security News	https://www.darkreading.com/vulnerabilities-threats/edr-killer-ecos...	T3
6 EDR Vendors For 2026	https://www.sentinelone.com/cybersecurity-101/endpoint-security/edr...	T3
Top 6 EDR Tools Compared [2025 Update]	https://www.cynet.com/security-foundations/endpoint-security/top-6-...	T3
What Is EDR? Endpoint Detection and Response	https://www.microsoft.com/en-us/security/business/security-101/what...	T1
15 Top Endpoint Detection and Response (EDR) Solutions	https://www.sangfor.com/blog/cybersecurity/15-top-endpoint-detectio...	T3

Generated 2026-04-14 18:30 UTC by TJS Security Command Center