ArmorxAI · RansomArmor

IOC vs. IOA

What EDR Misses. What RansomArmor Catches First.

Detection Is Not Prevention.

THE FUNDAMENTAL DIFFERENCE

EDR operates on Indicators of Compromise (IOCs)

IOCs are forensic artifacts — known file hashes, malicious IP addresses, C2 domain signatures, registry changes. By definition, an IOC only exists after something has already happened. The malware has executed. The file has been written. The process has already run.

EDR is evidence collection on an active crime scene.

RansomArmor operates on Indicators of Attack (IOAs)

IOAs are behavioral signals — the sequence of actions an attacker must take before a payload detonates. No hash required. No signature required. No cloud lookup required. The attack intent is visible in behavior long before any IOC forms.

RansomArmor intercepts at the kernel level in milliseconds — at the stage where EDR cannot yet see anything to flag.

"We don't wait for confirmed IOCs. We stop Indicators of Attack before they become IOCs — in milliseconds, at the kernel level."

01 · EXECUTION & INJECTION IOAS

1

Process Hollowing

A legitimate process (e.g., svchost.exe) is spawned in a suspended state, its memory unmapped and replaced with malicious code, then resumed. EDR sees a trusted process name and nothing more. RansomArmor detects the memory unmapping and rewrite sequence at the kernel I/O layer before the process ever resumes execution.

2

Thread Execution Hijacking

An attacker acquires a handle to a live trusted process, suspends an existing thread, injects shellcode into its memory space, and redirects execution. No new process is created — EDR has nothing to flag. RansomArmor detects the SuspendThread / WriteProcessMemory / SetThreadContext / ResumeThread API call chain as an IOA before the thread resumes.

3

Remote Shell Injection

Malicious code injected into a remote process to open a shell or establish command-and-control persistence. Operates entirely inside the memory space of a trusted process, invisible to application-layer tools. RansomArmor catches the unauthorized remote memory write and execution handoff at the kernel level before the shell activates.

4

DLL Sideloading

A legitimate signed application is tricked into loading a malicious DLL placed alongside it. The trusted application — not the attacker — loads the payload, bypassing application control and EDR allow-lists entirely. RansomArmor flags the anomalous DLL load path against the behavioral baseline before the DLL executes.

5

DLL Injection — Classic & Module Stomping

Malicious DLL path written into the virtual address space of a target process via VirtualAllocEx / WriteProcessMemory / CreateRemoteThread. The Module Stomping variant overwrites an already-loaded legitimate DLL's entry point to hide execution inside a trusted module. RansomArmor detects unauthorized cross-process memory writes and entry point manipulation before execution.

6

Portable Executable (PE) Injection

Malicious shellcode copied directly into the memory of a target process and triggered via CreateRemoteThread — no DLL file, no disk artifact, no file hash to scan. Completely invisible to signature-based and file-hash detection. RansomArmor catches the memory allocation pattern and execution handoff at the kernel layer.

7

BYOVD — Bring Your Own Vulnerable Driver

Attackers load a legitimate, signed but vulnerable driver to operate at the kernel level, bypassing EDR kernel protections entirely. One of the fastest-growing EDR evasion techniques in active ransomware campaigns. RansomArmor's Windows Kernel Driver access detects the anomalous driver load and weaponization sequence before the vulnerable driver can be exploited.

02 · PERSISTENCE & PRIVILEGE ESCALATION IOAS

8

Credential Harvesting Attempts

Unauthorized access to LSASS memory, SAM database reads, or credential store enumeration — the attacker acquiring the access they need before any payload deploys. EDR may catch known tooling like Mimikatz via hash. RansomArmor detects the access behavior pattern as an IOA regardless of what tool is used to execute it.

9

Privilege Escalation Sequences

Attempts to elevate from user-level to SYSTEM or admin via token impersonation, UAC bypass, or service exploitation — a required step before ransomware can encrypt system-protected files and critical directories. RansomArmor flags the escalation attempt sequence as an IOA before elevated access is granted, shutting off the attack path.

10

Registry Manipulation for Persistence

Attacker writes to run keys, AppInit DLLs, Image File Execution Options, or service entries to ensure the payload survives reboot or re-executes on schedule. RansomArmor monitors registry I/O operations at the kernel level for unauthorized write patterns associated with persistence staging — before the foothold is established.

11

Scheduled Task & Service Creation Abuse

Malicious actors create or hijack scheduled tasks or Windows services to maintain foothold or trigger payload execution at a chosen time — including after reboots. RansomArmor detects anomalous task and service creation behavior as an IOA before the scheduled execution fires, denying the persistence mechanism entirely.

03 · BEACON DOWNLOAD & C2 STAGING IOAS

12

Beacon Download — Disk-Based

Payload delivery via disk-based beacon — writing a secondary stage executable to disk before execution. RansomArmor detects the file write and execution initiation sequence before the beacon runs, regardless of whether the file matches any known hash or appears in any threat intelligence feed.

13

Beacon Download — Memory-Based (Fileless)

Fileless beacon delivery — payload injected and staged entirely in memory, never written to disk. Completely invisible to file-scanning, hash matching, and all signature-based detection. RansomArmor operates at the memory I/O layer and catches the injection and staging behavior before the beacon activates.

14

Unauthorized Outbound Connection Staging

Pre-exfiltration network behavior — attempts to establish connections to external IPs or domains for C2 check-in or data staging, before encryption or exfiltration begins. A critical IOA signaling the attacker is establishing command and control before deploying ransomware across the environment.

04 · LATERAL MOVEMENT & PROPAGATION IOAS

15

Lateral Movement Process Spawning

Anomalous process spawning across the environment — a compromised process creating children on remote hosts, deploying the ransomware payload across the network in preparation for mass encryption. RansomArmor detects the process chain behavior and propagation pattern before encryption begins on secondary hosts.

16

Bulk File Access Prior to Encryption

Rapid, anomalous I/O reads across file directories — the reconnaissance pass an attacker makes to identify, enumerate, and stage target files before the encryption payload fires. A detectable IOA in the file I/O behavior pattern that precedes the encryption event EDR would catch only after files are already locked.

17

Shadow Copy & Backup Deletion Attempts

Execution of vssadmin delete shadows, wbadmin delete backup, or equivalent commands — the attacker destroying recovery infrastructure before deploying the encryption payload to ensure maximum leverage. EDR often catches this only after deletion completes. RansomArmor detects the process call pattern as an IOA and terminates it before the deletion executes.

05 · POST-EXPLOITATION WORKFLOW IOAS

18

Post-Exploitation Framework Staging

Loading of post-exploitation tooling — Cobalt Strike, Sliver, Brute Ratel — into memory via reflective DLL injection or beacon staging. These frameworks are the infrastructure attackers build and stabilize before deploying ransomware. RansomArmor detects the behavioral fingerprint of post-exploitation framework deployment at the kernel I/O layer before the framework becomes operational.

19

Defense Disabling & Anti-Analysis Behavior

Attempts to disable security tools, stop security services, clear event logs, or tamper with logging infrastructure before payload deployment — attackers clearing the field before they act. A strong pre-execution IOA. RansomArmor catches process calls targeting security services as an IOA regardless of whether the specific tool or technique has a known signature.

20

Living-Off-The-Land Script Abuse — PowerShell / WMI / CMD

Using trusted Windows scripting tools to run malicious commands, download payloads, or stage attacks — no malicious binary required, no file hash to scan. RansomArmor detects the process ancestry and execution chain anomaly at the kernel level, where the script interpreter's child processes and memory activity reveal attack intent before any payload fires.

THE CONTRAST — SIDE BY SIDE

	EDR *IOC-Based · Reactive*	**RansomArmor** *IOA-Based · Preemptive*
When it sees the threat	After execution starts	Before execution — milliseconds
What it requires	Known hash, signature, C2 domain	Behavioral pattern — no signature needed
Zero-day coverage	Blind — no hash to match	Full — behavior doesn't change
Fileless attack coverage	Minimal to none	Full — kernel-level memory I/O
BYOVD coverage	Bypassed by design	Blocked at kernel driver layer
Time to act	4–60 min (industry avg)	Milliseconds
Cloud / internet required	Yes — for most lookups	None — fully offline capable
Files encrypted before response	Yes — gap is unavoidable	Zero — stopped before encryption begins
Downstream IR invoked	Yes — IR, backup, ransom, insurance	Never — execution denied

EDR waits for a crime scene. RansomArmor stops the crime from happening.

Why This Matters in a Sales Conversation

The confusion between detection and prevention is the single most exploited gap in enterprise security buying decisions. EDR vendors benefit from buyers assuming 'we have EDR, we're covered.' The question that exposes this assumption in any discovery call:

"What is your EDR's average time from ransomware execution to containment?"

EDR industry average: 4–60 minutes. RansomArmor: milliseconds.

In that 4–60 minute window, files are being encrypted, backups are being deleted, and IR is being invoked. When execution is denied entirely — IR, backup recovery, ransom negotiation, and insurance claims are never invoked. That is the shift from resilience to avoidance.

NSA CRADA Research Partner · NSF SBIR Phase II Winner · armorx.ai