ENDOR LABS

MALWARE DEFENSE

A Multi-Agent detection engine and package firewall

Robert Haynes
Principal Technical Marketing Engineer

Table of Contents

Introduction: A New Category of Threat

Software composition analysis (SCA) has long focused on one type of risk: known vulnerabilities in third-party dependencies. These are unintentional weaknesses—bugs introduced by maintainers without malicious intent, cataloged in databases like the National Vulnerability Database (NVD) and tracked via CVEs. For years, this has been the dominant concern in open source security.

But a different threat has grown substantially more serious: open source malware. Unlike vulnerabilities, malware is intentional. An attacker deliberately crafts a package (or compromises an existing one) to execute harmful code when developers install, import, or run it. The goal is typically to steal credentials, exfiltrate data, establish persistence, or create a backdoor into the developer's environment or downstream systems.

This is not a hypothetical concern. High-profile incidents like the XZ Utils backdoor, the ongoing wave of malicious packages targeting npm and PyPI, and campaigns impersonating popular packages have demonstrated that supply chain attacks through open source are a proven and scalable attack vector. Attackers have realized that a single malicious package, if installed broadly, can compromise thousands of development environments and production systems simultaneously. Traditional SCA tools are poorly equipped to detect malware. They match package identifiers against CVE databases, but a newly published malicious package has no CVE assigned to it. It has no NVD entry. Its version number looks valid, its name may closely resemble a legitimate package, and its `README` may appear entirely professional. Signature-based detection alone will miss it.

Endor Labs takes a fundamentally different approach. While we obviously subscribe to multiple vulnerability feeds (a mix of commercial and open source), we also take a parallel proactive approach. Rather than waiting for a package to appear in a vulnerability database, we analyze packages directly: examining their code, metadata, publisher behavior, network activity, and more to identify suspicious patterns before any CVE is ever filed (if ever).

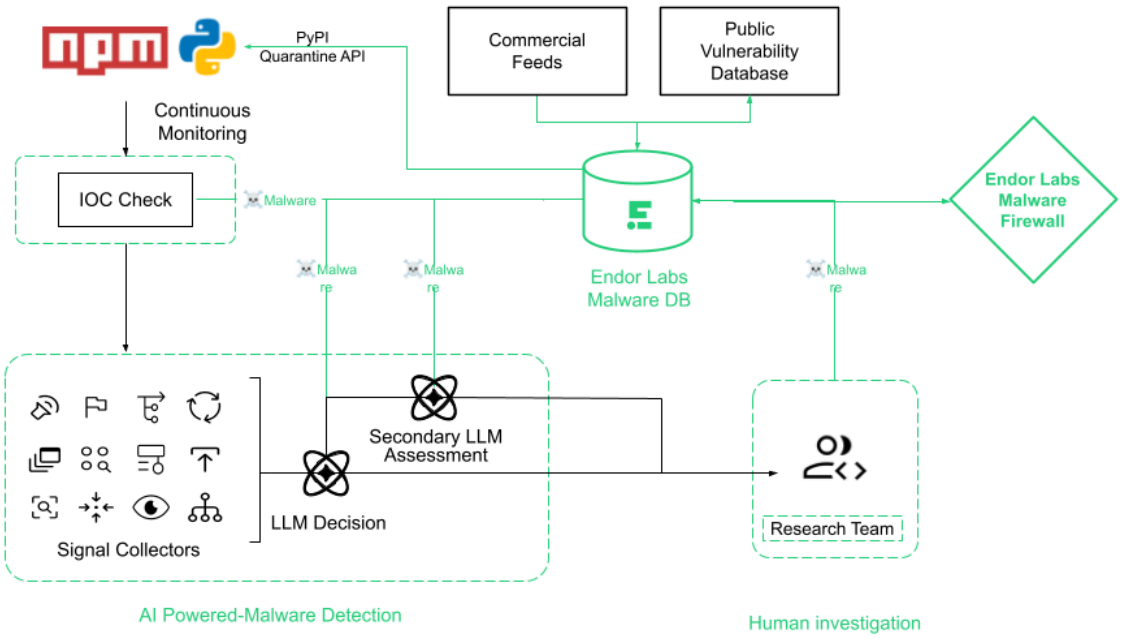
How we identify malware: a multi-signal approach

Effective malware detection at scale requires more than a single heuristic. Sophisticated attackers deliberately craft packages to evade simple checks: using code obfuscation to hide malicious logic, legitimate-looking metadata (such as inflated download numbers) to avoid name-based filters, delayed execution to bypass sandbox analysis, and many more... To stay ahead of these techniques, Endor Labs employs a multi-signal detection architecture that combines over a dozen independent analysis modules.

Each module — called a signal producer — examines a different aspect of a package and provides an independent assessment. Those assessments are then aggregated and passed to a large language model (LLM) for final scoring. This layered approach means no single evasion technique can defeat the system, and that novel attack patterns are more likely to be caught by at least one signal even before dedicated rules are written for them.

Signal producers

Signal producers look at the contents of the package (including pre- and post-install hooks), the package metadata, package maintainer characteristics, and package behavior on install to build a comprehensive suite of signals for LLM-assessment. The signals are regularly reviewed and enhanced, and the architecture is designed for easy inclusion of additional collectors as we develop them.



Endor Labs Malware Detection Architecture

LLM-Based assessment: from signals to verdicts

Raw signals from individual producers provide evidence, but converting that evidence into a reliable malicious/benign verdict requires reasoning across multiple, potentially conflicting signals simultaneously. This is where Endor Labs' LLM-based assessment layer comes in.

The system uses a three-phase approach that balances speed with thoroughness. In the first phase, fast indicator-of-compromise (IoC) checks run against all packages. Packages that return a positive result are immediately flagged as malicious, and further processing is stopped, apart from matching their attributes with known campaigns for tracking purposes. Packages that pass with no flags proceed to the second phase: a full LLM assessment, in which all collected signals, code samples, and metadata are analyzed holistically. Under some circumstances (e.g., where the package has a high number of downloads, resulting in a larger impact in the event of an erroneous result), results are processed by a second LLM stage for deeper analysis. This staged approach delivers results quickly and requires the minimum computational resources to reach a decision.

The LLM produces a confidence score indicating its assessment of the package's maliciousness. This scoring approach means that Endor can rapidly block clearly malicious packages and surface lower confidence findings for review by Endor security researchers rather than silently passed or incorrectly blocked.

The detection pipeline is designed to be declarative and extensible. New signal producers can be added and configured without requiring changes to the core detection engine. As the threat landscape evolves with new attack patterns, package ecosystems, and delivery mechanisms, detection coverage can be updated rapidly.

High confidence findings from the LLM stages are also fed back into the initial rule-based IoC stage to ensure faster detection of malicious packages in the future.

As part of our responsible disclosure principles, malicious packages are reported to package registries as quickly as possible, including using tools like the PyPI API, to ensure that malicious packages are blocked or quarantined as soon as possible.

From identification to protection: the package firewall

Detection without enforcement is incomplete security. Knowing that a package is malicious is only valuable if you can stop it from being installed. Endor Labs' Package Firewall closes that gap by acting as a policy-enforcing proxy between your development environment and public registries like npm and PyPI.

When a developer runs `npm install` or `pip install`, the request is routed through the Package Firewall. The Firewall evaluates each requested package against a configurable policy

and either allows the download to proceed, returns a warning, or blocks the request outright. This happens transparently—developers interact with their existing tooling as normal, with protection operating in the background.

> ***The Package Firewall evaluates packages before they're installed — not after the damage is done. There's no cleanup step, no incident response: malicious packages simply never reach the developer's machine.***

Policy evaluation order

Packages are evaluated against four policy layers in a defined sequence. The ordering is intentional: exceptions are checked first so that explicitly approved packages are never blocked, while the strictest safety check (malware) comes before license and age restrictions. This prevents a scenario where a malicious package would be allowed through on a technicality.

Step	Policy Check	Outcome if Matched
1	Exception list	Always allow: bypass all other checks
2	Malware detection	Block: package identified as malicious
3	License restriction	Block or warn:non-compliant license
4	Minimum age	Block or warn: package too new to be trusted

Configurable policy via declarative YAML

Security teams define their Package Firewall policy using a declarative YAML configuration that is version-controlled and applied consistently across all environments. This gives organizations granular control without requiring code changes or custom integrations.

The configuration supports flexible version range matching, so exceptions can be scoped to specific versions rather than blanket-allowing an entire package. License policies can be defined as allowlists or blocklists. Minimum age thresholds can be tuned — a common best practice is to require that packages be at least 21 days old before they are permitted, since a disproportionate share of malicious packages are published and then quickly used in attacks within the first few days of appearing in a registry.

Key capabilities of the policy engine include:

- **Version range matching**: Allow or block specific version ranges (e.g., allow `>=1.2.0, <2.0.0`) without affecting other versions
- **Exception lists**: Pre-approve specific packages and versions that are known good, ensuring they are never incorrectly blocked by other policy layers
- **License policy**: Enforce license allowlists or blocklists to ensure compliance with your organization's IP policies
- **Minimum package age**: Require packages to have existed in the registry for a minimum period before they can be installed
- **Logging**: Every Firewall decision is logged with full context: the policy rule that triggered, the action taken, and the package details, creating a complete audit trail

Firewall threat defense

The Package Firewall's effectiveness stems from the detection layer feeding into it in real time. As Endor Labs' detection pipeline identifies new malicious packages, the Firewall can immediately begin blocking them for all customers. Protection is not limited to previously known malware families — any package that exceeds the confidence threshold is blocked, including newly discovered attacks.

The Firewall is particularly effective against the most common malware delivery vectors:

- **Install-time attacks** — Packages that execute malicious code via lifecycle hooks (postinstall, preinstall) are blocked before installation begins
- **Typosquatting** — Packages with names designed to be confused with legitimate dependencies are identified by the typosquatting signal producer and blocked
- **Dependency confusion** — Attacks that exploit how package managers resolve internal vs. public package names are caught by metadata and publisher account analysis
- **Compromised legitimate packages** — The code diff signal producer detects when malicious code is injected into a new version of an existing, trusted package.

A complete picture of open source risk

Vulnerability management remains important — but it represents only part of the open source risk landscape. As attackers have recognized that the software supply chain is a high-leverage target, intentional malware in open source packages has become a threat that security teams can no longer afford to treat as an edge case.

Endor Labs provides the detection depth and enforcement capability needed to address this threat at scale. The multi-signal analysis pipeline catches malicious packages across a broad range of techniques and attack vectors. The LLM-based assessment layer converts raw signals into actionable verdicts with calibrated confidence. And the Package Firewall translates those verdicts into real-time protection that stops malicious packages before they ever touch a developer machine or production build.

Together, these capabilities give security teams a complete picture of open-source risk — one that includes not just what is broken, but also what is malicious.

> ***Endor Labs monitors public registries continuously. When a new malicious package is detected, all customers with the Package Firewall enabled are protected immediately — without any manual action required.***

Next Steps

If you're ready to see Endor Labs' malware detection and Package Firewall in action, we offer a guided evaluation tailored to your environment and package ecosystem.

- Visit **endorlabs.com** to learn more and request a demo
- Talk to our team about integrating the Package Firewall into your existing CI/CD pipeline
- Explore how Endor Labs can consolidate your SCA, malware detection, and supply chain security into a single platform