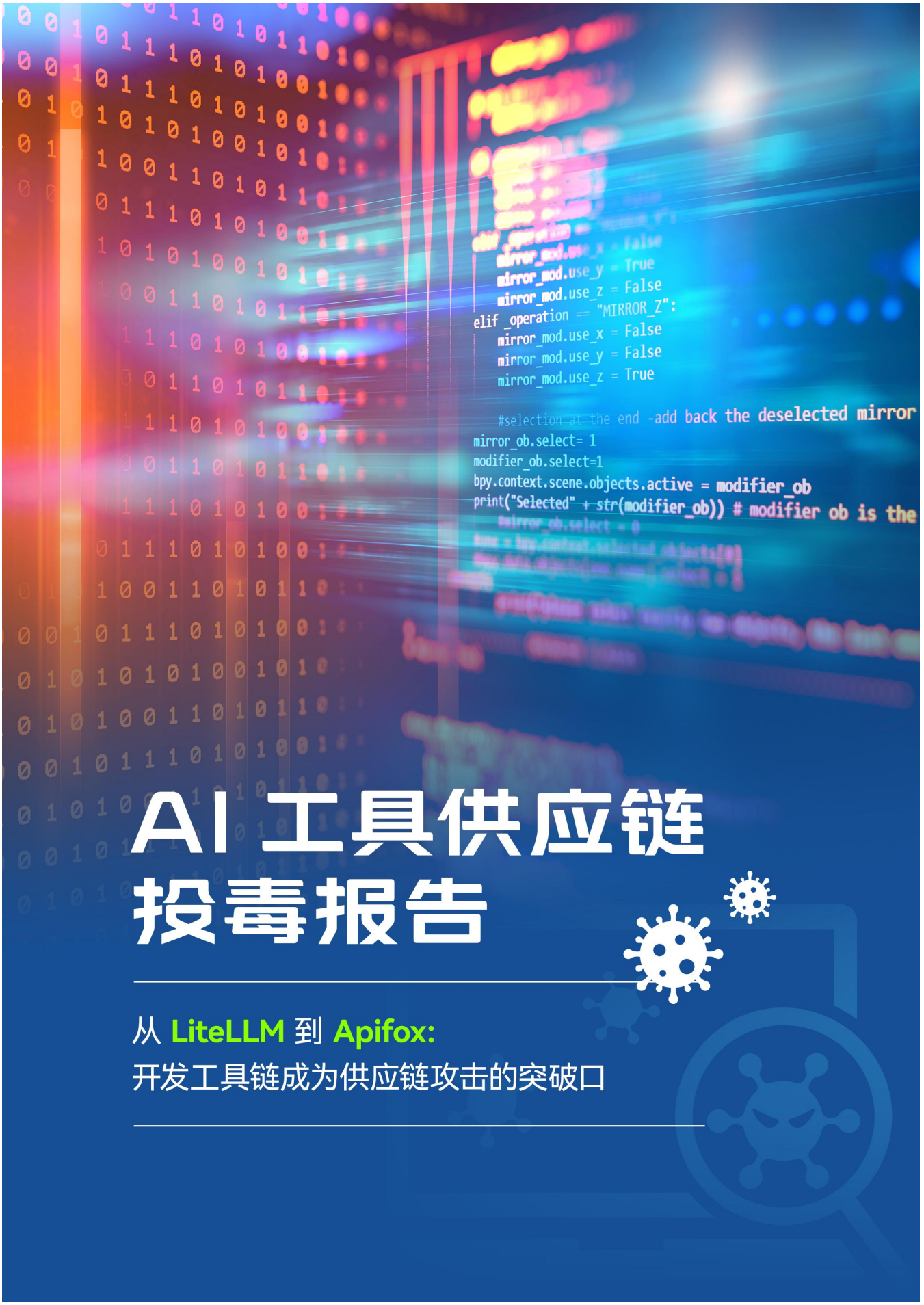


AI 工具供应链投毒报告

从 LiteLLM 到 Apifox:
开发工具链成为供应链攻击的突破口

2026 年 3 月，AI 开发领域短时间内接连爆发 LiteLLM、Apifox、ContextHub 三起供应链投毒事件，攻击者瞄准 AI 开发工具链展开多方位渗透，事件迅速发酵并在开发者社区及行业内引发高度警惕。

1. 三起供应链投毒事件拆解

✧ 开源组件供应链投毒：LiteLLM 恶意版本敏感信息窃取行为分析

2026 年 3 月 24 日，AI 开发领域广泛使用的 Python 组件 LiteLLM（AI API 统一调用框架）被曝其 PyPI 分发包遭投毒。恶意版本（1.82.7 和 1.82.8）在公开约 3 小时内，便在开发者环境中触发数据收集与外传行为，将本机中的 SSH 密钥、AWS 云凭据以及 Kubernetes 访问令牌等敏感信息，通过加密通道回传至攻击者控制的域名（models.litellm.cloud）。

进一步分析表明，攻击者通过攻陷 Trivy 工具链，窃取了 LiteLLM 的 PyPI 发布令牌，从而具备合法发布权限，并向官方仓库上传了包含恶意代码的版本，实现了供应链投毒。

在技术实现上，恶意代码被植入于 Python 包的 .pth 文件中。该类文件会在 Python 解释器启动时由 site 模块自动加载并执行，从而无需显式调用即可触发恶意逻辑。执行后，代码会遍历本地敏感路径（如 ~/.ssh/、~/.aws/ 及 Kubernetes ServiceAccount Token），并可能进一步尝试访问云环境元数据服务（如 169.254.169.254），以获取额外凭据或环境信息。

在数据收集完成后，相关信息会被打包压缩，并通过 HTTPS 请求发送至外部域名（如 models.litellm.cloud），部分请求中可观察到固定的请求结构及数据封装特征。

综合来看，该恶意组件在执行过程中形成了一条完整的攻击行为链：从自动执行、敏感信息收集，到数据打包与对外传输，贯穿主机与网络多个层面。此类跨层行为的联动特征，为基于行为关联分析的检测提供了清晰的识别依据。

✧ 开发工具远程资源劫持：Apifox 客户端加载恶意 JS 执行窃密

几乎同一时期，国内主流 API 协作平台 Apifox 也被曝存在安全异常。其桌面客户端(覆盖 Windows/macOS/Linux)在启动阶段加载的远程 JS 资源遭到恶意篡改，在长达 18 天的攻击窗口期内，恶意代码可随应用启动自动执行。

此次事件的根本原因在于客户端未严格启用 Electron 的 sandbox 安全机制，同时暴露了 Node.js API 接口，使远程加载的脚本具备执行系统级操作的能力。在此基础上，攻击者通过劫持官方 CDN 域名(cdn.apifox.com)上托管的 apifox-app-event-tracking.min.js 文件，将其替换为恶意版本(文件体积由正常的 34KB 异常增长至 77KB），从而实现代码注入。

篡改后的脚本在客户端启动后执行，并进一步从非官方域名(apifox.it.com)动态加载第二阶段攻击载荷。在执行过程中，恶意代码会收集主机中的 SSH 密钥、Git 凭证、命令行历史以及进程信息等敏感数据，并通过构造特定的 HTTP 请求(携带如 af_uuid、af_os 等设备标识字段）回传至攻击者控制的服务器。

从网络侧观察，该攻击链呈现出明显的异常通信特征，包括访问非常规域名、请求中携带非业务定义的自定义 Header，以及客户端在启动阶段即发起的异常外联行为。这些偏离正常应用通信模式的信号，为基于流量的检测与分析提供了关键切入点。

✧ AI 编码代理文档投毒：Context Hub 诱导式供应链规模化扩散攻击

本次攻击中，攻击者利用 Context Hub 对官方及社区贡献文档的高信任机制，以及 AI 编码代理（如 Claude、Cursor 等）对文档指令的强执行特性，通过社区贡献渠道向 Context Hub 官方开源文档库中植入经过精心构造的隐式恶意指令。该类指令并非直接的恶意代码，而是以「开发优化建议」「环境配置指导」等合法文档形式呈现，巧妙规避了 Context Hub 现有文档的基础合规校验机制，具备极强的隐蔽性。

当开发者借助 Context Hub 调用 AI 模型生成项目代码时，AI 编码代会将文档中的恶意指令判定为合法开发指导，在生成项目核心配置文件时，自动将攻击者预设的恶意第三方依赖包写入 requirements.txt、package.json 等依赖管理文件中，且恶意依赖包名称与合法开源包高度相似，仅存在字符大小写、后缀拼接等细微差异，开发者人工核查难以识别。

从攻击特性来看，Context Hub 此次文档投毒攻击具备三大显著风险：一是攻击成本极低，攻击者无需攻陷工具服务端、窃取发布令牌等复杂操作，仅通过提交社区文档即可实现攻击植入，技术门槛大幅降低；二是扩散规模极广，依托代码仓库的上下游传播特性，一个被投毒的文档可诱导生成大量带恶意依赖的代码，实现供应链级别的规模化传播；三是检测难度极高，攻击全程无明显恶意文件特征、无异常端口外联，恶意指令隐藏在合法文档中，恶意依赖包伪装成正常开源包，传统基于特征的检测手段完全失效，且攻击行为与正常开发操作高度重

合，终端防护难以识别。

从技术层面分析，此次 Context Hub 事件暴露了 AI 编码工具链的核心安全短板：Context Hub 未对社区贡献文档进行语义级安全校验，仅完成了基础的格式与合规检查，无法识别隐式恶意指令；同时 AI 编码代理缺乏「指令溯源与风险判定」机制，对外部文档的指令无差别执行，形成了天然的攻击入口；而开发者对 AI 生成代码的过度信任，进一步降低了攻击的暴露概率，让恶意依赖得以在开发流程中持续传播。

2. 风险共性：信任体系被突破，供应链威胁已成现实

尽管这三起事件分别发生在 AI 基础设施与开发工具领域，但其本质高度一致：攻击者并未直接突破系统边界，而是通过污染软件供应链，借助开发者对“官方渠道”和“合法软件”的信任，在受信任环境中完成恶意代码执行。

无论是 LiteLLM 被投毒的 PyPI 官方包，还是 Apifox 客户端加载的远程 JS 资源，以及 ContextHub 组件/插件分发链路中的恶意注入，均属于开发者环境中默认可信、通常被安全策略放行的内容。一旦这些分发路径被劫持，恶意代码即可绕过边界防护，在开发主机与内网环境中静默执行。

在此类攻击模式下，传统安全机制面临明显局限：

防火墙关注的是边界访问行为，无法识别“合法应用发起的异常请求”；

基于特征的检测依赖已知恶意样本，对通过官方渠道分发的“干净文件”难以及时识别；

终端防护则往往将此类行为视为正常开发活动，从而放行敏感操作。

在此背景下，开发工具链本身正逐渐演变为新的攻击突破口。AI 框架、API 协作工具及开源组件已深度嵌入研发流程（如 LiteLLM、Apifox、ContextHub 等场景），一旦被利用，攻击者可直接获取 SSH 密钥、云服务凭据、Kubernetes 配置及本地代码等高价值资产，并以此为基础向云环境及集群侧进一步扩展。

这也意味着，供应链攻击的影响已不再局限于单一终端，而是具备向开发环境、云资源乃至业务系统扩散的能力，其风险从“单点失陷”演变为“体系性暴露”。

3. **检测实践：基于行为分析的供应链攻击预警能力**

供应链攻击依托合法应用与可信渠道开展，传统边界防护与特征检测难以发现，需从行为分析入手，对程序执行与网络通信进行联合建模。

通过主机行为与网络流量联动分析，可有效识别此类攻击。以 LiteLLM、Apifox、ContextHub 事件为例，攻击过程中会出现敏感文件访问、异常外联、可疑域名访问、应用启动即发起网络请求等可关联异常。

基于这些行为可还原攻击链：

LiteLLM 场景：监控 Python 进程读取 ~/.ssh/、~/.aws/ 等敏感路径，结合访问云元数据地址与恶意域名的外联行为，识别凭据窃取与数据外传。

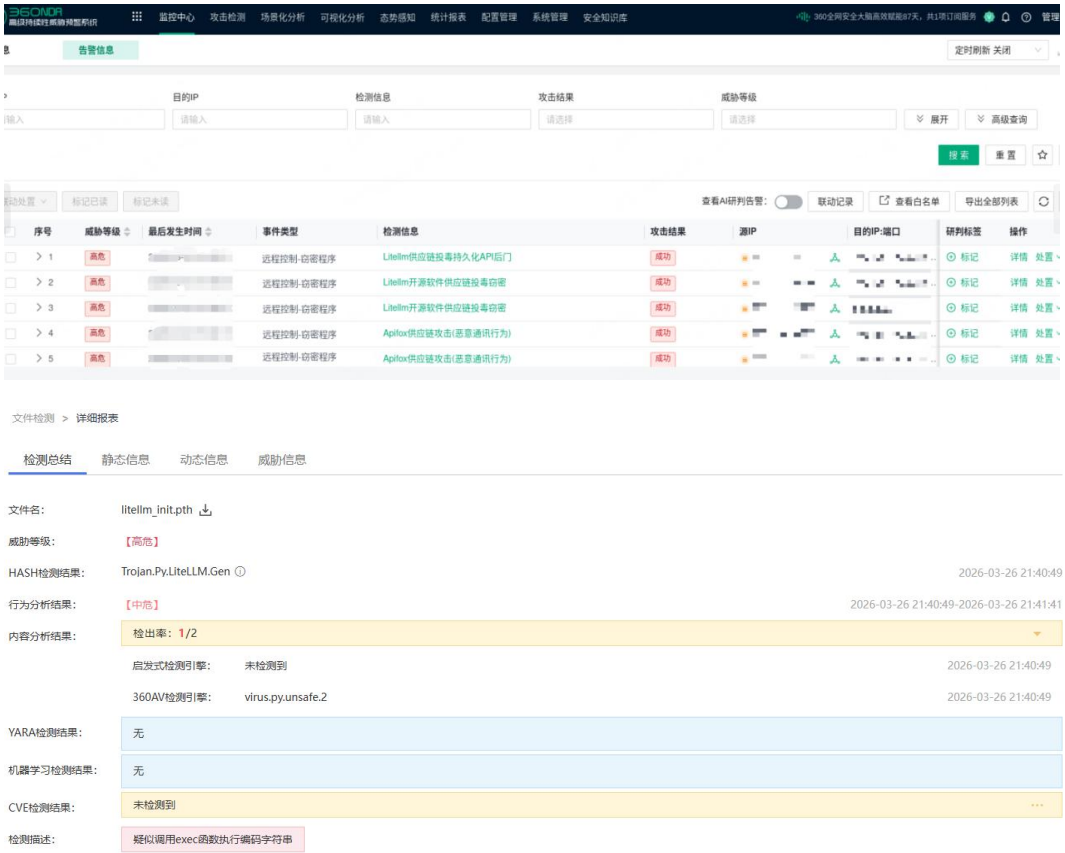
Apifox 场景：通过客户端启动阶段访问异常域名、携带特殊请求头、动态加载远程恶意 JS 等特征，发现恶意执行链。

ContextHub 场景：关注插件/组件加载与运行阶段的异常行为，例如：应用或插件进程在初始化时加载远程脚本或动态更新配置；访问非常规域名或第三方资源仓库；读取本地开发配置（如 .env、云凭据文件、Kubernetes 配置等）并伴随外联请求；以及插件执行链中出现的异常子进程或脚本执行行为。通过将插件加载行为+本地敏感访问+异常外联进行关联分析，可识别潜在的供应链投毒与数据外传风险。

通过 NDR 流量分析+沙箱动态分析协同，结合威胁情报与行为建模，可实现从单点异常到完整攻击链的精准检测。

该能力已在 360NDR（流量检测智能体）落地，可对异常外联、敏感信息读取、可疑执行链等关键行为实时监测、预警，并支持攻击路径还原与溯源。360 NDR 已覆盖 LiteLLM、Apifox、ContextHub 等典型供应链攻击场景，可在攻击早期发现并告警。

结合网络流量监测与沙箱行为分析，本次供应链相关攻击告警示例如下：



4. 关键 IOC 汇总

models.litellm.cloud

checkmarx.zone

apifox.it.com

apifox.it.com/public/apifox-event.js

apifox.it.com/event/0/log

apifox.it.com/event/2/log

cdn.openroute.dev

upgrade.feishu.it.com

system.toshinkyo.or.jp

ns.feishu.it.com

ns.openroute.dev

api.feishu.it.com

d.feishu.it.com

panel.feishu.it.com