TECHNICAL REPORT — TR-2026-01

Operation "OpenClaw"

Anatomy of an AI-Driven Cyberattack Against a Pharmaceutical Company

Phase 2 — Weaponization

ClawHub Supply Chain, LLM-Driven Ransomware and Prompt Injection Vector Preparation (D-15 to D-7)

Author: Fabrice Pizzi

Affiliation: Université Paris Sorbonne

Date: February 2026

Version: 8.0

Academic Publication – Information Systems Security & Artificial Intelligence

Module: S1-ISI5 – AI and Cybersecurity

Date: February 2026

Classification: Fictional scenario for educational purposes

⚠ WARNING

This document presents the second phase of a fictional but realistic attack scenario. It details the weaponization mechanisms of three families of offensive artifacts: a malicious supply chain skill, an LLM-driven ransomware, and indirect prompt injection payloads.

NO actual attack was conducted. PharmEurys SA does not exist.

Objective: identify and understand emerging risks related to AI agent security to improve defensive postures. AVERTISSEMENT

Abstract

This document constitutes the second installment of the Operation "OpenClaw" analysis and covers Phase 2 — Weaponization (D−15 to D−7), in the Cyber Kill Chain sense: after reconnaissance, the adversary prepares the offensive artifacts that will be deployed in subsequent phases. The analysis is structured around three axes corresponding to three families of weaponized artifacts.

Scenario object definitions: OpenClaw designates a fictional open-source autonomous AI agent (inspired by real tool/skill-based agent architectures) with terminal access, Slack and Outlook integrations, and a persistent conversational memory. ClawHub designates its community skill registry. PharmEurys SA is a fictional pharmaceutical mid-size company (~500 employees). All technical vulnerabilities and offensive tools described are documented in the public literature.

The analysis examines three families of offensive artifacts: (1) application supply chain compromise within the skill ecosystem (alignable with MITRE ATT&CK T1195.002 — Compromise Software Supply Chain), (2) LLM-driven ransomware (PromptLock) where a local language model generates variable payloads at runtime, and (3) indirect prompt injection payloads designed to hijack the behavior of a trusted AI agent via its legitimate data channels.

Reminder: This document models threat scenarios for risk analysis and derivation of defensive controls. It does not provide operational instructions for attack execution.

Keywords: supply chain attack, autonomous AI agent, malicious skill, weaponization, polymorphic ransomware, LLM-driven malware, indirect prompt injection, Promptware Kill Chain, OWASP LLM01:2025, MITRE ATT&CK T1195.002, MITRE ATLAS

Introduction: From Reconnaissance to Weaponization

Phase 1 of Operation OpenClaw (D−30 to D−15) produced actionable intelligence on the target PharmEurys SA: an exposed OpenClaw instance identified via querying Internet asset databases (Shodan), an organizational chart reconstructed via social graph mining, and the inference of a potential Fortinet VPN vulnerability (CVE-2024-55591). This intelligence now guides the development of offensive artifacts.

Phase 2 — Weaponization — materializes the transition from information collection to offensive artifact development. Within the Lockheed Martin Cyber Kill Chain framework, this phase transforms reconnaissance intelligence into deployable weapons: exploitation code, malicious payloads, delivery infrastructure.

In this scenario, the supply chain refers to the entire skill distribution pipeline within an AI agent ecosystem: community registry (ClawHub), discovery mechanism (search, ranking), installation (agent configuration), and runtime execution. Each stage of this pipeline constitutes a potential attack surface.

The specificity of this phase in the context of autonomous AI agents rests on three properties:

(a) The delivery vector (the skill published on a community registry) is natively integrated into the target's ecosystem, reducing dependence on traditional delivery vectors (phishing, exploit kits) and exploiting the implicit trust granted by users to community artifacts.

(b) The malicious payloads generated by LLM exhibit high structural variability due to the stochastic nature of language model inference, significantly complicating signature-based detection.

(c) Indirect prompt injection (OWASP LLM01:2025) enables hijacking a trusted AI agent's behavior without modification of its binary or configuration, by exploiting the processing of untrusted content in legitimate data channels.

2. Axis 1: Supply Chain Compromise via ClawHub

2.1 OpenClaw Skill Architecture and Attack Surface

In the OpenClaw ecosystem, a "skill" is a folder containing a SKILL.md file (metadata and instructions in natural language) along with optional scripts and associated resources [25]. The agent reads the SKILL.md at runtime and interprets its instructions, which can include directives for tool calls (terminal commands, file access, network calls) [43].

1Password emphasizes in their February 2026 analysis: "a markdown file is not content in an agent ecosystem — a markdown file is an installer" [25]. Users install skills by adding them to their agent configuration, granting SKILL.md instructions implicit trust at the level of the agent's execution context.

Critical clarification on execution privileges: the effective privilege level of a skill depends on the agent runtime implementation — execution policy, sandboxing, tool allowlisting, and human-in-the-loop confirmation. Some configurations allow unrestricted command execution ("yolo mode"), while others enforce restrictions. The following analysis models a scenario where execution controls are insufficient, a configuration documented as prevalent in initial deployments [12, 73].

The ClawHub registry functions as an open community marketplace. According to available public analyses, publication barriers are minimal (developer account, terms acceptance) — the platform does not implement systematic security review of uploaded skills at the time of writing [44, 45].

This attack surface is structurally analogous to that of software package registries (npm, PyPI, RubyGems), for which supply chain compromise is an established and documented attack vector (MITRE ATT&CK T1195.002). The specificity of the agent skill ecosystem lies in the additional attack surface of natural language instructions, exploitable via prompt injection.

2.2 The ClawHavoc Campaign: Empirical Threat Analysis

Koi Security conducted a large-scale audit of the ClawHub registry, reporting analysis of 2,857 skills and identification of 341 malicious entries, of which 335 were attributed to a coordinated campaign dubbed "ClawHavoc" [44]. A subsequent update raised the total to 824 malicious skills across 3,984 analyzed entries.

Table — Malicious Technique Families Observed in the ClawHavoc Campaign

Category	Count (initial audit)	General Mechanism	Attack Class

Crypto tools (Solana, Phantom)	111	Functional lure triggering infostealer installation via fake prerequisites*infostealer* via prérequis factices	Dropper / Social engineering
Video utilities	57	Password-protected archive containing a concealed malicious executable	Dropper / Payload packagingPayload packaging
Finance / social networks	51	Exfiltration of agent environment secrets via third-party endpointendpoint tiers	Credential exfiltrationcredentials
Trading bots	34	Remote access component concealed in functional code	Backdoor / Remote accessRemote access
CLI typosquats	29	Typographic variants of legitimate package names redirecting to malware	Typosquatting (T1195.002) (T1195.002)
Auto-updatersupdaters	28	Malware disguised as system update tool	Masquerading (T1036) (T1036)
Productivity integrations	17	Exploitation of elevated agent permissions to access documents and emails	Permission abuse / Data accessaccess
Fake security tools	~25	Skills claiming to scan vulnerabilities — identified in later updates prétendant scanner les vulnérabilités — identifiées dans des mises à jour ultérieures	Masquerading (T1036) (T1036)

Source: Koi Security, "ClawHavoc" campaign, 2026 [9]. The listed techniques are generic supply chain compromise families (functional lures, typosquatting, droppers, credential exfiltration) applicable to any community registry ecosystem.

Convergence of Prompt Injection and Traditional Malware (Snyk ToxicSkills)

In parallel, Snyk's ToxicSkills study highlights a marked convergence between prompt injection and traditional malware in malicious skills. Snyk reports that 91% of confirmed malicious skills combined at least one prompt injection technique with a traditional malware component (dropper, infostealer, backdoor).

This prompt injection + malware convergence constitutes a distinctive property of AI agent ecosystems compared to traditional package registries (npm, PyPI): the attacker simultaneously has access to two complementary attack surfaces — the agent's execution engine (via injected code) and the agent's reasoning engine (via injected instructions).

2.3 Kill Chain of the "What Would Elon Do?" Skill

Cisco reports having analyzed, using its AI Defense Skill Scanner tool, a community skill "What Would Elon Do?" ranked at the top of the ClawHub registry — analysis published in January 2026 [43]. The documented behaviors include:

- **Silent data exfiltration: the skill executed a curl command toward an attacker-controlled external infrastructure, exploiting the agent's terminal access to transmit environment data (keys, tokens, configuration) without user notification.**

- **Direct prompt injection: the SKILL.md instructions contained directives aimed at bypassing the agent's behavioral guardrails, redirecting its behavior beyond the skill's declared functional scope.**

Terminological clarification: these findings are security flaws identified in the behavior of a specific skill, not vulnerabilities in the CVE/product sense. They nonetheless illustrate structurally exploitable attack patterns within the current ClawHub ecosystem.

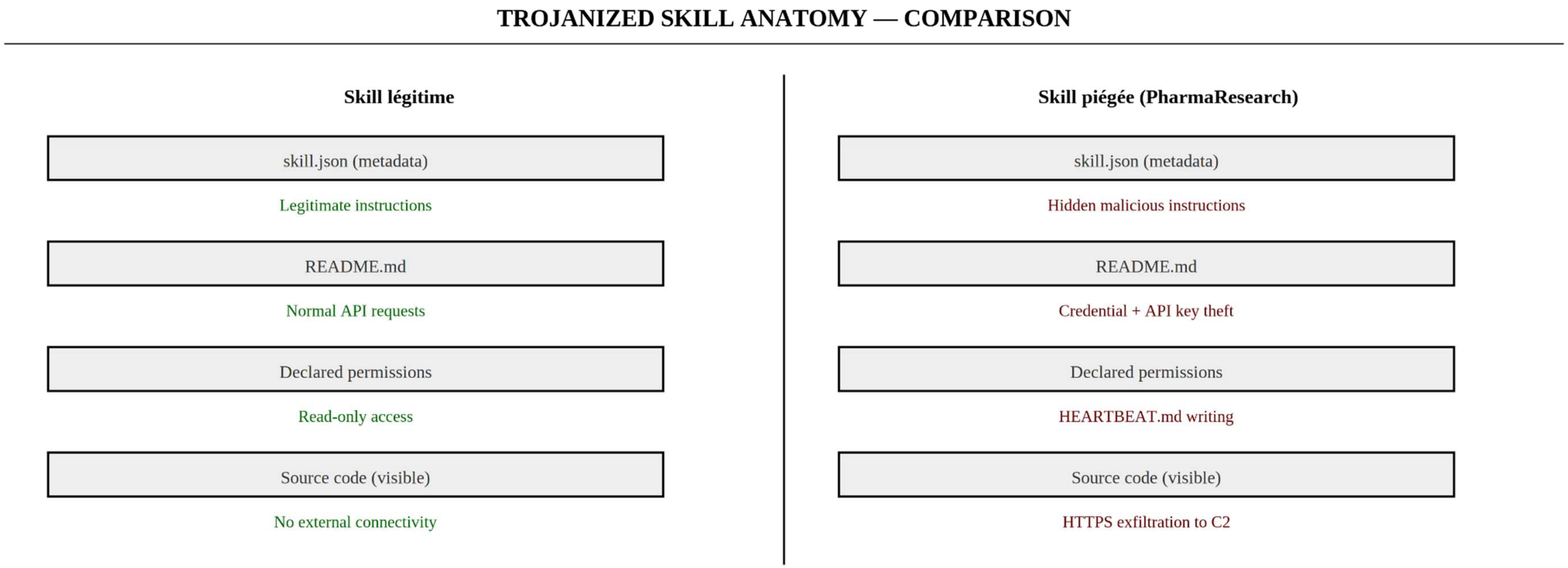
The positioning of this skill at the top of the ranking resulted from manipulation of the registry's ranking system. According to secondary analyses (AuthMind, iKangai), the ClawHub ranking mechanism was based on download metrics, which were susceptible to artificial inflation via automation [50].

This case study illustrates a supply chain risk specific to AI agent ecosystems: user trust in the most popular artifacts can be subverted by ranking manipulation, creating an implicit trust vector exploitable for mass distribution of malicious skills.

2.4 Design of the Malicious "PharmaResearch Assistant" Skill

Based on the actionable intelligence from Phase 1 — which identified R&D researchers using OpenClaw for pharmaceutical scientific monitoring activities — the attacker designs a supply chain artifact specifically targeting this user profile: a skill presenting itself as a pharmaceutical research assistant.

The malicious skill relies on a functional duality principle: each component fulfills a legitimate function visible to the user while embedding concealed malicious behavior. This approach exploits the implicit trust granted to functional skills in the ClawHub ecosystem.



TROJANIZED SKILL ANATOMY — COMPARISON

Skill légitime	Skill piégée (PharmaResearch)
skill.json (metadata)	skill.json (metadata)
Legitimate instructions	Hidden malicious instructions
README.md	README.md
Normal API requests	Credential + API key theft
Declared permissions	Declared permissions
Read-only access	HEARTBEAT.md writing
Source code (visible)	Source code (visible)
No external connectivity	HTTPS exfiltration to C2

Figure 9. Side-by-side comparison of a legitimate skill and the trojanized PharmaResearch Assistant skill. Surface files (skill.json, README, permissions) are identical; the difference lies in the system prompt instructions, invisible to the user without source code inspection.

Table — Component Classes and Attack Surfaces of a Malicious Skill

Component Class	Apparent Legitimate Function	Exploited Attack Surface	Applicable Defensive Control
Instruction file (SKILL.md)	Natural language instructions for the skill's declared operationskill	Injection of malicious operational directives in "documentation-like" instructions: conditional file exfiltration, environment reconnaissance, C2 callback	Systematic instruction review before installation, sandboxing of file/network operations, allowlist of authorized directivessandboxing des opérations fichier/réseau, *allowlist* de directives autorisées
Auxiliary scripts	Business functionalities (bibliographic search, report generation)génération de rapports)	C2 callback concealed in initialization routines, file system scan triggered by apparently legitimate functions [25]des fonctions apparemment légitimes [25]	Static code analysis, outgoing network call monitoring, script containerizationréseau sortants, conteneurisation des scripts
Direct prompt injection	—	Hijacking of agent guardrails: suppression of confirmation prompts for sensitive operations (network access, file manipulation)01:2025) [120]	Non-bypassable guardrails (hardcoded), separation of instruction and data channels*hardcoded*), séparation des canaux d'instruction et de données
Ranking manipulationranking	Popularity metrics displayed to users	Artificial inflation of download counts and reviews via automation, exploiting absence of anti-fraud verification in the registry [1]	Metric integrity verification, abnormal installation pattern detection, verified publisher skill signingskills par des éditeurs vérifiés

Note: the table above describes generic classes of malicious components observable in AI agent skill ecosystems, not a reproducible construction procedure. The specific mechanisms of each component are intentionally described at the conceptual level.

HTTPS Exfiltration Channel

The most critical property of this supply chain vector is the stealth of the exfiltration channel. In a non-sandboxed configuration (cf. section 2.1), the skill has legitimate network access via the agent's infrastructure, and its outgoing HTTPS calls are indistinguishable from legitimate API calls at the protocol level.

OWASP and MITRE Classification

The exploited vulnerability corresponds to OWASP LLM03:2025 — Supply Chain Vulnerabilities: "LLM supply chains are susceptible to vulnerabilities affecting data integrity, model, and deployment component security." In the case of a community skill ecosystem, the supply chain includes the registry, distribution, installation and execution mechanisms.

MITRE ATT&CK mapping: T1195.002 — Compromise Software Supply Chain: insertion of a malicious artifact into a trusted distribution mechanism (community skill registry) [25].

2.5 Registry Security Controls and Limits of Static Analysis

In response to documented supply chain risks (cf. section 2.3), OpenClaw announced the integration of a security control in partnership with VirusTotal: SHA-256 hash computation of each skill bundle and cross-referencing with the VirusTotal database. This control, described as a "first step" by the OpenClaw team, represents a hash and AV signature verification [52].

This type of control has structural limitations against several bypass classes:

- **Remote payloads: the artifact published on the registry does not directly contain malicious code, but downloads the payload from an external infrastructure at runtime (T1105 — Ingress Tool Transfer). The analyzed hash is clean.**

- **Conditional behaviors: the malicious payload triggers only under certain conditions (presence of specific files, particular network environment, time delay), rendering static analysis in an isolated environment (sandbox) ineffective.**

- **Linguistic attacks: prompt injections concealed in natural language instructions (SKILL.md) present no binary signature detectable by traditional antivirus engines — they are behavioral instructions, not executable code.**

Consequently, static analysis centered on the package (hash + AV scan + code analysis) must be considered necessary but insufficient. A defense-in-depth posture must add runtime controls (monitoring tool calls, egress traffic, behavioral anomalies) and human governance controls (systematic review before installation, verified publisher allowlisting).

3. Axis 2: Weaponization of Language Models — LLM-Driven Ransomware

3.1 From Static Binary to Dynamic Generation

The evolution of malware toward language model integration represents a paradigm shift: the malicious payload is no longer a static binary artifact analyzable before execution, but becomes partially or fully generated at runtime by a language model, introducing structural variability at each execution.

Taxonomy of LLM-driven architectures: three integration modes are distinguished in artifacts analyzed to date:

- **Locally accessible LLM:** the malware invokes a locally deployed model (via an Ollama-type interface) to generate scripts on the fly, without dependence on external infrastructure.
- **LLM via remote API**: the malware calls a cloud LLM (GPT-4 or equivalent) at runtime to generate code or commands, with the risk of leaving API call traces.
- **LLM hosted on third-party platform:** the malware queries a model deployed on a sharing platform (Hugging Face type) to obtain operational commands.

Table — Artifacts Integrating an LLM: Primary Sources and Defensive Effects

Artifact	Primary Source	LLM Integration Mode	Documented Behavior	Primary Defensive Effect
PromptLock	ESET (discovery, PoC) [42]PoC) [42]	Locally accessible LLM (Ollama) to generate malicious scripts at runtimeOllama) pour générer des scripts malveillants à l'exécution	Real-time Lua encryption script generation, high structural variability of produced payloadsLua de chiffrement en temps réel, variabilité structurelle élevée des charges produites	Reduces effectiveness of static signatures; requires hunting on prompts and LLM communicationshunting sur prompts et communications LLM
MalTerminal	SentinelLabs (primary analysis) (analyse primaire)	Runtime LLM API calls (GPT-4)	Runtime payload generation (ransomware, reverse shell); payload absent from initial binaryshell)	Complicates static analysis — payload not present before execution; runtime observability required (API keys, prompts, telemetry)payload non

			au moment de l'exécution; payload absent du binaire initial	présent avant exécution ; observabilité runtime requise (clés API, prompts, télémétrie)
LAMEHUG	CERT-UA (report), Splunk Threat Research (detailed analysis)Threat Research (analyse détaillée)	Hosted LLM (Hugging Face)Hugging Face)	Command generation for reconnaissance, collection and exfiltration — not ransomware but an LLM-driven command malware*il ne s'agit pas d'un ransomware mais d'un malware de commande LLM-driven*	Shifts IoCs toward the "prompt → command" flow and C2/exfiltration channels

Note on sources: Picus Security published an aggregated synthesis ("Malicious AI Exposed") covering PromptLock, MalTerminal and LAMEHUG [42]. This is a secondary source and does not constitute the original discovery of these artifacts.

Table — Malicious LLMs Oriented Toward Social Engineering (Distinct Category)

Tool	Primary Source	Architecture	Documented Capabilities
WormGPT 4 4	Unit 42, Palo Alto Networks [5]	GPT-J fine-tuned on malware data, commercialized via Telegramtuné sur données malware, commercialisé via Telegram	Advanced phishing/BEC generation, offensive scripts, reconnaissance automation [6]
KawaiiGPT	Unit 42 [5]	Guardrail-free variant (Grok/Mixtral base)Grok/Mixtral)	Exploitation code generation, filter bypass

Category distinction: WormGPT and KawaiiGPT are malicious (maligned) LLMs oriented toward social engineering and offensive tooling. They are not LLM-driven malware architectures in the sense of this section. They serve as general-purpose models for the attacker, not as runtime components of a specific malware.

3.2 Implications for Detection

The integration of an LLM into the malware execution cycle weakens approaches solely based on static signatures: the loader binary can be minimal and generic, while the actual malicious payload is generated at runtime — and varies with each execution.

This increased variability does not render signature-based detection obsolete — loader patterns, prompt strings, embedded API keys and network behaviors (calls to LLM endpoints) remain exploitable as IoCs (Indicators of Compromise) — but it requires the addition of at least three complementary detection layers:

- **Runtime observability: monitoring of LLM API calls (endpoints, frequency, volume), detection of API keys in running processes, analysis of intercepted prompts.**
- **Behavioral telemetry: detection of mass encryption patterns, abnormal file system access, post-generation data exfiltration.**
- **LLM artifact hunting: proactive search for configuration files containing malicious prompts, API keys for LLM services, or binaries invoking inference frameworks (Ollama, vLLM, llama.cpp).**

In the OpenClaw scenario, the attacker uses the PromptLock architecture (local LLM via Ollama) for the ransomware to be deployed in Phase 5, motivated by the absence of external API dependency (no traceable outgoing calls to a cloud provider).

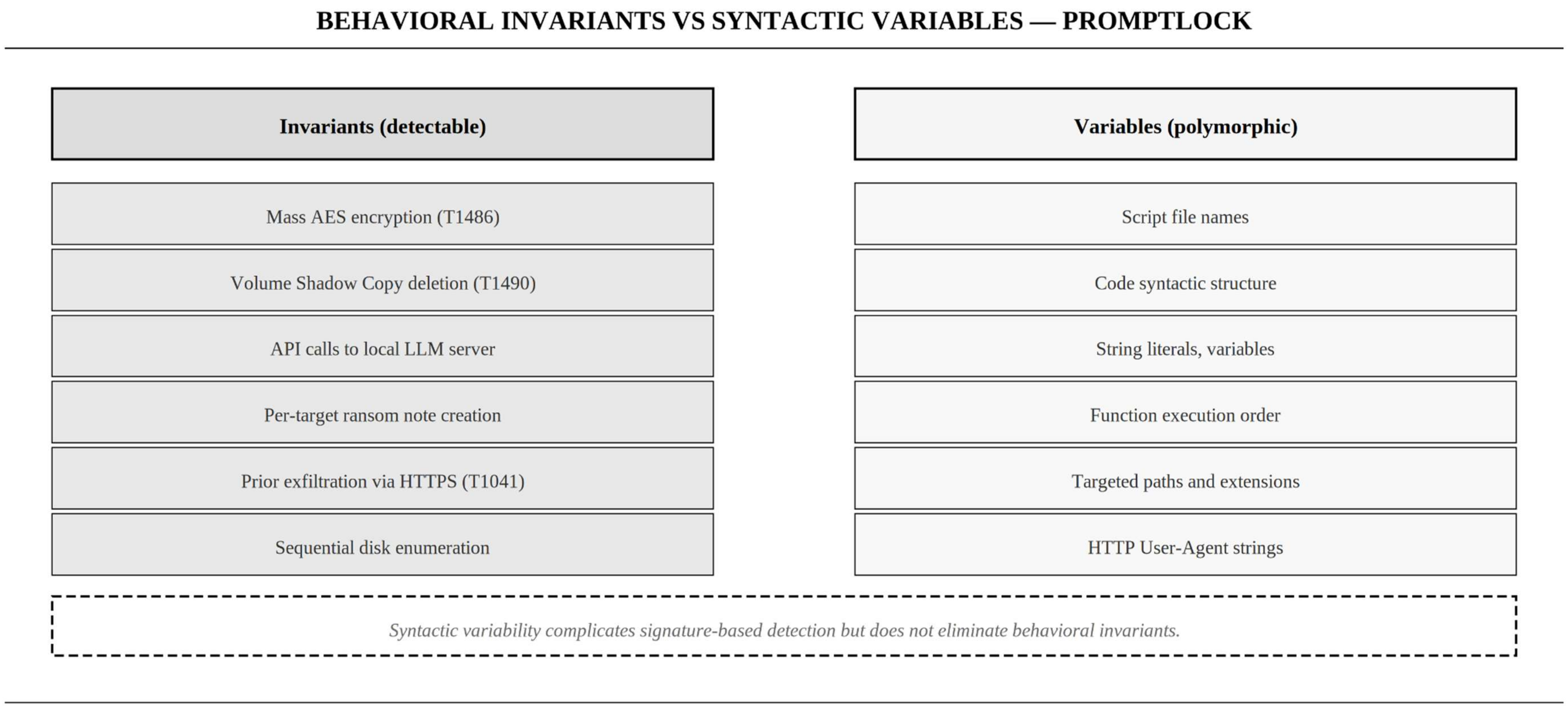
3.2 PromptLock Operational Architecture

Public analyses (ESET, Splunk) describe PromptLock as an artifact where a Go-written orchestrator binary queries a model accessible via the Ollama API to generate at runtime scripts adapted to the local environment. The generated scripts serve three documented functions:

- **Environment information collection: OS detection, volume listing, file identification by extension.**
- **Destructive actions: file encryption, with syntactic variability of generated scripts at each execution (variable names, control structures, calling methods).**
- **Recovery inhibition (MITRE ATT&CK T1490 — Inhibit System Recovery): alteration or deletion of recovery mechanisms, a frequent invariant of ransomware campaigns regardless of the generation mode.**
- **Content generation: drafting of ransom notes contextualized to the target's sector.**

3.3 LLM-Driven Variability vs Traditional Polymorphism: Invariants and Variables

Traditional ransomware uses metamorphism (binary rewriting) or polymorphism (payload encryption with variable decryption stub), leaving detectable structural patterns (headers, API calling conventions, characteristic strings).



BEHAVIORAL INVARIANTS VS SYNTACTIC VARIABLES — PROMPTLOCK

Invariants (detectable)	Variables (polymorphic)
Mass AES encryption (T1486)	Script file names
Volume Shadow Copy deletion (T1490)	Code syntactic structure
API calls to local LLM server	String literals, variables
Per-target ransom note creation	Function execution order
Prior exfiltration via HTTPS (T1041)	Targeted paths and extensions
Sequential disk enumeration	HTTP User-Agent strings

Syntactic variability complicates signature-based detection but does not eliminate behavioral invariants.

Figure 8. Comparison of PromptLock behavioral invariants and syntactic variables.
Invariants (left column) remain constant regardless of the LLM-generated variant and constitute priority detection signals for behavioral EDR solutions.

However, this variability is primarily syntactic — the overall behavior remains observable. SentinelLabs notes that these architectures "blur the boundary between code and conversation" (Cyber Insights 2026), but the runtime behavioral patterns (mass file creation, extension modification, backup deletion) remain detectable via behavioral telemetry.

Table — Invariants vs Variables in LLM-Driven Malware

Dimension	Traditional Malware (static)	LLM-Driven Malware (runtime)driven (runtime)
What varies	Decryption stub, polymorphic key	Full payload syntax (variables, control structures, methods), ransom note contentpayload (variables, structures de contrôle, méthodes), contenu des notes de rançon
What remains invariant	Functional logic of encrypted payloadpayload chiffré	Observable behavior (mass encryption, extension modification, backup deletion), embedded prompts, LLM endpoint/API key, orchestrator binaryendpoint/clé API LLM, binaire orchestrateur
Primary detection surface	Static signatures (hash, YARA, strings)	Behavioral telemetry (mass file creation/modification, process chains), hunting on LLM artifacts (prompts, API keys, Ollama calls)hunting sur artefacts LLM (prompts, clés API, appels Ollama)
Specific weakness	Binary mutation bypasses signatures	Dependence on LLM endpoint; prompts and access keys exploitable as IoC; failure if model unavailablel'endpoint LLM ; prompts et clés d'accès exploitables comme IoC ; échec si le modèle est indisponible

In the OpenClaw scenario, the attacker chooses the PromptLock architecture (local LLM via Ollama) to minimize network traces toward external APIs. The ransomware is prepared in Phase 2 but will not be deployed until Phase 5 (D+6), once lateral movement has secured complete access to the information system.

3.4 C2 Concealment in LLM API Traffic

Recent analyses report malware strains capable of camouflaging C2 communications by mimicking LLM API request structures (chat-completions type endpoints, characteristic headers and payloads), making traffic indistinguishable at the protocol level from legitimate calls to an AI service.

In the OpenClaw scenario, this camouflage becomes all the more plausible as the agent already makes regular calls to LLM services in the course of its normal operation. Organizations that have allowlisted OpenClaw's API traffic have thereby created a legitimate communication channel exploitable by an attacker.

This situation does not make detection impossible, but it reduces the effectiveness of purely perimeter-based controls. A WAF primarily protects HTTP server-side applications and is not positioned to analyze outgoing API calls from an agent. A generic outgoing HTTPS proxy will see encrypted traffic to authorized domains.

Detection necessarily shifts toward behavioral and telemetry signals:

- **Process origin of connections: an LLM call initiated by an auxiliary script of a skill (and not by the agent's main process) constitutes an identifiable anomaly via endpoint monitoring.**
- **Volume and periodicity anomalies: a beaconing pattern (regular intervals, constant volumes) differs from legitimate LLM calls which are typically triggered by stochastic user interactions.**
- **Rare domains and TLS fingerprints: calls to LLM endpoints not referenced in the agent's configuration, or presenting unusual TLS fingerprints (JA3/JA4), constitute indicators exploitable in threat hunting.**
- **Outbound/inbound ratio: an exfiltration generates an abnormally high outgoing data ratio compared to normal LLM API usage patterns (where requests are short and responses are long — the inverse of an exfiltration).**

These detection strategies align with MITRE recommendations for C2 over web protocols (HTTPS/WebSockets) and illustrate the necessity of endpoint + network + strict allowlist correlation to maintain visibility in environments where legitimate AI traffic coexists with potential C2 channels.

4. Axis 3: Preparation of Indirect Prompt Injection Vectors

4.1 Prompt Injection as an Architectural Vulnerability

Prompt injection (direct or indirect) occurs when inputs modify an LLM's behavior in a way unintended by the designer. OWASP classifies it as the number one risk for LLM applications (LLM01:2025 — Prompt Injection), a ranking maintained since the first version of the ranking in 2023.

The UK NCSC warns against an oversimplified analogy with SQL injection: the attacks share an intuition (injection of instructions into a data flow), but LLMs have no strict equivalents to parameterized queries or prepared statements that serve as definitive defenses for SQL injection.

Reported success rates in the literature vary significantly depending on the agentic architecture, model and evaluation protocol, but can exceed 80% in certain configurations with established techniques (ReAct chains, instruction concealment in documents).

4.2 The Promptware Kill Chain (C. Schneider, 2026)

C. Schneider proposes a Promptware Kill Chain in seven stages, describing how attacks initially qualified as prompt injection can evolve into multi-step mechanisms close to classical attack chains (lateral movement, persistence, exfiltration), constituting what he terms "promptware."

Stage	Designation	Description	Associated Defensive Control
1	**Initial Access**	Textual payload enters the LLM context via an external data channel (document, message, web page)payload textuel entre dans le contexte du LLM via un canal de	Input source filtering and classification, separation of data and instruction channels

		données externe (document, message, page web)	
2	**Privilege Escalation / Jailbreaking Escalation / Jailbreaking**	Bypassing the agent's behavioral guardrails, obtaining expanded tool access	Non-bypassable guardrails (hardcoded), jailbreak attempt monitoring, scope restrictions*hardcoded*), monitoring des tentatives de jailbreak, restrictions de scope
3	**Reconnaissance**	The compromised agent explores its environment: available tools, permissions, connectors, accessible data	Least privilege principle on tools, discovery call auditing
4	**Persistence**	Writing to the agent's long-term memory or configuration files to maintain altered behavior beyond the session	Persistent memory governance, configuration file integrity, modification alerts
5	**Command & Control**	Establishing a communication channel between attacker and compromised agent via legitimate channels	Egress control, monitoring of agent-initiated network calls, domain allowlistingegress, monitoring des appels réseau initiés par l'agent, *allowlist* de domaines
6	**Lateral Movement Movement**	Propagation via agent connectors (messaging, shares, connected services) to other users or systems	Inter-service permission segmentation, agent isolation, abnormal propagation detectioninter-services, isolation des agents, détection de propagation anormale
7	**Actions on Objective**	Data exfiltration, information manipulation, or triggering destructive actions	Sensitive operation monitoring, data transfer alerts, human validation controls

Source: C. Schneider, "The Promptware Kill Chain," February 2026 [120].

4.3 Crafting Payloads for Phases 3 and 4

Indirect injection payloads can be concealed in content ingested by the agent (messaging, documents, web pages) via textual obfuscation techniques — visually hidden content, invisible characters, or contextual embedding in legitimate text.

Persistence via HEARTBEAT.md

HiddenLayer (reported by specialized press) describes a scenario where an indirect injection in a web page — for example during a "summarize this page" request — causes OpenClaw to deposit persistent instructions in its working directory or configuration files.

exploitable: any modification of its content alters the agent's behavior in subsequent cycles. This mechanism corresponds to stage 4 (Persistence — memory and retrieval poisoning) of the Promptware Kill Chain.

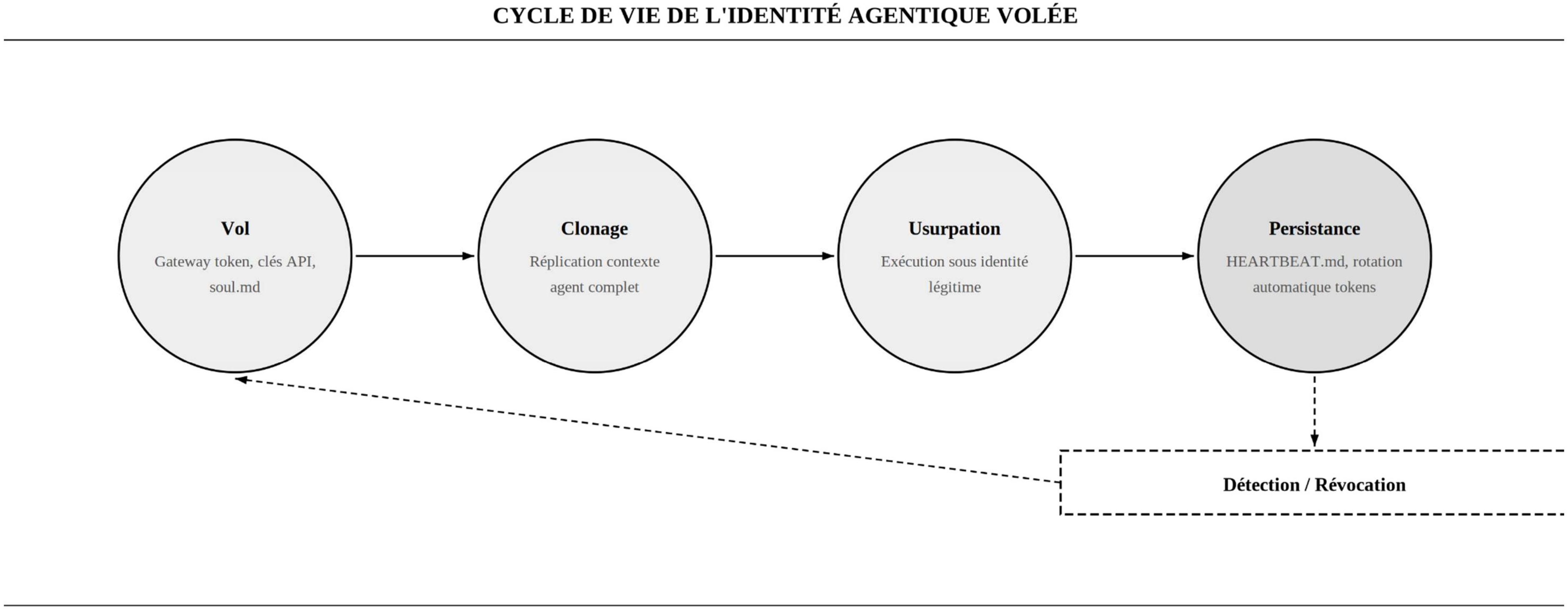

CYCLE DE VIE DE L'IDENTITÉ AGENTIQUE VOLÉE
Vol
Gateway token, clés API, soul.md
Clonage
Réplication contexte agent complet
Usurpation
Exécution sous identité légitime
Persistance
HEARTBEAT.md, rotation automatique tokens
Détection / Révocation

Figure 11. Cycle de vie de l'identité agentique volée. Du vol initial (tokens, clés, configuration) au clonage du contexte agent, puis à l'usurpation d'identité et à la persistance via HEARTBEAT.md. La boucle de détection/révocation (pointillés) représente le mécanisme défensif de rupture du cycle.

Persistent Memory Poisoning

Unit 42 (Palo Alto Networks) emphasizes that persistent memory of AI agents acts as a risk amplifier: an injection can produce lasting behaviors and delayed effects, the injection being executed once but its effects persisting across subsequent sessions.

This property mandates strict governance of persistent memory and ingested sources: audit of memory writes, integrity of agent configuration files, and restrictions on sources authorized to modify the agent's long-term memory.

Lateral Movement

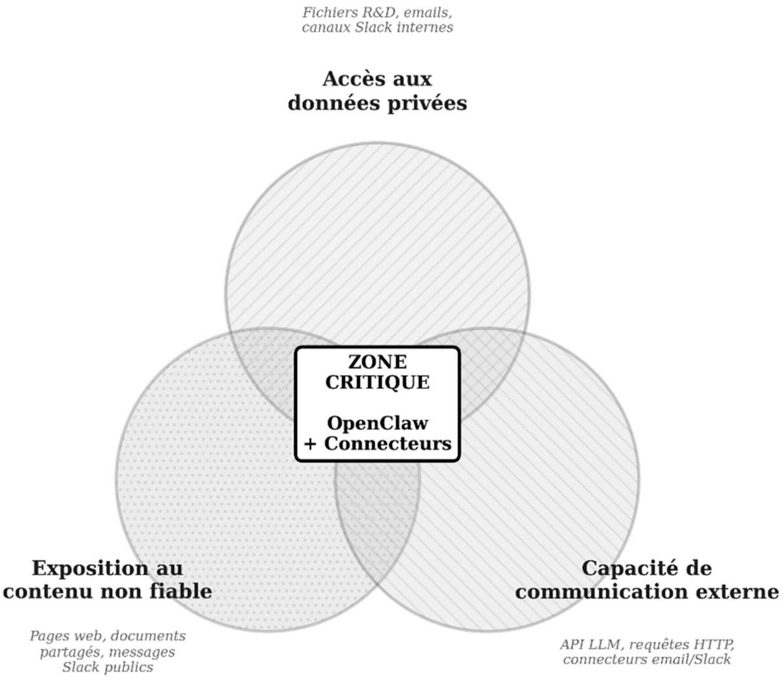
Sophisticated intrusions generally include a lateral movement phase, the attacker exploiting obtained access to propagate to other systems and users [112]. In the context of an AI agent, lateral movement can exploit the agent's connectors (Slack, email, shared drives) to propagate injections to other agents or users in the organization.

4.4 The "Lethal Trifecta" as a Necessary Condition

Willison proposes the lethal trifecta — access to private data, exposure to untrusted content, and external communication capability — as a particularly dangerous combination for AI agents [67, 127].

This trifecta should be understood as a risk amplifier — and, in many exfiltration scenarios, as a quasi-sufficient condition — rather than as a necessary condition for all attack types. Some attack classes (denial of service, information poisoning) can operate without full external communication capability.

Figure 12 — Trifecta létale de Willison appliquée à OpenClaw


Fichiers R&D, emails, canaux Slack internes
Accès aux données privées
ZONE CRITIQUE
OpenClaw + Connecteurs
Exposition au contenu non fiable
Pages web, documents partagés, messages Slack publics
Capacité de communication externe
API LLM, requêtes HTTP, connecteurs email/Slack

La co-présence des trois propriétés crée la condition nécessaire à l'exfiltration par prompt injection.

Source : S. Willison (2026), adapté au scénario MediFrance SA

In the case of OpenClaw, the simultaneous presence of connectors and tools (Slack, Outlook, terminal), external ingested sources (web pages, documents, messages), and network capabilities (API calls, curl) creates a configuration that meets the three properties of the trifecta in its strictest interpretation.

Several analyses characterize prompt injection in agent systems as a major threat to trust in AI systems deployed in enterprise [25][120]. (The formulation "existential threat" employed by some secondary sources is not adopted here.)

The following table synthesizes the offensive artifacts prepared in Phase 2 and correlates them with MITRE ATT&CK (Enterprise), MITRE ATLAS (AI-specific) and OWASP Top 10 for LLM Applications 2025 taxonomies.

Table — Phase 2 Weaponization Matrix: MITRE ATT&CK / ATLAS / OWASP Taxonomy

Offensive Artifact	Technique	ID	Description	OWASP LLM 2025	Mapping Note
Malicious skill on community registry piégée sur	Supply Chain Compromise: Software Chain	**T1195.002**	Insertion of a malicious artifact into a trusted distribution mechanism (skill	**LLM03 (Supply Chain Vulnerabilities)** (Supply	Direct mapping — T1195.002 explicitly covers

registre communautaire	Compromise: Software		marketplace)skills)	Chain Vulnerabilities)	software distribution compromise
Ranking manipulationranking	Stage CapabilitiesCapabilities	**T1608 (parent level)** (niveau parent)	Positioning and visibility of an offensive capability on the registry via artificial metric inflation	—	*T1608.006 (SEO Poisoning) specifically targets search engines and does not apply to internal marketplace ranking. Parent T1608 is used as closest mapping.Poisoning) vise spécifiquement les moteurs de recherche et ne s'applique pas au ranking interne d'une marketplace. Le niveau parent T1608 est utilisé faute de sous-technique plus précise. ATT&CK ne couvre pas nativement le « marketplace ranking fraud ».*
Direct prompt injection (in SKILL.md)	LLM Prompt Injection	**AML.T0051 (ATLAS)** (ATLAS)	Malicious instructions integrated into the skill's instruction file, altering agent behavior upon installationskill, altérant le comportement de l'agent dès l'installation	**LLM01 (Prompt Injection)** (Prompt Injection)	AML.T0051 covers prompt injection broadly. The direct/indirect distinction is qualified in the description per OWASP LLM01.la prompt injection au sens large. *La distinction direct/indirect est qualifiée dans la description conformément à OWASP LLM01, qui définit les deux variantes.*

Indirect injection payloads (messaging, documents, web pages) d'injection indirecte (messagerie, documents, pages web)	LLM Prompt Injection	**AML.T0051 (ATLAS)** (ATLAS)	Malicious content concealed in legitimate data channels (Slack, email, documents), later ingested by the agent	**LLM01 (Prompt Injection)** (Prompt Injection)	*No officially published ATLAS sub-technique for "indirect prompt injection" to date. AML.T0051 is used with "indirect" qualification.*
LLM-driven ransomware (PromptLock)driven (PromptLock)	Data Encrypted for ImpactEncrypted for Impact	**T1486**	Data encryption for ransom, with dynamic encryption script generation by local LLMdynamique des scripts de chiffrement par LLM local	—	Direct mapping — T1486 covers data encryption for impact, regardless of payload generation modeindépendamment du mode de génération du payload
Backup neutralization	Inhibit System Recovery System Recovery	**T1490**	Alteration or deletion of recovery mechanisms (backups, snapshots, restore points)	—	Direct mapping — frequent invariant of ransomware campaigns
C2 concealment in LLM API traffic	Application Layer Protocol: Web Protocols	**T1071.001**	C2 communications and exfiltration encapsulated in HTTPS requests mimicking legitimate LLM API format	—	Consistent mapping — T1071.001 covers C2 via web protocols (HTTP/HTTPS). LLM API traffic camouflage is a special case of this technique.
Persistent memory poisoning	*No confirmed official ATLAS ID*	—	Injection of persistent instructions into the agent's long-term memory, producing lasting and delayed altered behaviors	**LLM01 (Prompt Injection)** (Prompt Injection)	*AML.T0054 (LLM Jailbreak — Direct) does not correspond to memory poisoning per Microsoft source. In the absence of an official ATLAS ID for "memory poisoning," the technique is mapped to OWASP*

					LLM01.retrieval poisoning », la technique est rattachée à OWASP LLM01 et à l'étape 4 (Persistence) de la Promptware Kill Chain (C. Schneider, 2026).

5. Synthesis: Operational Arsenal at D-7

Table — Summary of Artifacts Prepared in Phase 2

Artifact	Objective	Planned Deployment Phase	Prerequisites	Applicable Defensive Control	Evidence Level
Malicious skill on community registry piégée sur registre communautaire	Supply chain vector: initial access via trusted marketplacesupply chain : accès initial via marketplace de confiance	Phase 3 (D-Day)	Registry without systematic signing/review, R&D users in Shadow AI situation	Skill code review, cryptographic signing, sandboxing, verified publisher allowlistingskills, signature cryptographique, sandboxing, *allowlist* d'éditeurs vérifiés	Prospective scenario based on documented components (Koi Security, Cisco, Snyk)Koi Security, Cisco, Snyk)
PromptLock engine (local LLM / Ollama)PromptLock (LLM local / Ollama)	Dynamic generation of ransomware payloads with high syntactic variability	Phase 5 (D+6)	Locally accessible LLM model, orchestrator binary	Hunting on LLM artifacts (prompts, API keys, Ollama calls), behavioral detection (mass encryption)Ollama), détection comportementale (chiffrement en masse)	Documented PoC (ESET, SentinelLabs, Splunk) documenté (ESET, SentinelLabs, Splunk)

Indirect injection payloads d'injection indirecte	Agent hijacking and inter-connector propagation via legitimate data channelsinter-connecteurs via canaux de données légitimes	Phase 4 (D+1 to D+5)	Active messaging/document integrations, absence of ingested source filtering	Data/instruction separation, ingested content filtering, tool call monitoring*tool calls*	Documented techniques (C. Schneider, HiddenLayer, Unit 42)HiddenLaye, Unit 42)
C2 infrastructure	Reception of exfiltrated data, remote command	Phases 3 to 5	Authorized outgoing HTTPS channel, LLM API traffic in allowlist*allowlist*	Strict egress control, beaconing detection, rare domain analysis, TLS fingerprintsegress strict, détection de beaconing, analyse de domaines rares, empreintes TLS	Documented pattern (secondary sources)

Note: the table describes artifact classes and their associated controls, not construction procedures. The statuses "prepared / prototype / elaborated scenario" reflect the narrative level of the scenario, not technical maturity levels.

5.1 Defensive Implications

The convergence of three trends — (i) skill registries equivalent to a supply chain attack surface, (ii) increased payload variability via LLM-assisted generation, and (iii) prompt injection as an architectural vulnerability without definitive fix — requires a defense-in-depth approach combining perimeter, endpoint and behavioral controls.

Terminological clarification: EDRs (Endpoint Detection and Response) are endpoint controls, not perimeter controls. The analysis therefore distinguishes: perimeter controls (WAF, IPS, network filtering), endpoint controls (EDR, antivirus, file system monitoring), and behavioral/AI-specific controls (prompt analysis, tool call monitoring, egress monitoring).

As a forecast, Michael Freeman (Armis) estimates that "by mid-2026, at least one major global enterprise will suffer a breach caused or significantly facilitated by an agentic AI system."

5.2 Transition to Phase 3 — Delivery and Intrusion (D-Day)

The next phase will analyze delivery mechanisms in the OpenClaw ecosystem and control points enabling kill chain interruption: human validation before skill installation, egress traffic control, and behavioral monitoring of agent actions.

References

Note: Numbering [41] to [75], continuing from Phase 1 ([1] to [40]).

[41] Lockheed Martin, « Cyber Kill Chain Framework ». https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[42] 1Password, « From magic to malware: How OpenClaw's agent skills become an attack surface », février 2026. https://1password.com/blog/from-magic-to-malware

[43] Cisco AI Threat & Security Research, « Personal AI Agents like OpenClaw Are a Security Nightmare », janvier 2026. https://blogs.cisco.com/ai/personal-ai-agents-like-openclaw-are-a-security-nightmare

[44] Koi Security, O. Yomtov, « ClawHavoc: 341 Malicious Clawed Skills Found by the Bot They Were Targeting », février 2026 (maj : 824 skills). https://www.koi.security/blog/clawhavoc

[45] PointGuard AI, « OpenClaw's 230 Malicious Skills: What Agentic AI Supply Chains Teach Us », février 2026. https://www.pointguardai.com/ai-security-incidents/openclaw-clawhub-malicious-skills-supply-chain-attack

[46] The Hacker News, « Researchers Find 341 Malicious ClawHub Skills Stealing Data from OpenClaw Users », février 2026. https://thehackernews.com/2026/02/researchers-find-341-malicious-clawhub.html

[47] Astrix Security, « OpenClaw & MoltBot: The First AI Agent Security Nightmare », février 2026. https://astrix.security/learn/blog/openclaw-moltbot

[48] eSecurity Planet, « Hundreds of Malicious Skills Found in OpenClaw's ClawHub », février 2026. https://www.esecurityplanet.com/threats/hundreds-of-malicious-skills-found-in-openclaws-clawhub/

[49] Snyk, « ToxicSkills: 91% of malicious ClawHub skills combined prompt injection with traditional malware », février 2026. Réf. Barrack.ai/TechInformed.

[50] J. O'Reilly (Dvuln), démonstration manipulation ranking ClawHub, février 2026. Réf. SecurityWeek/Barrack.ai.

[51] OWASP, « LLM01:2025 Prompt Injection » et « LLM05:2025 Supply Chain Vulnerabilities », Top 10 for LLM Applications 2025. https://genai.owasp.org/

[52] TechInformed, « OpenClaw adds VirusTotal scanning for ClawHub skills », février 2026. https://techinformed.com/openclaw-adds-virustotal-scanning-for-clawhub-skills/

[53] Picus Security, « Malicious AI Exposed: WormGPT, MalTerminal, and LameHug » (PromptLock), décembre 2025. https://www.picussecurity.com/resource/blog/malicious-ai-exposed

[54] SecurityWeek, « Cyber Insights 2026: Malware and Cyberattacks in the Age of AI » (SentinelOne, Armis), février 2026. https://www.securityweek.com/cyber-insights-2026-malware

[55] ESET, analyses PromptLock et malware piloté par LLM, décembre 2025–février 2026. Réf. Picus Security.

[56] Palo Alto Networks Unit 42, « The Dual-Use Dilemma of AI: Malicious LLMs » (WormGPT 4, KawaiiGPT), novembre 2025. https://unit42.paloaltonetworks.com/dilemma-of-ai-malicious-llms/

[57] CATO Networks, « WormGPT returns: New malicious AI variants built on Grok and Mixtral », juin 2025. https://www.csoonline.com/article/4008912

[58] Akamai, « What We Do In The Shadow (AI): New Malware Strain Vamps Up », novembre 2025. https://www.akamai.com/blog/security-research/

[59] OWASP, « Top 10 for Agentic Applications 2026 ». Réf. C. C. Schneider(2026).

[60] HackerNoob, « Prompt Injection Attacks Against LLM Agents: The Complete Technical Guide for 2026 ». Taux de réussite 66,9%-84,1%. Corroboré NCSC UK.

[61] Lakera, « Indirect Prompt Injection: The Hidden Threat Breaking Modern AI Systems », 2025-2026. https://www.lakera.ai/blog/indirect-prompt-injection

[62] C. C. Schneider, « From LLM to agentic AI: prompt injection got worse » (« Promptware Kill Chain », Christian schneider, 2026), février 2026. https://christian-schneider.net/blog/prompt-injection-agentic-amplification/

[63] Hudson Rock, « Infostealer Steals OpenClaw AI Agent Configuration Files and Gateway Tokens » (soul.md, openclaw.json, device.json), février 2026. Via The Hacker News.

[64] CrowdStrike, « Indirect Prompt Injection Attacks: Hidden AI Risks », décembre 2025. https://www.crowdstrike.com/en-us/blog/indirect-prompt-injection-attacks-hidden-ai-risks/

[65] HiddenLayer, démonstration injection via HEARTBEAT.md dans OpenClaw, février 2026. Réf. TechInformed.

[66] S. Mishra, S.P. Morgan, Palo Alto Networks Unit 42, « Persistent memory as accelerant for stateful, delayed-execution attacks » et « Lethal Trifecta », février 2026. Réf. The Hacker News.

[67] S. Willison, « The Lethal Trifecta of AI agents », réf. Palo Alto Networks Unit 42 et TechInformed, février 2026.

[68] MDPI Information, « Prompt Injection Attacks in Large Language Models and AI Agent Systems: A Comprehensive Review », janvier 2026. https://www.mdpi.com/2078-2489/17/1/54

[69] Debenedetti et al., « AgentDojo: A Dynamic Environment for Evaluating Attacks and Defenses for LLM Agents », 2024. ArXiv.

[70] 4sysops, « Scan OpenClaw agent skills for security vulnerabilities with the Cisco AI Skill Scanner », février 2026. https://4sysops.com/archives/scan-openclaw-agent-skills/

[71] Sophos, « The OpenClaw experiment is a warning shot for enterprise AI security », février 2026. https://www.sophos.com/en-us/blog/the-openclaw-experiment-is-a-warning-shot

[72] Menlo Security, « Predictions for 2026: Why AI Agents Are the New Insider Threat », janvier 2026. https://www.menlosecurity.com/blog/predictions-for-2026

[73] Barrack.ai, « OpenClaw is a Security Nightmare — Here’s the Safe Way to Run It », février 2026. https://blog.barrack.ai/openclaw-security-vulnerabilities-2026/

[74] MITRE, « ATLAS – Adversarial Threat Landscape for AI Systems ». https://atlas.mitre.org/

[75] StrongestLayer, analyses IBM Research / Harvard (statistiques phishing IA), 2025. Réf. Phase 1.

Note: the following references are defined in the bibliography of another phase of the document. They are reproduced here to allow autonomous reading of each phase.

[1] Application of OSINT Methods in Ensuring Cybersecurity, IPSIT Transactions on Internet Research, juillet 2025. https://ipsitransactions.org/journals/papers/tir/2025jul/p5.pdf
→ *Defined in Phase 1*
[9] Cisco AI Threat & Security Research, « Personal AI Agents like OpenClaw Are a Security Nightmare », janvier 2026. https://blogs.cisco.com/ai/personal-ai-agents-like-openclaw-are-a-security-nightmare
→ *Defined in Phase 1*
[11] SecurityWeek, « Vulnerability Allows Hackers to Hijack OpenClaw AI Assistant » (CVE-2026-25253, J. O'Reilly/Dvuln), février 2026. https://www.securityweek.com/vulnerability-allows-hackers-to-hijack-openclaw-ai-assistant/
→ *Defined in Phase 1*
[12] Barrack.ai, « OpenClaw is a Security Nightmare — Here's the Safe Way to Run It », février 2026. https://blog.barrack.ai/openclaw-security-vulnerabilities-2026/
→ *Defined in Phase 1*
[13] Hudson Rock, « Infostealer Steals OpenClaw AI Agent Configuration Files and Gateway Tokens », via The Hacker News, février 2026.
→ *Defined in Phase 1*
[25] MITRE ATT&CK, « Active Scanning: Vulnerability Scanning », Sub-technique T1595.002. https://attack.mitre.org/techniques/T1595/002/
→ *Defined in Phase 1*
[26] Recorded Future, « What is Banner Grabbing? Tools and Techniques Explained ». https://www.recordedfuture.com/threat-intelligence-101/tools-and-techniques/banner-grabbing
→ *Defined in Phase 1*
[112] CrowdStrike, « Indirect Prompt Injection Attacks: Hidden AI Risks » (mouvement latéral via agents compromis), décembre 2025. https://www.crowdstrike.com/en-us/blog/indirect-prompt-injection-attacks-hidden-ai-risks/
→ *Defined in Phase 4*
[120] C. Schneider (2026), Promptware Kill Chain. OWASP Top 10 for Agentic Applications 2026, ASI01 Agent Goal Hijack. https://christian-schneider.net/blog/prompt-injection-agentic-amplification/
→ *Defined in Phase 4*
[121] InstaTunnel, « Prompt-to-Insider Threat: When AI Agents Become Double Agents ». CVE-2025-32711 EchoLeak (M365 Copilot, CVSS 9.3), février 2026. https://instatunnel.my/blog/prompt-to-insider-threat/
→ *Defined in Phase 4*
[122] HackerNoob / Information 2026, 17(1), 54, « Prompt Injection Attacks in LLM and AI Agent Systems: A Comprehensive Review » (taux succès 66,9–84,1 %, 73 % déploiements affectés). doi:10.3390/info17010054
→ *Defined in Phase 4*
[127] S. Willison, « AI agents have a lethal trifecta of risks » (private data + untrusted content + external communication).
→ *Defined in Phase 4*
[133] Lakera, « The Year of the Agent: Q4 2025 Attacks » (attaques indirectes < tentatives que directes, system prompt extraction). https://www.lakera.ai/blog/the-year-of-the-agent
→ *Defined in Phase 4*
[135] OpenAI, « Understanding prompt injections: a frontier security challenge », janvier 2026. https://openai.com/index/prompt-injections/
→ *Defined in Phase 4*
[154] VikingCloud, « 46 Ransomware Statistics 2026 ». Coût total 1,8–5 M$/incident. https://www.vikingcloud.com/blog/ransomware-statistics
→ *Defined in Phase 5*
[159] Cisco, « State of AI Security 2025 Report » (34 % entreprises avec contrôles IA spécifiques, <40 % tests réguliers).

→ *Defined in Phase 5*
[160] OWASP, Top 10 for Agentic Applications 2026. Sandboxing agentique, moindre privilège, audit trafic sortant.
→ *Defined in Phase 5*