76 %

des tool calls dans les agents IA n'ont **aucune protection**

Reality Check — Un benchmark d'analyse statique sur 19 codebases open-source d'agents IA

16 repos Python · 3 codebases TypeScript · 16 500+ fichiers source scannés

Scanner	diplomat-agent v0.2.0 (Python) · diplomat-agent-ts v0.1.0 (TypeScript)
Date	15 mai 2026
Licence	Apache 2.0
Méthode	Analyse statique AST uniquement — zéro dépendance runtime (Python)
Contact	contact@diplomat.run · diplomat.run

EN UN COUP D'ŒIL

Résumé du benchmark

76 %	83 %	77 %
Python — sans garde	TypeScript — sans garde	Combiné — sans garde
14 repos valides, 7 029 tool calls	4 scopes, 669 tool calls	19 codebases, 16 500+ fichiers

Écosystème	Repos	Fichiers	Tool calls avec effets de bord	Sans garde
Python	16 (14 valides)	~5 100 .py	**7 029**	**5 344 (76 %)**
TypeScript	3 codebases, 4 scopes	11 379 .ts/.tsx	669	553 (83 %)
Combiné	**19 codebases**	**~16 500**	**7 698**	**5 897 (77 %)**

Le chiffre de 76 % est le finding principal — Python est l'écosystème le plus mature, avec le dataset le plus large. Le benchmark TypeScript confirme le même pattern dans un second écosystème : les frameworks et applications d'agents shippent sans gardes intra-fonctionnelles, par convention plus que par négligence.

À LIRE SI VOUS ÊTES…

RSSI / Head of Security	Un inventaire défendable de la surface d'effets de bord des agents IA en 2026. Sert à cadrer la discussion de gouvernance agentique avec l'équipe technique avant qu'un incident ne le fasse à votre place.
Platform / Infra Lead	Une recette de scan reproductible (pip install, npm install) et une checklist pré-déploiement qui mappe les catégories d'effets de bord aux gates runtime.
Fondateur / Product	Une manière de localiser la surface non protégée de votre propre agent, avant mise en production, en moins de cinq minutes.
Mainteneur open-source	Une baseline de gouvernance + la convention d'annotation (# checked:ok) pour acquitter les gardes que le scanner ne peut pas voir.

DÉFINITIONS

Ce que « sans garde » signifie — et ne signifie pas

> *« Sans garde » signifie : la fonction contient au moins un effet de bord (DB write, appel HTTP, subprocess, paiement, email, etc.) ET aucune garde détectée dans le corps de la même fonction.*

Un tool call sans garde n'est pas automatiquement une vulnérabilité. Les contrôles d'infrastructure (API gateways, IAM, network policies) et le code de couche service (middleware, décorateurs, validation) que le scanner ne peut pas voir peuvent couvrir une grande partie de ces cas.

Le chiffre de 76 % est un inventaire, pas un score de vulnérabilité. Il vous dit combien de surface d'effets de bord existe dans la codebase avant que quelqu'un ne décide quoi en faire.

Verdict	Action
CORRIGER	Ajouter une garde à l'intérieur de la fonction.
GATER	Router via diplomat-gate pour une décision CONTINUE / REVIEW / STOP.
ACQUITTER	Annoter # checked:ok — protected by [endroit] pour traçabilité.

Le scanner fournit l'inventaire. Votre équipe prend la décision — au design time, pas après un incident.

LIMITES CONNUES

Ce que nous ne voyons pas

Limite	Ce que cela signifie
Chaînes d'appels inter-fichiers	Si une garde vit dans la fonction A qui appelle la fonction B qui exécute l'effet de bord, le scanner voit B comme sans garde. L'analyse de call graph est sur la roadmap v0.2.
Aliases d'import (Python)	import requests as r puis r.post() n'est pas détecté. Limite connue.
Modèles ORM ré-exportés (TypeScript)	Mongoose / TypeORM / Drizzle distribués sur plusieurs fichiers passent à travers.
Enregistrement dynamique de tools	Les tools enregistrés dynamiquement à l'exécution sont invisibles. By design — analyse statique.

Absence remarquable : Zéro annotation # checked:ok sur les 19 codebases. Ce n'est pas parce que le code est mauvais — c'est parce que la convention d'annotation de gouvernance intra-fonctionnelle n'a pas encore été adoptée. Ce rapport est, entre autres, un argument pour en adopter une.

BENCHMARK PYTHON

diplomat-agent v0.2.0 — 16 codebases

16 codebases Python open-source d'agents IA. 14 ont produit des résultats valides (AIHawk et LangChain-community ont échoué au scan et sont exclus de tous les comptes).

Repo	Type	Tool calls	Sans garde	%	Durée
PraisonAI	Framework	1 028	911	89 %	~2s
CrewAI	Framework	348	273	78 %	~1s
Skyvern	Application	452	345	76 %	~2s
AutoGPT	Application	464	355	76 %	~2s
Dify (backend)	Platform	1 009	759	75 %	~3s
Khoj	Application	181	125	69 %	~1s
SurfSense	Application	315	165	52 %	~1s
OpenAI Agents (Python)	SDK examples	93	78	84 %	<1s
MetaGPT	Framework	205	186	91 %	~1s
Browser-use	Application	267	230	86 %	~1s
OpenAgents	Application	178	177	99 %	~1s
FinRobot	Application	69	55	80 %	<1s
Open-SWE	Application	35	34	97 %	<1s
GPT-Researcher	Application	31	27	87 %	<1s
Composio	Platform	28	26	93 %	<1s
Stripe Agent Toolkit	Benchmark	26	14	54 %	<1s
TOTAL (14 valides)	—	**7 029**	**5 344**	**76 %**	—

Par type de repo

Type	Repos	Sans garde (moy.)	Lecture
Framework	3	83 %	By design — les frameworks exposent des primitives et délèguent la gouvernance à l'application.
Application	9	80 %	Ces équipes ont construit le produit. Les gardes étaient à elles de les ajouter.
Platform	2	78 %	Même pattern que les applications.
Exemples SDK	1	84 %	Code de référence que les équipes copient-collent en production. Le chiffre qui se propage.
Benchmark	1	54 %	Environnement de test. Non représentatif.

BENCHMARK TYPESCRIPT

diplomat-agent-ts v0.1.0 — 3 codebases, 4 scopes

Repo (scope)	Type	Commit	Fichiers TS	Tool calls	Sans garde	%
OpenClaw (src/)	Application	49d9996d	7 874	419	332	**79 %**
Mastra (packages/)	Framework	38b87964	2 777	185	162	88 %
OpenAI Agents JS (packages/)	Framework	629d35af	426	33	31	94 %
OpenAI Agents JS (examples/)	Exemples	629d35af	302	32	28	88 %
TOTAL	—	—	**11 379**	**669**	**553**	**83 %**

FINDINGS PAR CATÉGORIE

Les deux écosystèmes — effets de bord détectés

Python (16 repos)

Catégorie	Sans garde	Repos	Ce que ça détecte
DB write	3 260	15/16	session.commit(), cursor.execute(INSERT/UPDATE), ORM .create()
DB delete	1 305	14/16	session.delete(), .drop(), cursor.execute(DELETE)
HTTP write	968	14/16	requests.post(), httpx.post(), écritures aiohttp
Subprocess / exec	697	10/16	subprocess.run(), os.system(), exec(), eval()
Publish / storage	609	7/16	s3.put_object(), .publish(), .upload()
Appels LLM	464	8/16	client.chat.completions.create(), anthropic.messages.create()
Emails	250	8/16	smtp.sendmail(), mailer.send()
Suppressions fichier	225	10/16	os.remove(), shutil.rmtree(), pathlib.unlink()
Invocations agent	206	8/16	graph.ainvoke(), chain.invoke(), agent.execute()
Paiements	41	3/16	stripe.PaymentIntent.create() + méthodes internes (~22% FP)

TypeScript (4 scopes)

Catégorie	Findings (approx.)	Scopes	Ce que ça détecte
destructive	~415	4/4	child_process.execSync(), spawn(), execFile(), execa()
file_delete	~210	3/4	fs.rm(), fs.unlink(), fsp.rm() avec flags destructifs
publish	~118	3/4	pubsub.publish(), topic.publish()
http_write	~80	4/4	fetch(POST/PUT/PATCH), axios.post()
agent_invocation	~50	4/4	runner.run(), agent.stream(), agent.execute()

Catégorie	Findings (approx.)	Scopes	Ce que ça détecte
llm_call	~10	3/4	openai.chat.completions.create(), anthropic.messages.create()
payment	0	0/4	Aucun usage Stripe/PayPal SDK trouvé dans les repos scannés

CAS ILLUSTRATIFS

Findings intéressants — du vrai code, dans de vraies codebases

Pas des exemples synthétiques. Six cas extraits directement des codebases scannées.

Khoj — ai_update_memories (Python)

Catégories : database_delete · agent_invocation

Un LLM décide quelles mémoires utilisateur garder et lesquelles supprimer. Appelle UserMemoryAdapters.delete_memory(user, memory) sans aucune confirmation humaine. L'agent réécrit de manière autonome ce qu'il « se souvient » de vous.

OpenAI Agents (Python) — pop_item

Catégories : http_write · database_delete

Appelle openai_client.conversations.items.delete() — supprime des items de conversation de l'API OpenAI sans confirmation. Un appel API externe qui ne peut pas être annulé.

Skyvern — workflow_delete (Python)

Catégories : destructive · database_delete

Appelle tool_workflow_delete(workflow_id=workflow_id, force=force) avec un paramètre force et aucune étape d'approbation. Le chemin force=True court-circuite tous les contrôles.

OpenAgents — 99 % sans garde (Python)

Catégories : database_delete · http_write

177 sur 178 tool calls n'ont aucune vérification. Principalement des redis_client.delete() et requests.post() sans rate limiting ni vérification d'auth.

OpenClaw — système d'approbation flaggé destructive (TypeScript)

Catégories : destructive — gouverné en externe

OpenClaw possède une infrastructure execApproval* sophistiquée. Le scanner identifie correctement les spawn() sous-jacents comme destructive, mais la gate d'approbation vit souvent dans une fonction différente. Le code est gouverné ; la gouvernance ne tient juste pas dans une fonction. Cas canonique pour l'annotation # checked:ok.

Mastra — processWorkflowSleep() (TypeScript)

Catégories : publish — pubsub interne

Quatre pubsub.publish('workflow.events.v2.${runId}', ...) chaînés dans une seule fonction. C'est le bus d'événements interne de Mastra — une primitive de framework non gardée by design.

CHECKLIST PRÉ-DÉPLOIEMENT

Quoi faire de vos résultats de scan

Chaque finding reçoit un verdict aligné sur le modèle diplomat-gate : STOP (bloquer en production), REVIEW (bloquer jusqu'à approbation humaine), ou CONTINUE (autoriser avec logging).

Catégorie	Verdict	Action recommandée
Paiement	**STOP**	Policy amount_limit + duplicate_detection + recipient_blocklist dans diplomat-gate.
Email / messaging	**STOP ou REVIEW**	Policy domain_blocklist + rate_limit + business_hours.
DB delete / drop	**REVIEW**	Queue TTL + approbation humaine requise via diplomat-gate.
Subprocess / exec	**STOP**	Allowlist d'entrée + STOP sur commandes inconnues.
HTTP write (externe)	**REVIEW**	Rate limit + auth check. Gater si endpoint externe et critique.
Appels LLM	**À évaluer**	Budget tokens + retry bound. Gater si invocation récursive d'agent.
Invocations agent	**À évaluer**	Limite de profondeur. Gater les appels orchestrateur → sous-agent.
DB write (commit)	**À évaluer**	Si Pydantic validation en couche service, acquitter. Sinon gater.
Suppressions fichier	**REVIEW**	Gate de confirmation si chemin fourni par l'utilisateur.
Pubsub interne	**ACQUITTER**	Annoter # checked:ok — internal pubsub, no process boundary crossed.

Le process en trois étapes

```
# 1. Scanner votre repo — 30 secondes pip install diplomat-agent && diplomat-agent scan . npm install -g @diplomat-ai/diplomat-agent-ts && diplomat-agent-ts scan src/ # 2. Triage # Revoir chaque finding. Corriger, gater, ou acquitter. # Utiliser # checked:ok — pour les findings protégés ailleurs. # 3. Gater pip install diplomat-gate # Ajouter le gate dans le chemin d'invocation des tools de votre agent. # Policies en < 1 ms. Zéro dépendance obligatoire.
```

MÉTHODOLOGIE

Comment les chiffres ont été produits

Scanner	Module ast de la stdlib Python (zéro dépendance) et ts-morph TypeScript.
Détection	40+ patterns matchant des appels de fonction avec effets de bord sur 10–12 catégories.
Gardes détectées	Validation d'entrée (Pydantic / Zod), rate limiting, auth, approval gates, clés d'idempotence, retry bounds — dans la même fonction que l'effet de bord.
Scope	Intra-procédural. Chaque fonction analysée indépendamment, avec résolution des décorateurs de la même classe.
Reproductibilité	Tous les SHAs de commit sont épinglés. Commandes de reproduction dans le repo.

Verdict	Condition
UNGUARDED / no_checks	Effets de bord + zéro garde détectée
PARTIALLY_GUARDED / partial_checks	Quelques gardes mais pas de couverture complète
GUARDED / confirmed	Annoté # checked:ok OU 2+ types de gardes distincts
LOW_RISK	Aucun effet de bord détecté (scanner Python uniquement)

GLOSSAIRE

Vocabulaire Diplomat

Tool call	Un appel de fonction effectué par un agent IA qui a des effets en dehors du contexte du LLM.
Effet de bord (side effect)	Toute opération qui modifie un état externe : DB write, HTTP write, subprocess, paiement, email, suppression de fichier, invocation d'agent.
Garde (guard)	Un contrôle intra-fonctionnel qui contraint un tool call : validation d'entrée, rate limit, auth, approval gate, clé d'idempotence, retry bound.
Verdict (statique)	UNGUARDED, PARTIALLY_GUARDED, GUARDED, LOW_RISK — classification par diplomat-agent.
Verdict (runtime)	CONTINUE, REVIEW, STOP — décision diplomat-gate en < 1 ms.
Receipt	Enregistrement immuable d'un verdict runtime, incluant le hash SHA-256 du tool call. EU AI Act Article 12.
Behavioral BOM (toolcalls.yaml)	Inventaire par repo de chaque tool call que l'agent peut effectuer. Le « requirements.txt pour ce que l'agent peut faire ».

ET APRÈS

Trois étapes pour gouverner vos agents en production

01

Know — diplomat-agent

Scannez votre repo. Obtenez l'inventaire complet des tool calls avec effets de bord et leur statut de protection. 30 secondes.

```
pip install diplomat-agent && diplomat-agent scan .
```

github.com/Diplomat-ai/diplomat-agent

02

Decide — diplomat-gate

Gater les tool calls critiques. Verdict CONTINUE / REVIEW / STOP en < 1 ms. 15+ policies pré-construites. Zéro dépendance obligatoire. Audit trail SHA-256.

```
pip install diplomat-gate
```

github.com/Diplomat-ai/diplomat-gate

03

Prove — diplomat.run

Control plane hébergé. Audit trail cross-tenant immutable, dashboard temps réel, approval routing managé, export conformité EU AI Act Article 12.

diplomat.run

Scannez votre repo maintenant

pip install diplomat-agent && diplomat-agent scan .

Prenez 30 minutes avec nous : calendly.com/josselin-guarnelli

Diplomat Services SASU · SIREN 100 251 834 · Cagnes-sur-Mer, France