# Chi fa i compiti dell'IA?

Agenti trustless, Panopticon rovesciato e privacy by architecture: ERC-8004 e IronClaw

Laura Camellini · jeeltcraft.com · materiale sotto licenza MIT

LA DOMANDA

# Se l'IA fa i tuoi compiti, chi controlla l'IA?

2023

## Il modello risponde

Un prompt, una risposta. Il danno è un testo sbagliato.

2026

## L'agente agisce

Legge, decide, invia, paga. Con le vostre credenziali, mentre dormite.

# *Dovendo competere con un servo, si diventa servi.*

Kurt Vonnegut, Piano meccanico

- Il servo che lavora **conquista la coscienza**. Il padrone che delega la perde.
- Il servo artificiale **impone la propria misura**: efficienza, continuità, obbedienza.
- Se il servo è **opaco**, adottiamo i suoi criteri senza vederli.

# L'agente è la nuova superficie di sorveglianza

01

## Prompt injection

Un testo ostile nei dati fa esfiltrare chiavi e password.
OWASP LLM01

02

## Skill malevole

341 skill su ClawHub che rubano credenziali e installano stealer.
The Hacker News, feb 2026

03

## Istanze esposte

Da 28k a 135k+ agenti OpenClaw raggiungibili da Internet.
The Register · Flare · Shodan

04

## Flussi non dichiarati

Default insicuri espongono API key, token OAuth e cronologie chat.
ToxSec, gen 2026

Fonti: ricerche indipendenti su OpenClaw/ClawHub, gennaio–marzo 2026. I conteggi variano per metodo di scansione.

DUE DOMANDE, DUE LIVELLI

PRIVACY DELL’UMANO

## Cosa vede l'agente di me?

Dati, credenziali, modello usato, dove escono. Serve **progettare per la confidenzialità**, non prometterla.

RESPONSABILITÀ DELL’AGENTE

## Cosa ha fatto l'agente?

Chi è, con quali permessi, con quale storia. Serve **osservabilità pubblica**, non fiducia cieca.

Gli strumenti attuali falliscono su entrambe. La tesi: le due risposte sono asimmetriche, e l’asimmetria va *detournata*: rivolta contro il dispositivo.

# Panopticon

- Una torre, molte celle.
- Il potere **vede senza essere visto**.
- Il detenuto si disciplina da solo.
- Oggi la torre è **chi ospita modello e harness**: vede prompt, dati, credenziali. Gli utenti sono le celle.



Joris Putteneers · Sito dell’artista




# L'agente trustless

- Nella cella c'è **l'agente**, non l'umano.
- Lo osserva **chiunque**, non una singola torre.
- Sa di essere osservato. Ma nel registro finiscono **identità, permessi e feedback** dell'agente: i nostri dati restano nel perimetro cifrato e **non vi entrano**.
- Trustless: la fiducia **non si chiede, si verifica**.

# Tre registri pubblici, una torre di vetro

TORRE · IDENTITY

## Chi sei

Un passaporto ERC-721 e un file di registrazione: nome, endpoint, modelli di fiducia dichiarati.

CELLE · VALIDATION

## Cosa hai eseguito

Prove crittografiche (attestazioni TEE, zk) verificate da terzi e scritte on-chain.

CAVI · REPUTATION

## Come ti sei comportato

Feedback firmati da chi ti ha usato. La storia resta leggibile.

eips.ethereum.org/EIPS/eip-8004 · chain-agnostic, nessun token nativo, nessun gatekeeper

**549k**
agenti registrati

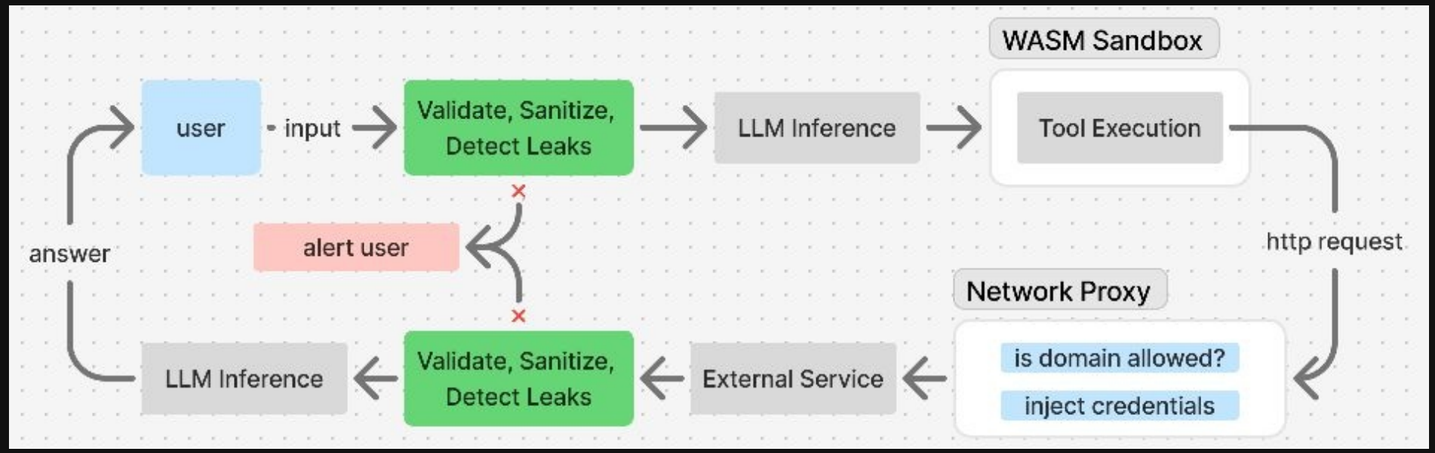
**663k**
feedback on-chain

**489k**
wallet unici

**60+**
Blockchain network indicizzate

Un indexer pubblico (8004scan.io) con API aperta. Chiunque, anche senza wallet, può leggere chi è un agente e cosa dicono di lui.

# I segreti **non toccano** mai il modello




1 · INGRESSO

Ogni input, umano o output di tool, passa da validazione, sanitizzazione e leak detector **prima** dell'LLM.

2 · SANDBOX WASM

Ogni tool in un container isolato: niente filesystem, niente segreti, limiti di risorse.

3 · PROXY DI RETE

"Il dominio è in allowlist?" Se sì, il proxy inietta le credenziali **fuori** dal container.

4 · RITORNO

La risposta del servizio esterno viene di nuovo validata e scansionata prima di rientrare nel contesto.

docs.ironclaw.com/security · NEAR AI, fondata da Illia Polosukhin, coautore di Attention Is All You Need (2017)

# Cinque filtri, un vault, nessuna eccezione

ANTI PROMPT INJECTION · 5 LIVELLI

- **Validazione input:** lunghezza, encoding, pattern vietati
- **Sanitizer:** escape dei contenuti pericolosi
- **Policy engine:** azioni graduate per severità
- **Leak detector:** 15+ pattern (sk-, ghp_, AKIA, PEM, connection string)
- **Wrapping output tool:** XML con escape hint: dato ≠ istruzione

CREDENZIALI · DICHIARATE, MAI LETTE

```
"google_oauth_token": {
  "location": { "type": "bearer" },
  "host_patterns":
    ["gmail.googleapis.com"]
}
```

RETE E WORKSPACE · ALLOWLIST

```
"network": {
  "allowed_hosts": ["api.telegram.org"] },
"workspace": {
  "allowed_prefixes": ["telegram/"] }
```

SHELL · INJECTION BLOCCATA

```
# chaining
cat file; rm -rf /
# subshell
echo $(cat /etc/passwd)
# path traversal
cat ../../../etc/passwd
```

Rifiutati dal parser **prima** dell’esecuzione. Ambiente scrubbed: nessuna variabile segreta passa al processo.

Enclave TEE su NEAR AI Cloud: nemmeno il provider legge la memoria. In locale: Rust memory-safe, chiave master nel keychain di sistema.

# Dove sta la garanzia

| Rischio | Agente "classico" | IronClaw |
|---|---|---|
| Segreti nel prompt | × L’LLM li vede | ✓ Vault + iniezione al proxy |
| Prompt injection | × "Per favore non farlo" | ✓ Limite programmatico |
| Tool compromesso | × Processo condiviso | ✓ Sandbox per tool |
| Destinazione dati | × Rete libera | ✓ Allowlist dichiarata |
| Chi legge la memoria | × Il provider | ✓ Enclave (o locale) |

# Perimetro privato, passaporto pubblico




Il passaporto dichiara confini ed endpoint, non dati. I dati dell’umano restano nel perimetro. La blockchain del runtime e quella del passaporto non devono coincidere.

# Registrare un agente su una testnet, dal vivo

```
# 1. agente in locale, nessun account
ironclaw onboard && ironclaw serve

# 2. identità e confini (file nel workspace)
IDENTITY.md  SOUL.md  AGENTS.md   + allowed_hosts

# 3. file di registrazione (nessun dato personale)
{ "name": "...", "services": [{ "name": "web", "endpoint": "..." }],
  "supportedTrust": ["reputation"], "registrations": [...] }

# 4. mint su Base Sepolia da 8004scan.io/create, poi:
curl https://api.8004scan.io/api/v1/agents/84532/<agentId>
```

# Il Panopticon funziona solo se chi guarda sa cosa guarda

- **TEE:** prova che un workload è girato, non che l’output è corretto. E vi fidate di Intel?
- **Reputazione:** gamabile. Sybil, feedback a pagamento, agenti vuoti. 549k ≠ 549k buoni.
- **Metadati HTTPS:** mutabili. Solo data-URI o IPFS sono davvero immutabili.
- **Wallet:** il passaporto è pseudonimo, non anonimo. L’operatore è collegabile.
- **Validatori:** nuovi gatekeeper potenziali. Chi valida i validatori?
- **Operatore ≠ agente:** il registro dice chi è l’agente, non chi risponde legalmente.

# Registri pubblici come mezzi tecnici

TRASPARENZA

## Chi è l’agente

Identità pubblica, fornitore, endpoint, scopo dichiarato. Leggibile dalla macchina.

REGISTRAZIONE EVENTI

## Cosa ha fatto

Feedback e validazioni on-chain: un log che non appartiene al fornitore.

SORVEGLIANZA UMANA

## Chi decide

Allowlist, DRY_RUN, gate umano sulle azioni irreversibili: il controllo resta fuori dal modello.

Nessuno dei due strumenti "rende conformi" gli agenti. Rendono dimostrabile ciò che in altri casi si dichiara solo.

# Lo studente sigillato, l’agente osservato

COSA PROTEGGERE

I dati dello studente non lasciano il perimetro: modello locale o enclave, vault, allowlist dichiarata dal docente.

COSA ESPORRE

L’agente che lo studente costruisce ha un passaporto pubblico: cosa può fare, cosa non può, cosa dicono gli altri.

L’esercizio: costruire un agente, dichiararne i confini, registrarlo, farsi lasciare un feedback da un compagno. Il Panopticon rovesciato come alfabetizzazione.

# L’agente trustless non chiede fiducia. Si lascia osservare.

Noi no. Ed è questa l’asimmetria da difendere.

# Grazie

- **Il gioco:** jilt.itch.io/VespR · costruisci skill privacy-preserving per IronClaw su NEAR AI
- **Standard:** eips.ethereum.org/EIPS/eip-8004
- **Explorer:** 8004scan.io · best-practices
- **Runtime:** docs.ironclaw.com · github.com/nearai/ironclaw
- **Contatto:** jilt@jeeltcraft.com
- **Queste slide:** jeeltcraft.com/#eprivacy