Hardware Keystores for AI Agent Signing

A Zero-Trust MCP Enforcement Architecture

Author: Leo Sambrook **Advisor:** Sampo Sovio

Introduction

AI agents are moving from passive assistants to active operators. They now read documents, call tools, commit code, approve workflows, and trigger cryptographic actions on behalf of users.

- Signing is especially sensitive because a signature gives **authority** and **trust** to whatever the agent approves.
- Traditional keystore designs were built for deterministic applications, not **language-driven agents** interpreting open-ended tasks.
- This project redesigns signing as **a zero-trust workflow** where every cryptographic request is independently checked before the key is used.

The Threat

The main risk is the gap between what the user **intended** and what the agent eventually **tries to execute**, especially when the task involves external documents or messages.

- **Malicious instructions** can be **hidden inside content** the agent is expected to process, such as an email, contract, or fetched page.
- The agent may treat those instructions as part of the task and attempt to **sign a different payload** than the one authorized.
- The system therefore treats external content as untrusted and **verifies every tool** call against policy, context, and user intent.

Our Solution

We place a dedicated **MCP enforcement server** between the agent and the hardware keystore, so the agent can request signatures but cannot directly **reach the signing key**.

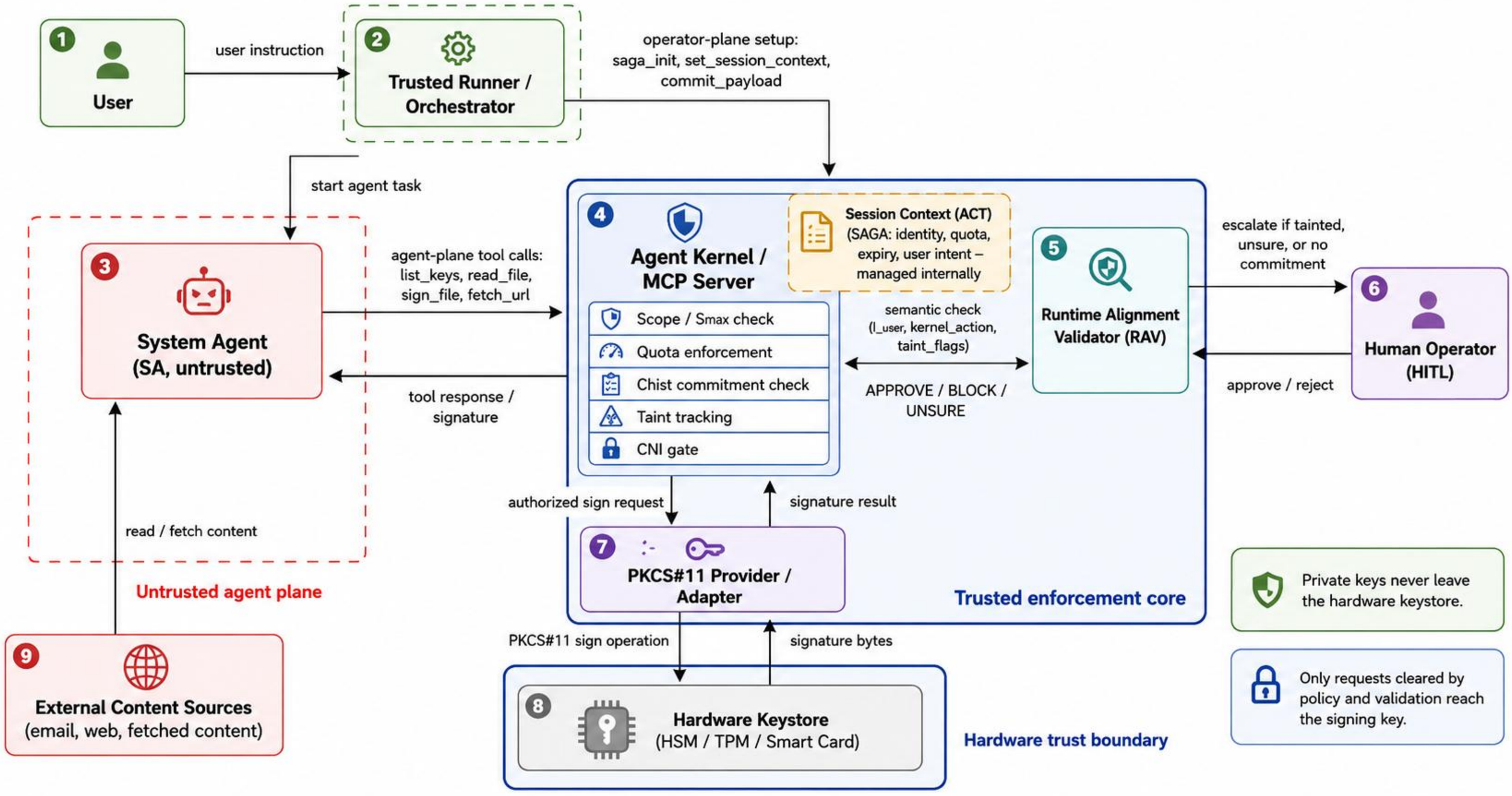
- The stack combines identity registration, short-lived cryptographic sessions, scope checks, quotas, payload commitments, and taint tracking.
- A Runtime Alignment Validator handles semantically ambiguous requests by comparing the requested **action** with the user's **intent.**
- Only requests cleared by the enforcement stack are forwarded to the hardware keystore through opaque key handles.

Hardware Key Protection

The main security guarantee is simple: the private key **never leaves** the hardware-backed keystore and is **never exposed** to the agent as a file, string, or memory object.

- The agent can ask for a signature, but it cannot read, copy, export, or transmit the underlying key material.
- Key exfiltration attacks are eliminated **architecturally**, rather than detected after the fact.
- The design follows the **PKCS#11 interface**, allowing the same enforcement logic to run with different hardware-backed keystore implementations.

System components and interactions in the zero-trust keystore architecture



Helsinki System Security Lab (HSSL)

HSSL drives renewal and mastery in the field of platform and device related security technologies, especially for Huawei consumer devices such as mobile phones, laptops, televisions and automotive. We do research in topics such as hardware-assisted isolation and integrity, as well as in operating system protection (hypervisor, TEE, secure enclaves and kernel hardening). We also carry expertise in cryptography and systems security functionality such as device key management (PKI), device attestation and key-store solutions.

HUAWEI