Security and Privacy Without Surrendering Interoperability

A cryptographically verifiable middle ground for AI-assistant access to consequential device functions

Problem. A platform should not have to choose between a closed first-party assistant and giving a third-party AI unrestricted authority over messages, files, payments, credentials, sensors, device settings, or other consequential functions. The execution-finality model separates **requesting an action** from **possessing authority to make that exact consequence externally effective**.

1. Commit the exact proposed consequence. For Candidate Act A, define a cryptographic commitment

$$C_A = H(\mathsf{Canon}(I \,\|\, R \,\|\, D \,\|\, X \,\|\, S \,\|\, E_p \,\|\, N))$$

where I is requester identity, R the exact release-form resource/content, D destination, X consequence class, S designated Finality Sink, E_p policy/governance epoch, N a fresh nonce, and H a collision-resistant cryptographic hash. The Candidate Act remains non-effective while protected validation runs.

2. Protected validation issues bounded authority, not a reusable permission.

$$V(A, P, \Sigma) \in \{0, 1\}, \qquad L = \mathsf{Sign}_{K_P}(C_A \,\|\, P_{id} \,\|\, E_p \,\|\, N \,\|\, S)$$
$$B_A = \mathsf{Auth}_{K_P}(C_A \,\|\, S \,\|\, N \,\|\, T \,\|\, Q)$$

V is protected validation against policy P and protected state Σ; L is protected validation evidence; B_A is scoped finality authority; T and Q bound validity and quota/cumulative authority. Possession of B_A alone is insufficient to create the consequence.

3. The Finality Sink re-derives what is actually about to leave the system.

$$C_A^* = H\big(\mathsf{Canon}(I^* \,\|\, R^* \,\|\, D^* \,\|\, X^* \,\|\, S^* \,\|\, E_p^* \,\|\, N^*)\big)$$

$$\boxed{E(A) = 1 \;\Rightarrow\; V(A, P, \Sigma) = 1 \;\wedge\; F(A, B_A, L, \Sigma) = 1 \;\wedge\; C_A^* = C_A}$$

Thus a recipient change, payload substitution, stale epoch, replayed nonce, altered sink, or changed resource causes the exact-release check to fail. In fail-closed form:

$$\boxed{\neg V \;\vee\; \neg F \;\vee\; (C_A^* \neq C_A) \;\Rightarrow\; \neg E}$$

4. Privacy: prove equality and policy application without exposing the underlying content.

A verifier need not receive the private message, file, prompt, protected key, proprietary fraud model, or complete platform internals. A deployment may use a keyed or ephemeral commitment, for example

$$C_R = \mathsf{HMAC}_k(R \,\|\, N), \qquad C_{\text{authorized}} = C_{\text{actual release}}$$

so that the verifier can establish that the authorized resource equals the released resource without publishing a stable reusable fingerprint of the content. Selective disclosure, short-lived commitments, attestations, or other privacy-preserving proofs may be used where appropriate.

Security middle ground: Interoperability + Protected Finality + Privacy-Preserving Evidence + Independent Verifiability.
A third-party AI may request an equivalent consequential action, while protected infrastructure independently decides whether *that exact consequence* is authorized to become real. The platform retains security control; interoperability does not require surrendering unrestricted execution authority.

Technical note: these equations express an architectural security invariant, not a claim that cryptography alone proves human intent, eliminates every hardware bypass, or guarantees exactly-once completion on remote systems. Assurance remains proportional to the protected paths, trusted inputs, cryptographic assumptions, and implementation actually deployed.