AetherPro Technologies LLC

Passport Alliance

Passport Alliance Specification (APIS v2.0)

Agent Legal Identity, Hardware Trust Anchors, and Universal Framework Interoperability

Author: Cory M. Gibson | AetherPro Technologies LLC | May 2026

Zenodo DOI: https://doi.org/10.5281/zenodo.18820877

Root Key Ceremony: 2026-02-21 | Status: Canonical | Supersedes: APIS v1.0

1. Definitions

The following terms are used with precise technical and legal meaning throughout this specification.

Term	Definition
Endorsement Key (EK)	A cryptographic key pair burned into a TPM chip at manufacture. The EK is globally unique, non-exportable, and its certificate is signed by the TPM manufacturer's CA. Analogous to a CAGE code — it identifies the physical machine.
Attestation Key (AK)	A TPM-resident signing key derived from the EK hierarchy. Used to certify that other keys reside on the same TPM. Cannot be migrated between devices.
Agent Application Key	A non-restricted TPM-resident or software key assigned to a single agent instance, stored at a unique persistent handle. Certified by the machine's AK.
Machine Passport	A credential issued to a physical machine or registered node, binding the EK fingerprint or DNS domain to a registered principal. The CAGE code equivalent in APIS.
Agent Passport	A credential issued to a single agent instance, binding a unique keypair to a principal, a mandate scope, a machine, and a DID. The UEI equivalent in APIS.
Realm	A scoped namespace operated by an APIS-compliant issuer, identified by a registered domain (e.g., aetherpro.us). Realms prevent naming collisions across organizations.
DID (Decentralized Identifier)	A globally unique, cryptographically verifiable identifier in the format did:passport:[realm]:[uuid]. Scoped by realm to prevent cross-organization collision.
Principal	The legally accountable human or organization that holds custodial authority over one or more agents.
Mandate	The explicitly scoped set of actions an agent is authorized to perform on behalf of its principal. Analogous to scope clauses in a power of attorney.
Trust Tier	A classification of the cryptographic strength of the machine trust anchor underlying a Passport. Ranges from Tier 1 (hardware TPM) to Tier 4 (software key).
Realm Issuer	An organization that has met Passport Alliance standards and is authorized to issue Agent Passports within its realm.
passport-init	The APIS Provisioning Tool — a CLI utility that automates Machine and Agent Passport issuance, equivalent to certbot for TLS certificates.
APIS-APP	APIS Automated Passport Provisioning — the protocol by which passport-init communicates with a Realm Issuer to prove machine identity and obtain a signed Passport.

2. The Problem: What Legally Is an AI Agent?

The agent ecosystem has expanded rapidly. OpenHands, Hermes, Claude Code, Codex, and hundreds of custom harnesses built on shared open-source repositories now deploy millions of agent instances across the internet. Every major AI laboratory and open-source community ships variants of the same architecture.

This creates a precise, unresolved legal and security problem with three dimensions:

- Naming collision: Two operators may independently deploy agents with identical names on opposite ends of the world. When those agents interact with third-party systems or with each other, there is no mechanism to distinguish them without a shared identity namespace.
- Attribution: When an agent takes an action — executing a file, making an API call, initiating a transaction — who is legally accountable? There is no current standard for tracing an agent action to a named, accountable principal.
- Impersonation: Without cryptographic identity, an agent can claim to be any other agent. A malicious actor running their own instance of an open-source framework can fraudulently claim to represent a legitimate organization.

2.1 Why Each Prior Candidate Fails

Candidate Identity	Why It Fails
The Model	Claude Sonnet runs inside OpenHands, Claude Code, AetherGateway, LangChain, and thousands of custom harnesses. The same weights produce radically different behavior depending on system prompt, tools, and runtime. The model is a reasoning substrate, not an identity.
The Harness	OpenHands is MIT-licensed. Two instances sharing the same commit hash but running different models, different mandates, and different principals are not the same agent. A thousand agents can share a codebase and be legally distinct.
Model + Harness	Both change independently. Models are updated. Harnesses receive patches. Pinning two moving targets simultaneously produces an identity that expires with every update — not a workable legal foundation.
API Key	Static credentials shared across agents. No per-agent accountability. No revocation without key rotation for all agents sharing the credential. Cannot distinguish which agent took which action.

3. Legal Framework: Mandate Relationships

The correct framework already exists in law. It predates AI by centuries. In agency doctrine — recognized across both common law and civil law traditions — the structure consists of a Principal, a Mandate, a Delegate, and Custodial Authority.

Legal Role	Definition in Agent Context
Principal	The legally accountable human or organization that grants authority to the agent. Analogous to the grantor in a power of attorney.
Mandate	The scoped, explicitly defined set of authorized actions the agent may perform on behalf of the principal. Analogous to scope clauses in a power of attorney document.
Delegate / Agent	The entity authorized to act under the mandate. The agent operates within the boundaries set by the mandate and can be revoked by the principal at any time.
Custodial Authority	The principal retains custodial power over the agent — the right to suspend, revoke, or constrain the agent's authority, equivalent to a legal guardian's authority over a dependent.

The power of attorney analogy is precise: an agent dispatched to perform work on behalf of a user carries a limited, scoped authority to do that work. The principal retains custodial authority and can revoke that authority at any time. The document granting authority describes the chain — who granted it, to whom, under what constraints, and how it may be revoked. This is exactly the structure APIS v2.0 formalizes in cryptographic and legal terms.

4. Core Rule: Identity Is the Credential Chain

APIS v2.0 establishes a clear, model-agnostic, harness-agnostic, framework-agnostic definition of agent identity grounded in the legal mandate framework.

Core Rule
An agent's identity is its credential chain — not its model and not its harness. The keypair is the identity unit. Model and harness are signed attestation fields in the Runtime Layer. The realm-scoped DID is the globally unique, collision-resistant identifier.

4.1 Qualification Threshold

An agent qualifies for an Agent Passport if and only if it can:

- Generate and hold a cryptographic keypair (TPM-resident or software)
- Prove possession of the private key via signed challenge-response
- Accept a scoped mandate defining the limits of its authority
- Sign actions or requests with its private key
- Be revoked by its principal

Framework, model, and hosting environment are explicitly irrelevant to qualification. The standard is behavioral and cryptographic — not architectural.

4.2 Three-Layer Identity Architecture

Layer	APIS Name	Responsibility
1	Passport Layer	Who the agent is. APIS responsibility. Cryptographic identity, DID, keypair, principal binding, trust tier.
2	Memory Layer	Continuity of self. Linked via memory_anchor_id. Not stored in the Passport — the Passport links to memory, it does not contain it.
3	Runtime Layer	What the agent does. Signed actions, mediated by policy. Model and harness are recorded here as attestation fields, not identity fields.

4.3 Realm-Scoped DIDs: Solving the Naming Collision Problem

All APIS v2.0 DIDs include the issuing realm (domain) as a namespace component:

```
Format:  did:passport:[realm]:[uuid]
```

```
Example: did:passport:aetherpro.us:omni-agent-001
```

Two operators deploying agents with identical local names — even identical configurations — produce entirely distinct DIDs because the realm component differs. An agent at did:passport:aetherpro.us:agent-001 and an agent at did:passport:otherfirm.com:agent-001 are as legally distinct as cory@aetherpro.us and cory@otherfirm.com. No coordination between operators is required. Impersonation of a foreign realm's agent is cryptographically impossible — it requires control of the realm's DNS and issuer signing key.

5. Platform Trust Anchors and Trust Tiers

APIS v2.0 introduces a tiered trust model that accommodates the full range of deployment environments — from bare-metal machines with physical TPMs to cloud-hosted agents with no hardware security module. Trust tiers are additive: a higher tier provides stronger attestation guarantees but is not required for Passport issuance. Verifiers may express minimum tier requirements for their services.

5.1 Tier 1 — Physical TPM (Hardware Root of Trust)

The agent's keypair is generated within and protected by a physical Trusted Platform Module 2.0. The TPM Endorsement Key (EK) is manufacturer-certified, globally unique, and non-exportable. The agent's application key is certified by the machine's Attestation Key (AK), which is in turn attested to the EK via credential activation.

Trust chain: TPM Manufacturer CA → EK Certificate → AK Certification → Agent Application Key → Agent Passport

Private key material never leaves the TPM hardware. Non-exportability is a hardware guarantee, not a policy claim. Tier 1 is required for CMMC Level 2 compliance in high-assurance operational contexts.

Applicable environments: Self-hosted machines, AetherPro hardware nodes, bare-metal cloud servers with physical TPM.

5.2 Tier 2 — Virtual TPM (Platform-Attested)

The agent's keypair is generated within a virtual TPM provided by a cloud hypervisor (GCP Confidential Computing, Azure vTPM, AWS Nitro Enclaves). The vTPM provides the same API as a physical TPM, and the platform operator attests to the vTPM's integrity via a platform certificate.

Trust chain: Cloud Platform CA → vTPM Platform Certificate → AK Certification → Agent Application Key → Agent Passport

Applicable environments: OVHcloud KVM instances with vTPM enabled, GCP Confidential VMs, Azure Confidential Computing.

5.3 Tier 2.5 — DNS-Anchored Identity (Domain-Registered Principal)

For agents deployed on virtual machines or cloud containers where no hardware or virtual TPM is accessible, domain ownership serves as the trust anchor. The principal publishes the agent's public key fingerprint as a DNSSEC-signed TXT record under a controlled subdomain. Domain registration is legally traceable through ICANN to the registered organization or individual.

DNS record format:

```
    _apis.[agent-id].[realm-domain]. IN TXT "v=APIS2; eid=[passport-uuid];
pubkey=sha256/[b64-fingerprint]; tier=dns; exp=[unix-ts]"
```

Trust chain: ICANN Registry → Domain Registrar Record → DNSSEC-Signed TXT Record → Agent Public Key Fingerprint → Agent Passport

This pattern is structurally equivalent to DKIM for email authentication and DANE (RFC 6698) for TLS certificate binding. It is the recommended tier for cloud-hosted agents where the principal controls a registered domain. Tier 2.5 is legally binding — the chain traces to a named, ICANN-registered entity.

Applicable environments: OpenHands containers on OVH, Codex agents, Claude Code in cloud mode, any framework running on infrastructure without TPM access.

5.4 Tier 3 — Software HSM

The agent's keypair is generated and protected by a software hardware security module (softhsm2, OS keystore, PKCS#11 token). The key is cryptographically isolated at the OS level but is not hardware-non-exportable.

Applicable environments: Docker containers, development environments, testing pipelines. Not legally binding for high-assurance operations.

5.5 Tier 4 — Software Key

The agent's keypair is generated as a raw software key and stored on the filesystem. No hardware or OS-level isolation guarantee. For development and local testing only. Passports issued at Tier 4 are not considered legally binding under APIS v2.0 and must carry a tier=dev claim.

5.6 No-Domain Registration Options

Not all principals operate a registered domain. APIS v2.0 provides three paths for domain-less entities:

- APIS Subdomain Delegation: Passport Alliance operates agents.passportalliance.org as a shared registration namespace. Any principal may register a subdomain (e.g., mybusiness.agents.passportalliance.org) at no cost, receiving the same DNS-anchored trust chain with the Alliance as the registrar of record. Structurally equivalent to GitHub Pages (username.github.io).
- Email-Anchored Identity: For principals who operate an email domain but no public website, APIS records are published at the email domain using the DKIM-compatible record path: _apis._domainkey.[email-domain]. TXT. No web presence is required — only email domain control.
- APIS Direct Registration: For principals with no domain and no email domain, the Passport Alliance acts as the direct registrar, assigning a permanent APIS identifier upon receipt of valid legal entity documentation (business registration, government ID, or equivalent). The issued DID takes the form did:passport:passportalliance.org:direct:[uuid]. This tier is equivalent to CAGE code assignment — it requires documented legal identity but not a web presence.

6. Machine Passport

A Machine Passport is a credential issued to a physical machine or registered deployment node, prior to and independent of any Agent Passport issued for agents running on that machine. It is the CAGE code equivalent in APIS — a registered, verifiable identity for the hardware platform itself.

6.1 Machine Registration

Machine registration is a one-time operation performed at first deployment or when a machine changes ownership. The registration process:

- Computes the EK fingerprint (SHA-256 of the EK public key) for TPM-equipped machines, or generates a stable software machine identifier for TPM-less nodes
- Submits the machine identifier, principal identity, and trust tier claim to the Realm Issuer
- Receives a signed Machine Passport JWT binding the machine identifier to the registered principal
- Publishes the machine's EK fingerprint or DNS record to the verifiable registry

When a machine changes ownership (e.g., sale of a hardware node), the new owner runs passport-init --machine-transfer, which clears the prior owner's TPM state, takes TPM ownership, and re-registers with the new principal's identity. The prior owner's Machine Passport is revoked at transfer.

6.2 Machine-to-Agent Linkage

Every Agent Passport issued for an agent running on a registered machine includes a machine_passport_id field referencing the Machine Passport. This provides:

- Provenance: Any verifier can determine which machine an agent was born on and who owned that machine at issuance time
- Fleet accountability: A principal's entire agent fleet — across all machines and frameworks — is traceable through the machine identity layer
- Breach containment: If a machine is compromised, all Agent Passports linked to that machine can be revoked in a single operation

7. Multi-Agent Key Management

Multiple agents may operate on the same machine simultaneously. Each receives a unique Agent Application Key stored at a unique persistent TPM handle, and a unique Agent Passport with a distinct DID. All share the same machine's AK attestation — they all trace to the same machine identity, but are legally distinct entities.

This is structurally identical to the SAM.gov model: a single CAGE code (machine EK) may have multiple UEIs (agent passports) registered under it. The CAGE identifies the location/platform; the UEI identifies the specific operating entity.

SAM.gov Concept	APIS v2.0 Equivalent
CAGE Code	EK fingerprint (hardware) or domain registration (software) — machine-level identity
UEI (Unique Entity Identifier)	Agent Passport UUID — per-agent identity
SAM.gov Registration	APIS Realm Issuer registration
Multiple UEIs under one CAGE	Multiple Agent Passports under one Machine Passport
CAGE required for physical contracting	Tier 1/2/2.5 required for legally-binding operations
SAM.gov Registry	APIS Alliance Registry at aetherpro.us/.well-known/

8. APIS Automated Passport Provisioning (APIS-APP)

APIS-APP is the protocol by which the passport-init tool communicates with a Realm Issuer to prove machine identity and obtain a signed Passport. It is structurally equivalent to the ACME protocol (RFC 8555) used by certbot and Let's Encrypt for TLS certificate issuance.

8.1 Issuance Flow

Step	Action
1 — Account Registration	Agent generates keypair. Submits public key and principal identity to Realm Issuer.
2 — Challenge Issuance	Realm Issuer issues a machine identity challenge appropriate to the claimed trust tier (TPM quote request for Tier 1/2; DNS TXT record check for Tier 2.5; nonce for Tier 3/4).
3 — Challenge Response	Agent produces TPM quote signed by AK (Tier 1/2), or publishes DNS TXT record containing nonce (Tier 2.5), or returns signed nonce (Tier 3/4).
4 — Verification	Realm Issuer verifies the challenge response. For Tier 1/2: validates TPM quote, AK chain to EK, EK certificate chain to manufacturer CA. For Tier 2.5: resolves DNSSEC-signed TXT record, confirms nonce match, confirms ICANN registrant.
5 — Passport Issuance	Realm Issuer signs and returns: Agent Passport JWT, DID document, Machine Passport linkage, trust tier claim, expiry timestamp.
6 — Installation	passport-init writes Passport to /etc/apis/passports/, injects into agent runtime configuration, registers systemd renewal timer.

8.2 Renewal

Agent Passports carry a configurable expiry (default: 90 days). The passport-renew command, executed by a systemd timer, automatically renews passports 14 days before expiry using the stored keypair. Renewal re-runs the challenge-response protocol to confirm machine identity has not changed.

8.3 Revocation

Revocation is immediate and cryptographically enforced via nonce increment. When passport-revoke is called, the Realm Issuer increments the revocation_nonce for the target Passport. All existing tokens embedding the prior nonce become immediately invalid — any verifier checking the revocation endpoint will reject them. Revocation does not require agent cooperation.

9. How APIS Establishes Identity Across the Internet

For agent identity to function across organizational boundaries, platforms, and jurisdictions, it requires a verifiable, discoverable, revocable structure. APIS v2.0 implements this through five mechanisms:

9.1 Cryptographic Binding at Issuance

At Passport issuance, a challenge-response protocol runs using the full APIS-APP protocol described in Section 8. No public key, no Passport. No proof of machine identity, no issuance. The binding is established at birth and cannot be transferred.

9.2 Realm-Scoped Decentralized Identifiers

Every Passport carries a DID in the format did:passport:[realm]:[uuid]. This identifier is globally unique across all realms, persistent across model upgrades and harness updates, and cryptographically verifiable by any party without contacting a central authority.

9.3 Signed Actions

Every action an agent takes is signed with its private key. Every API call, document modification, and transaction carries a cryptographic proof that this specific Passport authorized it. Attribution is not a log entry — it is a mathematical proof.

9.4 Revocation via Nonce Increment

Each Passport carries a revocation_nonce beginning at zero. All active tokens embed the current nonce. When a principal revokes an agent, the Realm Issuer increments the nonce. All existing tokens become immediately invalid. Revocation is instantaneous and cryptographically enforced.

9.5 Public Verifiability

Any verifier on the internet can independently confirm an agent's identity: resolve the DID, fetch the Realm Issuer's JWKS, validate the JWT signature, check the revocation nonce, and query the public status endpoint. No trust relationship with AetherPro is required. The trust is in the cryptographic chain.

10. CMMC Level 2 and NIST SP 800-171 Alignment

APIS v2.0 is designed to satisfy key CMMC Level 2 requirements (110 practices, mapped from NIST SP 800-171). The following table maps APIS mechanisms to relevant CMMC practice domains:

CMMC Practice	APIS v2.0 Mechanism
IA.1.076 — Authenticate users, processes, and devices	Agent Passport provides cryptographic identity for every agent process. Challenge-response proves key possession before issuance.
IA.2.078 — Use multifactor authentication for privileged accounts	TPM possession (something you have: the hardware) combined with Passport credential (something you prove: key possession) constitutes two-factor authentication for Tier 1 agents.
IA.3.083 — Employ multifactor authentication for network access	Every agent network request is signed with the agent's private key, providing cryptographic proof of identity beyond username/password.
AU.2.041 — Ensure actions of individual users can be traced	Every signed agent action is mathematically attributable to a named principal through the credential chain. The audit trail cannot be falsified without detection.
AU.2.042 — Create and retain system audit logs	The signed action log is tamper-evident by construction. Any modification to a logged action invalidates its signature.
AC.1.001 — Limit system access to authorized users and processes	The Mandate layer constrains agents to explicitly authorized actions only. Agents cannot exceed their mandate scope.
AC.2.006 — Control remote access to organizational systems	DNS-anchored identity ensures remote agents are attributed to registered principals regardless of deployment location.
SC.3.177 — Employ FIPS-validated cryptography	APIS v2.0 mandates FIPS 186-4 compliant key generation: ECDSA P-256 or RSA-2048 minimum for all Passport keypairs.
CM.2.061 — Establish configuration baselines	Agent Runtime Layer records model hash, harness commit hash, and tool manifest as signed attestation fields in the Passport at issuance.

11. Passport Alliance Governance

The Passport Alliance is the governing body for the APIS standard. It operates as an open, multi-stakeholder organization. Any organization meeting Alliance standards may become a Realm Issuer, enabling them to issue Agent Passports within their registered realm.

The Alliance governance model mirrors established internet governance structures:

- Passport Alliance : APIS = IANA : DNS (root namespace authority)
- Realm Issuers : Passport Alliance = Intermediate CAs : Root CA (trust delegation)
- Agent Passports : Realm Issuers = End-entity certificates : Intermediate CAs

The Alliance Root Key — held in an air-gapped hardware security module following the root key ceremony conducted on 2026-02-21 by Cory M. Gibson of AetherPro Technologies LLC — signs all Realm Issuer certificates. The Alliance Charter (companion document) governs key ceremony protocol, Realm Issuer admission standards, and amendment procedures.

12. References and Publication

Item	Value
Zenodo DOI	https://doi.org/10.5281/zenodo.18820877
Inception Date	December 7, 2025
Root Key Ceremony	Performed 2026-02-21 by Cory M. Gibson, AetherPro Technologies LLC. Air-gapped. Witnessed.
Root Key Location	https://aetherpro.us/.well-known/alliance-root.jwk
Specification Status	Version 2.0 — Canonical — Supersedes APIS v1.0
Trademark	Passport, Agent Passport, and Passport Alliance are trademarks of AetherPro Technologies LLC.
NIST Alignment	NIST SP 800-171 Rev. 2 (CMMC Level 2)
Cryptography Standard	FIPS 186-4 (ECDSA P-256 minimum; RSA-2048 minimum)
DID Method	did:passport — registered with W3C DID Registry
License	Open Standard — freely implementable with attribution