AetherPro Technologies LLC

Passport Alliance

APIS v2.0 Implementation Guide

passport-init, DNS Integration, TPM Setup, and Framework-Specific Integration Paths

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

1. Prerequisites and Environment

Before provisioning any agent passport, confirm the following environment requirements:

Requirement	Details
Operating System	Ubuntu 22.04+ / Debian 12+ recommended. Windows WSL2 supported. macOS supported via Secure Enclave path.
TPM Access (Tier 1/2)	Run: ls /dev/tpm* — should show /dev/tpm0 or /dev/tpmrm0. If absent, Tier 2.5 (DNS) or Tier 3 (softhsm2) applies.
tpm2-tools	sudo apt install tpm2-tools tpm2-tss — required for Tier 1/2 provisioning.
DNS Control	For Tier 2.5: you must have API access to your DNS provider (Cloudflare, Route53, etc.) to publish TXT records.
DNSSEC	For Tier 2.5: confirm DNSSEC is enabled on your domain: dig +dnssec yourdomain.com — look for RRSIG records.
Node.js 18+	Required for passport-init CLI: node --version
Realm Issuer Access	Your APIS Realm Issuer endpoint (e.g., https://apis.aetherpro.us) must be reachable from the provisioning machine.

2. Checking Your TPM on Linux

If you have an Asus machine (or any modern PC), you almost certainly have a TPM 2.0 chip. Here is how to verify and inspect it:

2.1 Verify TPM Presence

```
# Check if TPM device exists
ls /dev/tpm*
# Should return: /dev/tpm0  and/or  /dev/tpmrm0
```

```
# Check TPM version and manufacturer info
sudo tpm2_getcap properties-fixed
# Look for: TPM2_PT_MANUFACTURER, TPM2_PT_FIRMWARE_VERSION_1
```

2.2 Inspect Your EK (Your Machine CAGE Code)

```
# Generate and display your Endorsement Key
sudo tpm2_createek -c /tmp/ek.ctx -G rsa -u /tmp/ek.pub

# Get your EK fingerprint (this is your machine's unique hardware ID)
openssl rsa -pubin -in /tmp/ek.pub -outform DER 2>/dev/null | sha256sum
# Save this fingerprint -- it is your machine's CAGE code equivalent
```

2.3 Create Your Platform Attestation Key

```
# Create the Storage Root Key (parent for all application keys)
sudo tpm2_createprimary -C o -g sha256 -G rsa -c /tmp/srk.ctx

# Create the Attestation Key (machine signing key)
sudo tpm2_createak -C /tmp/srk.ctx -c /tmp/ak.ctx -G rsa -g sha256 -s rsassa -u
/tmp/ak.pub -n /tmp/ak.name

# Persist the AK so it survives reboots
sudo tpm2_evictcontrol -c /tmp/ak.ctx 0x81010001
# The AK is now permanently stored at handle 0x81010001
```

2.4 Create an Agent Application Key (One Per Agent)

```
# Create a non-restricted application key for agent omni-agent-001
sudo tpm2_create -C 0x81010001 -G ecc:ecdsa -g sha256 -u /tmp/agent001.pub -r
/tmp/agent001.priv
```

```
# Load and persist at unique handle
sudo tpm2_load -C 0x81010001 -u /tmp/agent001.pub -r /tmp/agent001.priv -c
/tmp/agent001.ctx
sudo tpm2_evictcontrol -c /tmp/agent001.ctx 0x81020001
# Handle 0x81020001 = agent-001, 0x81020002 = agent-002, etc.
```

3. passport-init: The APIS Provisioning Tool

passport-init is the APIS equivalent of certbot. It automates machine registration, challenge-response proof of identity, passport issuance, installation, and renewal scheduling.

3.1 Installation

```
npm install -g @passport-alliance/passport-init
# Or via curl installer:
curl -sSL https://apis.aetherpro.us/install.sh | bash
```

3.2 Machine Registration (One-Time Per Machine)

```
# Register this machine with your realm issuer
# For Tier 1 (physical TPM):
sudo passport-init register-machine \
  --realm aetherpro.us \
  --principal "AetherPro Technologies LLC" \
  --tier tpm \
  --ak-handle 0x81010001

# For Tier 2.5 (DNS-anchored, no TPM):
passport-init register-machine \
  --realm aetherpro.us \
  --principal "AetherPro Technologies LLC" \
  --tier dns \
  --dns-provider cloudflare \
  --dns-api-token $CF_API_TOKEN
```

3.3 Agent Provisioning (Run Once Per Agent)

```
# Provision a passport for OmniAgent (Tier 1, TPM-backed)
sudo passport-init provision \
  --realm aetherpro.us \
  --agent-name omni-agent-001 \
  --framework custom \
  --model mistral-small-4 \
  --harness aether-gateway \
  --tier tpm \
  --ak-handle 0x81010001 \
  --app-key-handle 0x81020001

# Provision for an OpenHands agent (Tier 2.5, DNS)
```

```
passport-init provision \
  --realm aetherpro.us \
  --agent-name openhands-dev-001 \
  --framework openhands \
  --tier dns

# Output: /etc/apis/passports/omni-agent-001.passport.jwt
```

3.4 Renewal and Revocation

```
# Check status of all passports
passport-init status --all

# Manual renewal (auto-renewal is handled by systemd timer)
passport-init renew --agent omni-agent-001

# Revoke a passport (e.g., agent decommissioned)
passport-init revoke --agent omni-agent-001 --reason decommissioned
```

4. DNS Record Setup for Tier 2.5 Agents

For agents on VMs or cloud containers without TPM access, DNS is the trust anchor. passport-init handles DNS record publication automatically when --dns-provider is specified. The following documents the record format for manual setup or verification.

4.1 DNS TXT Record Format

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


# Example for OmniAgent on aetherpro.us:
_apis.omni-agent-001.aetherpro.us. IN TXT "v=APIS2; eid=abc-001;
pubkey=sha256/AbCdEf...; tier=dns; exp=1780000000"
```

4.2 Enabling DNSSEC on Cloudflare

1. Log into Cloudflare dashboard for your domain
2. Navigate to DNS > Settings > DNSSEC
3. Click Enable DNSSEC — Cloudflare handles key generation and signing automatically
4. Copy the DS record values and add them to your domain registrar
5. Verify: dig +dnssec _apis.omni-agent-001.aetherpro.us TXT — look for RRSIG in the response

4.3 For Agents With No Domain: APIS Subdomain

```
# Request a free APIS Alliance subdomain
passport-init register-subdomain \
  --subdomain mybusiness \
  --email contact@mybusiness.com \
  --legal-name "My Business LLC"
# Assigns: mybusiness.agents.passportalliance.org
# Then provision agents using --realm mybusiness.agents.passportalliance.org
```

5. Framework Integration Guides

5.1 OpenHands / OpenClaw

OpenHands runs agents in Docker containers. The recommended pattern is host-managed passports mounted into containers as read-only volumes.

```
# On the host: provision the passport
passport-init provision --realm aetherpro.us --agent-name oh-dev-001 --framework
openhands --tier dns


# Mount into container (docker-compose.yml)
volumes:
  - /etc/apis/passports/oh-dev-001.passport.jwt:/run/secrets/agent.passport:ro


# In your OpenHands config (config.toml):
[agent]
passport_path = "/run/secrets/agent.passport"
sign_actions = true
```

5.2 Claude Code

Claude Code runs as a CLI process on the operator's machine. If the machine has a TPM, the agent gets Tier 1 attestation — the highest available.

```
# Provision once per machine
sudo passport-init provision \
  --realm aetherpro.us \
  --agent-name claude-code-01 \
  --framework claude-code \
  --tier tpm


# passport-init automatically configures Claude Code's CLAUDE.md
# to include the passport path for signing
```

5.3 Codex (OpenAI Cloud)

Codex agents run on OpenAI's infrastructure. No hardware TPM is accessible. Tier 2.5 (DNS) is the highest attainable tier. The passport is provisioned from the principal's control plane and injected into Codex via environment variable.

```
# Provision from your control plane (not from OpenAI's infrastructure)
passport-init provision \
  --realm aetherpro.us \
  --agent-name codex-agent-001 \
  --framework codex \
```

```
  --tier dns

# Inject via OpenAI Codex environment config
APIS_PASSPORT=$(cat /etc/apis/passports/codex-agent-001.passport.jwt)
```

5.4 Custom Python Agents (OmniAgent, Echo-1, PolyMorph)

For custom agents using your AetherGateway / LiteLLM stack:

```
# Install the APIS Python client
pip install passport-alliance-client

# In your agent bootstrap code:
from passport_alliance import PassportClient

passport = PassportClient.load('/etc/apis/passports/omni-agent-001.passport.jwt')

# Sign any action before dispatch
signed_action = passport.sign({
    'action': 'file_write',
    'target': '/path/to/file',
    'timestamp': time.time()
})
```

6. Trust Tier Verification

Any party can independently verify an agent's trust tier and passport validity. No AetherPro account required.

```
# Verify a passport JWT
passport-init verify --passport /etc/apis/passports/omni-agent-001.passport.jwt

# Expected output:
DID:       did:passport:aetherpro.us:omni-agent-001
Principal: AetherPro Technologies LLC
Tier:      1 (Hardware TPM)
Machine:   machine_passport:ek-sha256-AbCdEf...
Issued:    2026-05-03T00:00:00Z
Expires:   2026-08-01T00:00:00Z
Revoked:   false
CMMC:      Tier 1 satisfies IA.2.078, IA.3.083
```

7. TPM Ownership Transfer (Hardware Node Sales)

When selling or transferring a node with passport-init installed, the following procedure transfers TPM ownership to the new principal. The prior owner's Machine Passport is automatically revoked.

```
# Run on the machine being transferred
# Step 1: Revoke all agent passports on this machine
sudo passport-init revoke --all --reason ownership-transfer

# Step 2: Revoke the Machine Passport
sudo passport-init revoke-machine --reason ownership-transfer

# Step 3: Clear TPM owner hierarchy (buyer runs this at first boot)
sudo tpm2_clear -c platform

# Step 4: New owner registers the machine
sudo passport-init register-machine \
  --realm new-owner-domain.com \
  --principal "New Owner LLC" \
  --tier tpm
```

8. Renewal Schedule and systemd Timer

passport-init installs a systemd timer at provisioning time that auto-renews passports 14 days before expiry.

```
# View renewal timer status
systemctl status passport-renewal.timer

# View timer schedule
systemctl list-timers passport-renewal.timer

# Manually trigger renewal check
sudo systemctl start passport-renewal.service

# View renewal logs
journalctl -u passport-renewal.service -f
```

Passport renewal follows the full APIS-APP challenge-response protocol. If the machine's trust anchor has changed (e.g., TPM replaced, domain transferred), renewal will fail and an alert is generated. Renewal failures do not immediately invalidate the existing passport — the prior passport remains valid until its expiry, allowing time to investigate and re-provision.