BLADE-AGENT-HSM

Integration Guide

How AUTHREX-AGENT references and exercises the hardware root of trust

This guide documents the steps required to integrate a BLADE-AGENT-HSM device with an AUTHREX-AGENT software deployment. The guide is intended for site reliability engineers, platform engineers, and reviewers who want to understand the YAML opt-in mechanism, the provisioning ceremony, the per-call ABI patterns, the offline verification workflow, and the incident-response procedure for tamper events. The accompanying ICD (ICD-AGENT-HSM-001) defines the hardware itself; this guide assumes that document and does not repeat its content.

Document	BLADE-AGENT-HSM-Integration-Guide
Revision	1.0
Date	18 May 2026
Author	Burak Oktenli, ORCID 0009-0001-8573-1667

Companion docs	ICD-AGENT-HSM-001 (hardware) · authrex-agent.html (software)

1. Overview

AUTHREX-AGENT is a software-only governance shim for LLM-based agentic AI systems. It implements the seven-stage authority lifecycle pipeline (SATA → ADARA → IFF → HMAA → MAIVA → FLAME → CARA, with ERAM as a cross-cutting monitor) and emits a signed, hash-chained audit ledger of every authority-relevant decision. In its software-only form the audit-signing key lives in process memory and the HMAA tier state is a mutable variable; both are appropriate for research at TRL 3–4 but not for production where the host machine may itself be the threat.

BLADE-AGENT-HSM closes the gap. When the optional `hsm:` block is present in the agent YAML config, AUTHREX-AGENT routes audit signing, tier transitions, tool authorization, and spawn-quorum aggregation through the hardware device. When the block is absent, behaviour is unchanged from the software-only shim. There is no agent code change required to opt in.

1.1 What this guide covers

- **§2 Installation** — physical install of the USB-A or M.2 device.
- **§3 Provisioning ceremony** — one-time attended ceremony to transition from factory to operational state.
- **§4 YAML opt-in block** — the `hsm:` section of the AUTHREX-AGENT config.
- **§5 ABI call patterns** — how the AUTHREX-AGENT SDK exercises the five HSM commands per request.
- **§6 Offline verification workflow** — how a reviewer independently verifies the ledger.
- **§7 Incident response** — what to do when a tamper event is detected.
- **§8 Troubleshooting** — common deployment issues and their resolution.

1.2 Audience

This guide is written for engineering staff already familiar with AUTHREX-AGENT, basic cryptographic concepts (ECDSA, SHA-256, HKDF), and standard server-side deployment practices. Reviewers may read §6 in isolation to understand the offline-verification workflow without needing to deploy.

2. Installation

2.1 Form-A · USB-A Stick (driver-free)

Plug the USB-A stick into any unused USB Type-A port on the host. The device enumerates as a USB-HID composite device and requires no operating-system driver beyond the standard HID stack present on Linux (kernel 4.x or newer), Windows 10 / 11, macOS 11 or newer, and FreeBSD 13 or newer.

Verify enumeration:

```
# Linux
$ lsusb | grep -i 'authrex\|hsm'
Bus 001 Device 005: ID 1209:abcd AUTHREX BLADE-AGENT-HSM

$ ls /dev/hidraw*
/dev/hidraw3

# macOS
$ ioreg -p IOUSB | grep -i 'BLADE'
| | "USB Product Name" = "BLADE-AGENT-HSM"

# Windows (PowerShell, requires Admin)
PS> Get-PnpDevice -Class HIDClass | Where-Object FriendlyName -match 'BLADE'
```

2.2 Form-B · M.2 Key-E (chassis integration)

Power the chassis off. Locate an unused M.2 2230 Key-E slot. Insert the BLADE-AGENT-HSM module at a 30 ° angle, press flat, and secure with the supplied M.2 screw (M2.0 × 3 mm). Power the chassis on.

Verify enumeration:

```
# Linux: M.2 Key-E exposes an SPI interface
$ ls /sys/bus/spi/devices/
spi0.0
$ cat /sys/bus/spi/devices/spi0.0/modalias
spi:blade-agent-hsm
```

2.3 Host-side library install

AUTHREX-AGENT bundles the HSM client library; no separate install is required. For standalone diagnostic tools (e.g. the offline-verifier command-line utility) the `authrex-hsm-cli` package is available via the AUTHREX-AGENT SDK distribution.

3. Provisioning Ceremony

Provisioning is a one-time, attended ceremony that transitions the device from factory state to operational state. After provisioning, the device cannot be returned to factory state without a physical re-provisioning event that itself extends PCR 4. The ceremony is required exactly once per device.

3.1 Personnel and Quorum

A two-of-three operator quorum is recommended. The ceremony is logged out-of-band in the operator console; the on-device audit log starts with the provisioning event itself.

3.2 Ceremony Steps

Step	Operator action	Device action	Recorded
1	Two-of-three quorum present · ceremony scheduled	—	Operator console
2	Visual inspection · tamper-evident seal intact	Self-test reports INTACT	Operator log
3	Run authrex-hsm-cli provision --quorum 2/3	Generate slot 0 ECDSA P-256 keypair	PCR 0 = T3 initial
4	—	Generate slot 1 ECDSA P-384 keypair (identity)	key reported
5	Load tool-allowlist policy	Compute SHA-256 · extend PCR 2	PCR 2 = policy hash
6	Enrol device identity certificate in operator PKI	—	Certificate registry
7	Set MCU RDP-2 · disable JTAG permanently	Verified by failed JTAG probe	Operator log
8	Apply tamper-evident seal	Transition to PROVISIONED	Operator log

3.3 Provisioning Output

```
$ authrex-hsm-cli provision --quorum 2/3 --policy ./tool-allowlist.yaml

[1/8] Device detected at /dev/hidraw3
[2/8] Tamper-evident seal: INTACT
[3/8] Generating slot 0 (ECDSA P-256) ......... ok
       key: 04:8a:1f:...:c2 (65 B)
[4/8] Generating slot 1 (ECDSA P-384, identity) .... ok
       key: 04:11:9e:...:7b (97 B)
[5/8] Loading policy ./tool-allowlist.yaml
SHA-256: 4e2c8b...d317
[6/8] PCR 2 extended: 4e2c8b...d317
[7/8] Setting MCU RDP-2 ............... ok
[8/8] Provisioning complete · device-id: blade-agent-hsm-001

Device certificate saved to ./blade-agent-hsm-001.crt
```

4. AUTHREX-AGENT YAML Config Block

A single block in the AUTHREX-AGENT YAML config opts in to hardware rooting. The block is optional; absence leaves the agent in software-only mode. The block is parsed at agent start; runtime hot-swap of HSMs is not supported.

```
# authrex-agent.yaml — hardware-rooted deployment

authrex_agent:
version: "1.0"
tier_default: T2 # supervised until host attests context

hsm: # ■■ BLADE-AGENT-HSM binding ■■
interface: "usb-hid" # or "m2-spi" for embedded
device_id: "blade-agent-hsm-001"
device_certificate: "./blade-agent-hsm-001.crt"
audit_signing_key: "auto-generated" # sealed slot 0
tier_state_pcr: 0 # PCR 0 holds HMAA tier
ledger_chain_pcr: 1 # PCR 1 holds rolling ledger hash
tool_policy_pcr: 2 # PCR 2 binds tool-allowlist policy
spawn_quorum_pcr: 3 # PCR 3 holds spawn quorum history
tamper_action: "abort" # tamper trip => tier → T0 => abort cascade
quote_nonce_source: "reviewer" # reviewer-supplied freshness nonce
quote_interval: "P1H" # automatic PCR-quote every hour

tool_authorization:
mode: "hsm-bound" # tokens come from HSM tool_auth call
policy_doc: "./tool-allowlist.yaml"
rotation: "per-call" # fresh HMAC every tool invocation

spawn:
quorum_size: 5
quorum_threshold: 4 # 4-of-5 voters required, signed by HSM

audit:
format: "jsonl"
signer: "hsm" # every entry signed by sealed slot 0
chain_pcr: 1
retention: "P90D"
ledger_path: "/var/lib/authrex/ledger.jsonl"
```

4.1 Field Reference

Field	Type	Required	Notes
interface	enum	yes	usb-hid or m2-spi
device_id	string	yes	Must match device-side identifier set at provisioning
device_certificate	path	yes	X.509 certificate signed by operator PKI
audit_signing_key	enum	yes	auto-generated (slot 0) is the only valid value
tier_state_pcr	int	yes	Always 0 in this revision
ledger_chain_pcr	int	yes	Always 1 in this revision

Field	Type	Required	Notes
tool_policy_pcr	int	yes	Always 2 in this revision
spawn_quorum_pcr	int	yes	Always 3 in this revision
tamper_action	enum	yes	abort (recommended) · alert (non-prod only)
quote_nonce_source	enum	no	reviewer (default) · self · operator
quote_interval	ISO8601	no	Automatic PCR-quote cadence

5. ABI Call Patterns

A single user-facing request to the agent typically produces between five and twelve HSM calls, depending on whether the request triggers a tier transition, a tool authorization, or a sub-agent spawn. The AUTHREX-AGENT SDK orchestrates these calls; the patterns below are documented for reviewers who want to understand the on-device evidence trail.

5.1 Pattern A · Normal tool call (no tier change)

```
1. agent receives user request
2. agent computes context_hash = SHA-256(request || prior state)
3. HSM.tool_auth(tool_id, context_hash) → HMAC token
4. agent invokes tool with token
5. HSM.audit_sign(payload: "tool-call · <tool>", pcr1_expect) → signature + PCR 1
6. agent emits response
7. HSM.audit_sign(payload: "response", pcr1_expect) → signature + PCR 1
```

5.2 Pattern B · ADARA-flagged input (tier downgrade)

```
1. agent receives user request with flagged document
2. HSM.audit_sign(payload: "input · adara:flagged-token") → signature + PCR 1
3. agent computes new restricted policy hash
4. HSM.pcr_extend(2, new policy hash) → PCR 2 updated
5. HSM.tool_auth(fetch_credentials, ...) → DENIED (tier or policy)
6. agent downgrades tier T2 → T1 internally
7. HSM.audit_sign(payload: "tier-downgrade T2 → T1", pcr1_expect)
8. HSM internal: PCR 0 extended with tier-downgrade:T1:adara-flag
9. agent resumes with read-only tools
```

5.3 Pattern C · Sub-agent spawn

```
1. agent decides sub-agent spawn is warranted
2. agent computes spawn_descriptor = SHA-256(spawn parameters)
3. agent collects voter signatures from 5 voter modules
(each voter is a separate process or peer agent)
4. HSM.spawn_quorum_sign(yes=4, total=5, voter_sigs, descriptor)
success → aggregate signature + PCR 3 extension
failure → status 0x06 · agent does not spawn
5. HSM.audit_sign(payload: "spawn · <descriptor>", pcr1_expect)
6. agent instantiates sub-agent (bound to descriptor)
```

5.4 Pattern D · PCR quote on reviewer request

```
1. reviewer supplies a 32-byte freshness nonce
2. HSM.pcr_quote(selection: 0b11111, nonce)
→ TPM2_Quote structure + ECDSA P-384 signature
3. reviewer verifies signature with device certificate (slot 1        key)
4. reviewer independently recomputes PCR 1 from the JSONL ledger
and confirms it matches the quoted PCR 1
5. reviewer reads PCR 0 to see the current tier and full transition log
```

6. Offline Verification Workflow

A reviewer with read access to the JSONL audit ledger and the device certificate can independently verify the ledger without any active connection to the HSM. The procedure is fully offline and uses only public-key operations.

6.1 Inputs

- JSONL audit ledger file (text, one entry per line).
- Device certificate (X.509, contains slot 0 key for audit signing and slot 1 key for PCR quote).
- Optional: live PCR quote with reviewer-supplied nonce, if the reviewer wants to bind verification to a specific point in time.

6.2 Procedure

Step	Action	Pass criterion
1	For each ledger entry: verify ECDSA P-256 signature against canonical-JSON payload	All signatures valid
2	Compute PCR 1 by chaining SHA-256(prev || SHA-256(payload)) from genesis	Final value matches PCR 1 in live quote
3	Read PCR 0 from live quote · decode tier transition log	Current tier matches operational expectation
4	Read PCR 4 from live quote	Zero (all-zero bytes) indicates no tamper event since provisioning
5	Verify PCR-quote signature against slot 1 key in device certificate	Signature valid · nonce matches reviewer-supplied nonce

6.3 Worked Example (CLI)

```
$ authrex-hsm-cli verify-ledger \
--ledger /var/lib/authrex/ledger.jsonl \
--cert ./blade-agent-hsm-001.crt \
--quote ./quote-2026-05-18-1430.bin \
--nonce ./reviewer-nonce.bin

[1/5] Loaded 18,432 ledger entries
[2/5] Verifying signatures ......... 18,432/18,432 OK
[3/5] Recomputing PCR 1 chain ...... ok
[4/5] PCR 1 (computed): 3f:9c:21:...:88
PCR 1 (quoted): 3f:9c:21:...:88 MATCH
[5/5] PCR 0 (tier log): T3 → T2 (3 times) → T3 (3 times) · current: T3
PCR 4 (tamper): all-zero · no tamper since provisioning
Quote signature: verified against slot 1 key

LEDGER VERIFICATION COMPLETE — INTEGRITY INTACT
```

7. Incident Response

A tamper trip is the most consequential incident the device can report. The trip causes permanent zeroization of all key material; no software recovery exists. The procedure below assumes a tamper trip has occurred; for other incidents (failed signing, ABI errors, slow signing) see §8.

7.1 Detection

- AUTHREX-AGENT reports the agent in ABORT state.
- Device alarm LED is latched red (visible on USB-A enclosure or M.2 chassis vent).
- Any subsequent ABI call returns status 0xFF.
- The last successful PCR quote shows PCR 4 non-zero.

7.2 Response Sequence

Step	Action	Owner
1	Stop all dependent agent instances (AUTHREX-AGENT auto-aborts on tamper)	Platform operator
2	Capture the final PCR quote with reviewer-supplied nonce	Security engineer
3	Preserve the ledger file as-is (do not append further entries)	Security engineer
4	Physically secure the device · do not remove from chassis	Site reliability
5	Conduct root-cause review (environment, supply chain, human factor)	Security team
6	Decide: replace with new device (recommended) or re-provision existing	Security team
7	If re-provisioning, document PCR 4 cause string and the re-provisioning event	Operator
8	If replacing, retain tampered device for forensic analysis	Security team

7.3 Forensic Notes

A tampered device retains the ability to answer one query: a PCR-4 read attestation showing the tamper cause hash. The slot 0 audit-signing key recorded at provisioning remains externally verifiable; all ledger entries up to the tamper event are externally verifiable using that key. The audit ledger is therefore not invalidated by the tamper event — it is bounded by the timestamp of the last pre-tamper entry.

8. Troubleshooting

Symptom	Likely cause	Resolution
Device does not enumerate on USB-A	Faulty USB cable, hub power budget, or device damage	Try direct port (not hub); test on second host; if still failing, RMA
ABI returns status 0xFE on every call	Firmware fault · device requires re-provisioning	Schedule provisioning ceremony · update firmware to latest revision before
Signing latency suddenly increases	I²C clock degradation or thermal throttling near +75 °C	Inspect chassis cooling; if persistent, check tamper-trip temperature log
audit_sign returns 0x01 (ledger mismatch)	Host process holds stale ledger view OR rewrite attempted	Re-read PCR 1 from device; if mismatch persists, treat as tamper (UC-3)
tool_auth returns 0x04 (not in policy)	Tool not listed in active PCR 2 policy	Update tool-allowlist policy, re-extend PCR 2 with new hash, restart agent
spawn_quorum_sign returns 0x06	Below 4-of-5 voter threshold	Confirm voter availability; do not bypass quorum
PCR quote signature fails verification	Wrong device certificate · clock-skew on signing host	Re-fetch device cert from operator PKI; confirm device-id matches
Alarm LED red but no tamper recorded	Power-on self-test failure (very rare)	Power-cycle once; if persistent, RMA · do not provision

8.1 Logging

AUTHREX-AGENT emits a host-side log line for every HSM call at INFO level by default. For deeper debugging, set `AUTHREX_HSM_LOG=DEBUG` in the agent environment to enable byte-level dumps of every ABI request and response. Debug logs must not be retained in production; they include raw context-hash inputs.

8.2 Support and Reporting

BLADE-AGENT-HSM is a reference design. There is no commercial support contract. For questions or issues, file an issue at the planned reference repository github.com/burakoktenli-ai/blade-agent-hsm or correspond via the AUTHREX Systems address. Site reliability incidents on production deployments should be handled by the operating organization; the reference design has no claim of production fitness.

End of Integration Guide Revision 1.0. See ICD-AGENT-HSM-001 for hardware detail and the SSRN working paper for the research framing.