

Axiom: Post-Quantum Hardware-Attested Provenance

Axiom Signer & Axiom Identifier Core Technical Specification

Patent Pending

The industry has spent decades protecting secrets. Axiom protects identity.

For decades, the software industry has relied on portable secrets to establish trust: private keys, certificates, passwords, configuration files, and signing credentials. Enormous effort has gone into protecting them because if they're compromised, trust is compromised.

But secrets can be copied.

Secrets can be stolen.

Secrets can be replayed.

An attacker doesn't have to break your cryptography if they can reproduce your identity. Axiom was built on a different principle: **Identity should not be a portable secret. It should be an intrinsic property of authenticated hardware.** That principle changes the foundation of digital trust.

The Problem

Today's software signing systems answer one question: *Did someone possessing the private key sign this artifact?* They do not answer a more important question: *Did this artifact originate from authenticated hardware that we explicitly trust?*

In today's cloud-first world, virtual machines are cloned, containers are replicated, infrastructure is automated, and signing environments can be recreated in minutes. Traditional trust models assume that protecting a private key is enough. It isn't. If an attacker compromises a signing environment, duplicates credentials, or reproduces software-managed secrets, traditional systems often cannot distinguish the original from the copy. The signature verifies—even though the identity behind it has fundamentally changed.

The software supply chain doesn't simply have a cryptography problem. It has an identity problem.

A Different Model of Trust

Rather than protecting portable secrets, Axiom protects identity itself. Instead of storing signing identities in files, cloud vaults, or software-managed key stores, Axiom derives its signing identity from authenticated hardware and binds that identity to post-quantum cryptography.

The result is an identity that cannot simply be copied to another machine. An attacker may compromise a server, steal software, or clone infrastructure—but they cannot take the authenticated hardware identity with them. That is the security boundary Axiom establishes. To bring this trust model out of complex command-line pipelines and make it instantly usable across heterogeneous enterprise environments, the platform bifurcates into two tightly coupled layers:

1. **Axiom Signer:** The core, low-level execution engine that communicates directly with machine hardware subsystems and enforces post-quantum signing contracts.

2. **Axiom Identifier:** The stateless, local-first multi-platform desktop application that coordinates package generation, ingests third-party deliverables, extracts automated metadata, and manages the lifecycle of attested software bundles.

Technical Architecture & Core Subsystems

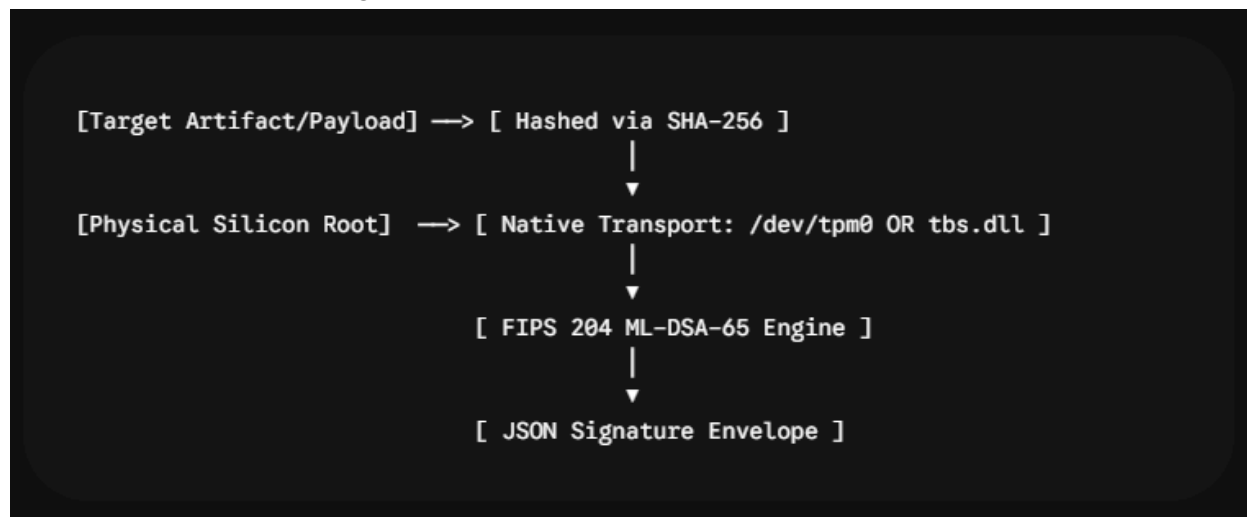
1. Axiom Signer: Core Hardware Interrogation

Axiom Signer is a stateless, post-quantum software signing engine that cryptographically binds software provenance to physical hardware. Every signature is rooted in authenticated silicon rather than software-managed secrets.

To achieve true multi-platform parity across modern DevOps environments, Axiom Signer operates natively across distinct hardware boundaries:

- **Linux Architectures:** The engine directly addresses Linux character devices (`/dev/tpmrm0` or `/dev/tpm0`). The local Trusted Platform Module (TPM 2.0) manufacturer-issued Endorsement Key (EK) certificate is retrieved and validated locally against an embedded collection of trusted semiconductor root certificates.
- **Windows Architectures:** The engine dynamically interacts with Windows TPM Base Services (TBS) by mapping procedures from native `tbs.dll` system libraries. Utilizing atomic hardware command submission structures (`Tbsip_Submit_Command`), Axiom establishes isolated cryptographic contexts directly on Windows motherboards, entirely removing the need for legacy virtual network sidecars or cloud lookups.

Once authenticated, Axiom Signer uses standard TPM 2.0 key derivation to produce a deterministic, hardware-resident signing identity. The private signing material never leaves the silicon. The engine then implements **FIPS 204 ML-DSA-65**, producing a standard quantum-resistant JSON signature envelope.



2. Axiom Identifier: The Lifecycle & Bundle Orchestrator

The Axiom Identifier encapsulates the execution engine into a zero-dependency, local-first utility designed to reduce operational friction. It provides a deterministic, visual interface for both software creators (Export & Sign) and software consumers (Ingest & Verify).

Cradle-to-Grave Package Architecture ([.axiom.zip](#))

When a user targets a file or directory for export, Axiom Identifier constructs a rigid, standardized, air-gapped deployment bundle containing four critical supply chain pillars:

1. `payload/`: The raw, un-modified binary assets, libraries, or document directories being governed.
2. `signature.jsonl`: The cryptographic signature envelope generated straight from Axiom Signer's standard output.
3. `cyclonedx.sbom.json`: An automatically generated Software Bill of Materials indexing the deep dependency tree and deterministic file hashes of the payload.
4. `oscal.assessment.json`: A machine-readable compliance manifest containing the environmental profile data, execution metadata, and hardware binding declarations passed to the validator.

Architectural Resiliency & Development Fallbacks

To guarantee continuous operations across restricted enterprise landscapes—where developers may be running on legacy hardware or in restricted environments lacking active BIOS-level TPM configurations—Axiom Identifier implements a clear cryptographic contingency loop:

- **Primary Loop (Silicon Attestation):** The tool invokes Axiom Signer with zero arguments, anchoring the transaction to physical motherboard hardware. The signature envelope explicitly signs and outputs a native hardware anchor contract.
- **Contingency Loop (Hardware-Bound Fallback):** If the local machine lacks an active TPM, the wrapper intercepts the native hardware fault, transitions to an explicit Software-BYOK track via standard file parameters, and reads the host's immutable cryptographic operating system footprint (`HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid`). It injects this unique hardware fingerprint directly into the OSCAL compliance stream alongside a designated `"anchor-type": "hardware-bound-fallback"` declaration.

This guarantees absolute provenance tracking and clear audit transparency without ever halting real-world engineering velocity.

Operational Paradigm Shift

Design Dimension	Traditional Cloud Key Vaults / Legacy Code Signing	The Axiom Paradigm
Identity Storage	Software-managed files, certificates, or cloud keys that can be copied, exported, or leaked.	Locked in physical motherboard hardware. Non-portable and un-copyable.
Connectivity	Requires persistent connections to cloud Certificate Authorities, API keys, or web token endpoints.	100% Stateless & Air-Gapped. Operates entirely local-first without a single external cloud control plane dependency.

Delivery Model	Detached, loose signature and JSON metadata files that are easily modified, lost, or separated in transit.	A self-contained <code>.axiom.zip</code> capsule that bundles the payload, post-quantum signature, SBOM, and compliance manifest together.
-----------------------	--	--

Why This Matters

Post-quantum cryptography is often viewed as a simple algorithm replacement: Replace RSA, replace ECC, and move on. Axiom takes a different view. The transition to post-quantum cryptography is an opportunity to rethink digital identity itself. Instead of replacing one signing algorithm with another while preserving the same flawed assumptions about portable, software-managed keys, Axiom replaces the underlying trust model. Traditional systems protect keys; Axiom protects identity. By pairing deep, physical silicon interrogation through Axiom Signer with an intuitive, zero-dependency visual deployment workflow via Axiom Identifier, the platform achieves the ultimate security objective: **It prevents an attacker from taking a trusted machine's identity with them.**

Axiom is a Patent Pending Technology