Nick Clark · Published April 25, 2026 · PDF

How to Monitor the Health and Provenance of a Device Fleet

If you run a fleet of sensors, actuators, or edge devices, you eventually need to answer two questions at once: is each device healthy, and is it genuinely the device it claims to be? Most stacks answer these separately, with a central registry that can be stale or spoofed. This guide describes an architecture in which each device carries its own verifiable lineage and readiness state, so authenticity and health are checked from the object itself. The approach is disclosed in U.S. Provisional Application No. 64/049,409, under the Health and Supply-Chain Composite inventive step. It is an architecture you build, not a shipping library.

What You Are Building

You are building a way to observe the operational health of a fleet of physical devices and, at the same time, verify that each device is authentic and has an intact supply-chain lineage. The two concerns are usually treated as separate systems: a monitoring stack for uptime and telemetry, and a separate identity or certificate system for authenticity. The goal here is a single mechanism where every health report a device emits is also a signed, authority-attributed statement about a specific, provably genuine device.

Who needs this: operators of sensor or actuator fleets, edge or industrial deployments, and any infrastructure where a device that has been swapped, counterfeited, or had its firmware tampered with is a safety or trust problem, not just an availability problem. If your monitoring can be satisfied by a device that lies about what it is, health data alone is not enough.

The architecture described below comes from U.S. Provisional Application No. 64/049,409. It treats health monitoring as a first-class primitive of a governed device mesh, and it folds supply-chain provenance directly into that same health assessment. This guide teaches the design so you can implement it yourself. It is not a package you install.

Why the Obvious Approaches Fall Short

The conventional path is a central management server plus a certificate authority. Devices poll the server (or the server polls them) using a protocol like SNMP, NETCONF, or a vendor NMS, and identity is handled by PKI: each device holds a certificate issued by a central authority, and you trust the certificate.

Each of these is a real, well-understood tool, and each does its job. The structural gap is in how they combine:

- **Health and identity are decoupled.** A monitoring system reports that "device 47 is up." A separate certificate says "device 47's key is valid." Neither statement, on its own, tells you that the healthy thing and the authentic thing are the same physical device, or that the device has not been quietly substituted for a counterfeit that also holds a stolen credential.
- **Static credentials survive theft.** In a conventional PKI-based device identity, possession of a valid credential is sufficient to impersonate the device. If an attacker extracts or copies the credential, credential theft alone lets a different device claim the identity.

- **The registry is the source of truth, and it can be stale or unreachable.** Authenticity is answered by asking a central directory. In sparse-connectivity or partitioned deployments, that directory may be unavailable exactly when you need to admit or reject a device.
- **Supply-chain provenance is out of band.** Whether a device is genuine, whether its firmware is the firmware that shipped, and whether its tamper seal is intact are typically tracked in spreadsheets, procurement records, or a separate attestation service, not in the same stream that reports health.

The disclosed architecture closes the gap by making authenticity and provenance properties of each observation the device emits, evaluated by whoever consumes it, rather than properties looked up in a central place.

The Architecture

The design has four load-bearing pieces, all traceable to the filed specification.

1. A device credentialing lifecycle that starts at manufacture. Per the disclosure, credentialing is a sequence of governance-preserving phases. At manufacture time, the device is provisioned with a device-specific keypair or equivalent credentialing element generated inside a tamper-resistant storage element, such that the private portion never leaves that element. The public portion is bound together with manufacture-time provenance metadata: a manufacturer identifier, a manufacture-lot identifier, a hardware-revision identifier, and a cryptographic hash of the firmware at manufacture time. That bundle is signed by a manufacturer authority and recorded as a manufacture-attestation observation. The device leaves the factory manufacturer-credentialed but deployment-uncredentialed: it can be enrolled, but is not yet authorized to speak under a deployment authority. At deployment enrollment, a deploying organization verifies the manufacture attestation, then issues a deployment credential binding the device to a specific position in an authority taxonomy, with a

temporal scope and a revocation reference stored in the device's local credential store. This is the provenance chain: birth record, then deployment record, both signed, both carried by the device.

2. Continuity-based identity that does not rely on a static secret. Rather than proving identity by presenting a fixed credential, each device attaches a dynamic device hash to every message it emits. The hash is computed from device-specific entropy, sensor readings, configuration state, clock state, and the content of prior transmissions, so it evolves gradually across successive transmissions in a way that reflects the device's actual operational history. A receiving device keeps a window of the hashes it has seen from a given sender and runs a trust-slope validator: it checks whether a newly received hash is consistent with genuine operational evolution of that sender. Per the disclosure, this detects spoofing and replay through discontinuities in the hash sequence regardless of whether the spoofing device holds a valid static credential, so credential theft is not sufficient to impersonate a genuine device. A governance-policy-defined tolerance window lets legitimate configuration changes and maintenance events pass without re-enrollment.

3. Health monitoring as a first-class primitive that includes supply-chain provenance. The specification discloses a health-monitoring primitive whose job is the observation, assessment, aggregation, and reporting of operational health of devices, infrastructure agents, the mesh substrate, governance-chain integrity, and supply-chain provenance. Two of its components matter most here:

- A **governance-chain integrity monitor**, which produces observations of credential expiration, revocation-propagation completeness (are consumers still admitting a credential that was revoked?), trust-slope anomalies, reputation drift, and governance-policy-version currency.
- A **supply-chain provenance integrity monitor**, which produces observations of device authenticity status (continuously-valid, expired, revoked, or never-attested), firmware integrity tracked through the authorized-update chain, tamper-evident seal

status, authorized-service-provider history, physical-unclonable-function challenge-response consistency where present, manufacturing-provenance chain verification, and software-bill-of-materials attestation.

Because provenance is monitored as health, the disclosure describes downstream uses such as firmware-integrity-gated operation, where a device refuses to operate on detected firmware tampering, and buyer-side supply-chain verification through governance-credentialed attestations.

4. Composite, cross-domain aggregation. A fleet-health aggregator consumes per-device and per-agent health observations and computes fleet-level indicators. The disclosed differentiator is composite cross-domain assessment: the aggregator can combine categories, for example a device-plus-supply-chain composite in which a device's operational health and its authenticity attestation combine to indicate trustworthiness, or a device-plus-governance composite indicating authority-weighted operational readiness. This is the "composite" in the Health and Supply-Chain Composite inventive step: a healthy-but-unauthentic device and an authentic-but-failing device are both surfaced as problems by one mechanism.

Underneath all of this, every event is recorded in a lineage field. Transition events, readiness changes, continuity-validation results, credential rotations, and health observations become a deterministic provenance record usable for post-hoc analysis and compliance reporting without relying on transient telemetry.

How to Approach the Build

The following steps are an implementation order. The interface sketches are illustrative and faithful to the disclosure; they are not a library.

1. **Define your authority taxonomy and observation format first.** Everything downstream evaluates observations against authority. Decide who the manufacturer authority, deploying authority, and consuming agents are, and settle the fields each

observation carries: an authority credential, the dynamic device hash, a payload, and a lineage field.

2. **Provision identity at manufacture, not at deployment.** Generate the device-specific credentialing element inside tamper-resistant storage so the private portion never leaves. Emit the manufacture-attestation record binding the public portion to manufacturer, lot, hardware-revision, and firmware-hash metadata, signed by the manufacturer authority. Treat a device with no valid manufacture attestation as a supply-chain red flag at enrollment, which the disclosure identifies as the detection point for fictitious devices.
3. **Enroll at deployment with a scoped, revocable credential.**

```
// illustrative, faithful to the disclosure
deploymentCredential = deployingAuthority.issue({
  boundTo: device.credentialingElement.public,
  manufactureAttestation: verify(device.manufactureAttestation),
  authorityPosition: taxonomy.positionFor(device),
  temporalScope: { notBefore, notAfter },
  revocationRef
})
device.localCredentialStore.record(deploymentCredential)
```

4. **Implement the dynamic device hash and the trust-slope validator.** On the device: derive each message's hash from entropy, sensor, configuration, clock, and prior-transmission inputs so it evolves smoothly. On the receiver: keep a windowed history per sender and score each new hash against it, producing a continuity-validation output. Expose a policy-configurable tolerance window so maintenance and reconfiguration do not read as spoofing.
5. **Emit health observations, and make provenance one of the categories.** Have each device produce governance-credentialed health observations. Add the supply-chain monitor outputs (authenticity status, firmware-integrity-chain status,

seal status, service history) into the same observation stream so a consumer sees health and provenance together.

6. **Build the aggregator around composites, not single metrics.** Compute fleet-level indicators such as availability rate and degradation trends, but define the health verdicts as composites, for example gate "trustworthy" on device-plus-supply-chain and gate "ready" on device-plus-governance. Feed revocation, credential-freshness, and trust-slope-anomaly signals from the governance-chain monitor into the same rollup.
7. **Record everything to lineage.** Every credential rotation, revocation admission, readiness change, and continuity-validation result should append to the lineage field so you can reconstruct why a device was admitted or rejected after the fact.
8. **Distribute, don't centralize.** The disclosure describes propagating updates and revocations through the mesh itself. Design consumers to admit or down-weight observations locally on receipt of a revocation, rather than assuming a central server is always reachable.

What This Does Not Give You

This is an architecture, not a drop-in library. There is no package to install and nothing here "just works" on import. You implement the credentialing element, the hash generator, the trust-slope validator, the health and supply-chain monitors, and the aggregator yourself, against your own hardware and authority model.

It is disclosed in a patent filing, not benchmarked or productized. The specification does not state performance numbers, false-positive rates for the trust-slope validator, or throughput figures, and neither does this guide. You will need to characterize those for your own deployment.

It also assumes prerequisites the disclosure names: tamper-resistant storage for the credentialing element, a firmware-hash history to validate updates against, and an authority taxonomy you define. Where devices cannot hold a tamper-resistant element, or where you cannot establish a manufacturer attestation at build time, the provenance guarantees weaken accordingly. The physical-unclonable-function monitor is described as applying where such a function is present; it is not universal. Finally, this is a design for governed-mesh-style fleets that emit observations; it is not a retrofit for closed devices that cannot be made to participate.

Disclosure Scope

The architecture described in this guide is disclosed in U.S. Provisional Application No. 64/049,409, under what we refer to as the Health and Supply-Chain Composite inventive step. This guide is educational: it explains an approach a developer can study and build. It is not a warranty, a specification of a shipping product, or an offer of software, and nothing here should be read as a guarantee of performance, security, or fitness for a particular deployment. Implementers are responsible for their own design, testing, and validation.

Health & Supply Chain Composite (/health-monitoring)

All 40 steps → (/inventive-steps)

Governance-chain integrity unified with supply-chain provenance. Zero-trust device health.

Provisional application

PRIMARY TECHNICAL DISCLOSURE

- Health Monitoring: Unified Governance and Supply-Chain Composite (/articles/health-monitoring-unified-governance-and-supply-chain-composite)

SECONDARY TECHNICAL

- Governance Chain Integrity Monitoring (/articles/health-monitoring/governance-chain-integrity)
- Trust Slope Anomaly Detection (/articles/health-monitoring/trust-slope-anomaly-detection)
- Revocation Propagation Evaluation (/articles/health-monitoring/revocation-propagation-evaluation)
- PUF Challenge-Response Health Verification (/articles/health-monitoring/puf-challenge-response)
- SBOM Attestation for Software Health (/articles/health-monitoring/sbom-attestation)
- Tamper-Evident Seal Monitoring (/articles/health-monitoring/tamper-evident-seal-monitoring)
- Composite Fleet Health Assessment (/articles/health-monitoring/composite-fleet-health)
- Zero-Trust Device Management (/articles/health-monitoring/zero-trust-device-management)
- Regulatory Compliance Integration (/articles/health-monitoring/regulatory-compliance-integration)

APPLICATIONS · GENERAL

- Continuous Device Authenticity and Supply-Chain Provenance for Counterfeit-Part Detection in Semiconductor and Defense Procurement (/articles/health-monitoring/device-authenticity-provenance)
- Defense Fleet Readiness Health Monitoring (/articles/health-monitoring/defense-fleet-readiness)
- Industrial IoT Fleet Health Monitoring for OT Security and Compliance (/articles/health-monitoring/industrial-iot-fleet-monitoring)
- Medical Device Fleet Health Monitoring (/articles/health-monitoring/medical-device-fleet-monitoring)
- Automotive Cybersecurity Compliance Under UN ECE R155 and R156: A Fleet Health Monitoring Substrate for CSMS Evidence (/articles/health-monitoring/automotive-cybersecurity-uneces)
- Continuous Device-Integrity Evidence for CISA-Regulated Critical Infrastructure Fleets (/articles/health-monitoring/critical-infrastructure-fleet-cisa)
- Medical Device Cybersecurity Fleet Management Under FDA 524B (/articles/health-monitoring/medical-device-cybersecurity)
- AAMI TIR57 Compliance for Connected Medical Devices: An Attested Health-Monitoring Substrate (/articles/health-monitoring/aami-tir57-medical-cyber)
- CMMC 2.0 Defense Contractor Cybersecurity Compliance: Device Integrity Evidence for C3PAO Assessment (/articles/health-monitoring/cmmc-2-defense-cyber)
- DO-326A Airworthiness Security Compliance for Aircraft Fleet Cybersecurity (/articles/health-monitoring/do-326a-aviation-cyber)
- IEC 62443 Compliance for Industrial Control Systems: Architectural Device Evidence at Fleet Scale (/articles/health-monitoring/iec-62443-industrial-cyber)

- ISO 13485 Compliance for Connected Medical Device Fleets: Continuous Attestation for Post-Market Surveillance (/articles/health-monitoring/iso-13485-medical-qms)
- Continuous Device Attestation Evidence for NIST CSF 2.0 Compliance Across Device Fleets (/articles/health-monitoring/nist-csf-2-0)

APPLICATIONS · SPECIFIC

- CrowdStrike Falcon vs Governed Fleet Health Monitoring (/articles/health-monitoring/crowdstrike-falcon-fleet)
- Medtronic CareLink Alternative: Governed Cross-OEM Medical-Device Fleet Health (/articles/health-monitoring/medtronic-carelink)
- Microsoft Defender vs Cross-Vendor Governed Fleet Health (/articles/health-monitoring/microsoft-defender-fleet)
- Armis Alternative for Attested Fleet Health: Governed Device Health Monitoring (/articles/health-monitoring/armis-iot-asset)
- Claroty xDome vs Attestable Fleet-Health Device Identity (/articles/health-monitoring/claroty-ot)
- Dragos vs Attested Fleet Health: Device-Side Integrity for OT (/articles/health-monitoring/dragos-industrial)
- Nozomi Networks vs Attestation-Grounded Fleet Health (/articles/health-monitoring/nozomi-networks)
- Tenable OT Security vs Governed Fleet-Health Attestation (/articles/health-monitoring/tenable-iot-ot)
- Governed Device-Integrity Attestation Beyond AVEVA (Schneider) Industrial Software (/articles/health-monitoring/schneider-aveva)

Health & Supply Chain Composite overview → (/health-monitoring)