The Device Should Not Know Its Own Identity

CoreBond Research Series — CBRS-002
August 2026
Todd Kirton
DOI: 10.5281/zenodo.22027779

A provocation about persistence, privacy, recovery, and trust in device identity

Executive Summary

A device has an identity. That sentence sounds simple enough that we rarely stop to examine it. In practice, the "identity" may be a key, certificate, serial number, manufacturer credential, registry record, or some combination of them. The model is useful. It supports automated first-contact trust, provenance, onboarding, inventory, recovery, and operation when a remote authority is unavailable. Durable asset identifiers solve a related problem: finding and accounting for the same physical thing across a long lifecycle. These mechanisms exist for good reasons.

The trouble starts when all of those jobs are folded into the same word. Manufacturer provenance is not ownership. An operational certificate is not current software state. Attestation evidence is not authorization. A peer-facing identifier does not necessarily need to live as long as the physical asset behind it. Mature architectures already separate some of these jobs, often preserving a durable root while rotating or replacing what is used in day-to-day operation.

This paper does not argue that devices should have no persistent identity. The title is a provocation, not a prescription to erase every durable root. The evidence does not support that extreme. The narrower question is more useful: which parts of a device's identity must persist, which should rotate, which should remain private, and which should be evaluated fresh at the moment of trust? The answer may be less about one identity artifact than about several identity-related properties with different lifetimes, authorities, disclosure rules, and recovery requirements.

1. The Identity We Rarely Define

Ask whether an embedded device needs an identity and the answer is usually easy: yes. Ask what, exactly, that identity is, and the answer gets slippery.

Is it the device's serial number? The private key in a secure element? An X.509 certificate? A manufacturer-issued credential? A network address? An owner-assigned certificate? A TPM endorsement or attestation key? A record in a cloud registry? A collection of measurements proving that the expected firmware is running?

Every one of those can matter. They are still not the same thing.

An identifier distinguishes. A credential supports a claim. A private key proves control of cryptographic material under a protocol. A certificate binds a public key to claims made by an issuer. Attestation provides evidence about state, composition, or trustworthiness. Authorization decides what an entity may do. None of those definitions requires us to pretend that one artifact carries the full meaning of the device.

Modern attestation architecture makes the distinction especially clear. The IETF Remote ATtestation procedureS (RATS) architecture separates the Attester that produces evidence, the Verifier that appraises that evidence using policy and reference information, and the Relying Party that decides what to do with the result.[1] Recognition is already a distributed process. The device supplies something important, but it does not make the final trust decision about itself.

> *Before asking how strongly a device should protect its identity, ask what we have bundled into the word identity in the first place.*

2. Why We Gave Devices Durable Identity

There is an easy version of this argument: stored device identity is old baggage, and newer architectures should get rid of it. That version does not survive contact with real systems.

Durable device identity solves difficult problems. IEEE 802.1AR formalizes secure device identifiers that can provide a manufacturer-rooted Initial Device Identifier and locally significant identities for later domains.[2] Trusted Computing Group guidance builds on the same distinction for device identity and attestation keys.[3] BRSKI uses manufacturer-installed identity to bootstrap a device into a new administrative domain, where it can receive a local identity appropriate to that environment.[4] FIDO Device Onboard uses manufacturing and ownership-transfer mechanisms so a device can be built before its final owner or service provider is known.[5]

The reasons are practical. Automated first-contact onboarding typically requires some pre-existing basis for trust: a manufacturer root, a protected bootstrap credential, a physically conveyed secret, or another controlled process. A manufacturer may need to prove provenance. An owner may need to recover or re-enroll a device after operational credentials expire. An industrial controller may spend long periods offline. And outside authentication itself, safety and asset-management systems may need an identifier that survives ownership changes and software updates.

The strongest counterexamples to the broader idea that persistent identifiers are merely legacy baggage come from outside authentication architecture. A vehicle identification number is not a cryptographic credential, and a medical-device UDI does not authenticate an endpoint. They solve a different identity problem: durable physical-asset identification and traceability. NHTSA relies on VINs in safety-defect and recall programs.[6] The FDA's Unique Device Identification system supports precise identification and postmarket surveillance of medical devices.[7] The U.S. Department of Defense uses item-level identification for lifecycle traceability of selected materiel.[8] These examples do not prove that an operational authentication credential should persist forever. They prove that some identity claims about the physical asset may need to.

The current model is not foolish. It solves real problems. The harder question is whether those problems all require the same identity artifact, with the same lifetime, visibility, and authority.

3. One Word, Several Jobs

Separate the jobs and device identity starts looking less like one object and more like a stack of claims.

A manufacturer or provenance identity answers: where did this device come from? An owner or domain identity answers: who governs it now? An operational credential answers: what cryptographic material does it currently control? Attestation answers: what state is it in? Authorization answers: what may it do here? A session identity

answers: is this the same participant in this exchange? A privacy-visible identifier answers another question entirely: what information about continuity should this peer be allowed to observe?

Those claims do not naturally share one lifetime. Picture a controller bolted into an industrial plant. Its manufacturer provenance may need to survive for decades. The plant owner's domain credential may change when the asset is sold or the site is reorganized. Its firmware state can change tonight. Its authorization to open a valve can change with policy. Its session key should die in minutes or hours. One physical controller can participate in all of those claims without requiring every claim to live as long as the steel cabinet around it.

The RATS model already keeps evidence separate from appraisal and relying-party decisions.[1] IEEE and TCG models distinguish manufacturer-rooted and locally assigned device identities.[2][3] NIST's IoT onboarding work separates manufacturer, purchaser, owner, onboarder, manager, user, maintenance, reset, and end-of-life responsibilities.[9] The industry may use the convenient phrase device identity, but its own architectures already reveal several different relationships hiding underneath it.

Once the jobs are separated, their natural lifetimes stop looking identical. If a property changes independently, it is fair to ask why it must inherit the persistence of the device's deepest root.

4. The Cost of Persistence

Persistence earns its keep by preserving continuity. It also creates obligations.

Long-lived identifiers can create privacy and correlation risks. RFC 9039 warns that stable device identifiers may enable tracking and device-specific targeting.[10] The IETF's work on randomized and changing MAC addresses documents the pressure to reduce that linkability, but it also documents the price: changing identifiers can interfere with network state, authentication associations, troubleshooting, and policy.[11] Privacy is not solved by making everything ephemeral. The problem simply changes shape.

Credentials have their own clock. Production IoT platforms expect certificate renewal, replacement, revocation, and reprovisioning. AWS recommends unique per-device certificates and supports rotation and replacement.[12] Microsoft documents certificate rollover and renewal procedures that preserve an existing device relationship while changing the operational certificate representing it.[13][14] That distinction is important: the thing we want to recognize can persist while the credential used to recognize it changes.

Cryptographic algorithms introduce another clock. NIST's crypto-agility work is built around the reality that algorithms, protocols, software, firmware, hardware, keys, and certificates eventually need to change.[15] Current post-quantum migration guidance reinforces the same problem for systems expected to remain in service through a cryptographic transition.[16] A physical asset can outlive the algorithm chosen when it was manufactured.

Operational technology makes the mismatch harder to ignore. CISA and NIST guidance treat certificate lifecycle, availability, update schedules, and authentication overhead as real operational concerns for long-lived cyber-physical systems.[17][18] A device identity architecture cannot be judged only by cryptographic elegance if recovery requires a maintenance outage that the physical process cannot tolerate.

None of this makes persistence a mistake. It means persistence has a scope and a cost. That is enough to ask where it actually belongs.

5. The Industry Is Already Separating the Layers

This does not require tearing down existing identity systems and starting over. Mature systems already separate selected layers.

Several established architectures already use a deeper root to create or recover a shorter-lived operational identity. Secure device-identity standards distinguish manufacturer-rooted identities from identities assigned by the local domain.[2][3] BRSKI follows that pattern during automated onboarding, while FIDO Device Onboard separates ownership transfer from deeper device-attestation material.[4][5] Azure provides a commercial example in which an existing device relationship can survive renewal of the operational certificate that represents it.[14]

Automotive systems provide an especially useful example because privacy, safety, and accountability collide in the same architecture. ETSI Intelligent Transport Systems security keeps a canonical identity while using authorization tickets or pseudonym certificates for operational communications.[19] The visible identity can change during normal operation even though the vehicle remains the same underlying asset. ETSI also documents why that is not free: pseudonym changes can affect continuity and introduce abuse risks such as Sybil behavior if handled poorly.[20]

Privacy-preserving attestation pushes the distinction further. Direct Anonymous Attestation is designed so a verifier can establish the authenticity of attestation evidence from a legitimate member of an approved group without learning a persistent unique identity for that attester. The current RATS DAA document is still an Internet-Draft, so it should not be treated as settled standard text, but the architecture demonstrates an important point: attestation authenticity and persistent individuality are not the same requirement.[21]

These examples do not prove that layered identity is always better. They prove something narrower: a system can preserve a durable root while limiting which identity is exposed or used for a particular relationship.

6. What Must Survive?

Layered identity looks clean on a diagram. Failure is where it gets uncomfortable.

BRSKI explicitly considers the death of a manufacturer, devices kept in storage for many years, disconnected deployments, and emergency onboarding paths.[4] Those failure cases matter because externalizing trust does not remove the need for durable authority. It moves the burden. If the manufacturer, verifier, registry, or owner record is the only place that remembers why a device should be trusted, that institution has become part of the device's recovery architecture.

FIDO Device Onboard provides a useful opposite pattern. During resale or re-onboarding, operational state can be removed while selected FDO credentials remain so a new ownership relationship can be established.[5] The retained material survives precisely so other secrets and owner-specific state can be safely discarded.

Operational certificate expiry shows the same hierarchy. Azure documents recovery in which an expired operational certificate can be replaced using a retained onboarding credential.[14] The renewable credential is not the deepest trust anchor. If everything had been erased together, remote recovery would become harder or impossible.

Repair creates an even uglier question. Replace storage and the credentials may disappear while the physical asset remains. Replace the motherboard or security module and the original cryptographic root may disappear while the chassis, serial record, ownership, and mission remain. Replace firmware and attested state changes deliberately while provenance does not. NIST's onboarding concepts assign lifecycle responsibilities for maintenance, reset, repair, decommissioning, and end of life, but there is no universal rule for when a repaired device becomes a new identity.[9]

Offline operation imposes another constraint. A cloud-only verifier can become an availability dependency. A local verifier reduces that dependency but must protect and synchronize its own state. Cached decisions can become stale. Long-lived credentials support autonomy but weaken rapid revocation. Emergency physical procedures restore availability by intentionally lowering automated assurance.

Failure reveals why some durable anchor may still be necessary. In the architectures examined here, erasing all durable device state does not erase the recovery problem. It pushes that problem onto another durable authority, retained credential, custody record, or physical process capable of re-establishing why the device should be trusted.

7. The Better Question Is About Properties

The title of this paper is deliberately uncomfortable: The Device Should Not Know Its Own Identity. Taken literally, it goes too far. Some systems clearly benefit from durable device-associated identity, and some cannot recover safely without it.

The useful part of the provocation survives in a different form. Persistence, possession, visibility, credential lifetime, ownership, state, authorization, and trust are not the same identity-related property. They should not automatically inherit the same lifecycle simply because they are often bundled in discussion and design under the convenient phrase device identity.

That leads to a more disciplined design test. For each identity-related property, ask: who needs to know it, and for how long? Who may change it? What survives factory reset or ownership transfer? What changes after repair? What must work offline? What happens when the verifier disappears or the cryptography ages out? What can rotate without losing provenance or recoverability?

A durable manufacturer or recovery anchor may deserve decades of persistence. An owner credential may deserve only the lifetime of ownership. Operational credentials may need regular renewal. Attested state should be fresh enough to describe the device now. Authorization may change with policy. A peer-facing identifier may deserve privacy limits that a regulator-facing asset identifier does not.

This decomposition is not automatically more secure, and it does not need a new ontology to be useful. Existing systems already give these claims different issuers, lifetimes, and decision points. More layers can mean more records, more authorities, more synchronization problems, and more recovery paths. The benefit is simply that those engineering differences become explicit enough to argue about instead of being silently inherited.

8. What Deserves to Be Discovered

Future device-identity design should begin with neither extreme. Permanent identity is not obsolete, and one permanent identity need not serve every purpose.

The boundary is what deserves investigation.

How little durable state can a device retain without making first contact, offline operation, or disaster recovery fragile? How should a system preserve provenance while rotating algorithms and operational credentials? When a device is repaired, which changes preserve identity and which create a new asset? Can privacy-preserving identifiers remain accountable when they need to be? How durable must verifier records become when identity moves away from the device? What is the operational cost of managing several scoped identities compared with one stable identity? Under what conditions does fresh evidence add meaningful trust rather than merely another layer of complexity?

Those questions are harder than “should devices have keys?” They are also more useful.

A device may need something durable enough to be recognized after everything around it changes. That does not mean every credential, owner relationship, visible identifier, state claim, and authorization decision should become equally permanent.

> **The future question is not identity versus no identity. It is which properties should persist, for whom, for how long, and why.**

That distinction is not a finished architecture. It is a better place to start.

Research Agenda

- What is the minimum durable identity-related state required for secure first contact and recovery?
- Which forms of provenance must remain globally stable, and which operational identities should be scoped to an owner, domain, peer, or session?
- How should identity survive legitimate repair, component replacement, and cryptographic migration?
- When does privacy-preserving pseudonymity create unacceptable losses in accountability or recoverability?
- How should a system recover when the verifier, manufacturer, certificate authority, or ownership registry disappears?
- What evidence would demonstrate that a layered identity model is operationally better rather than merely more conceptually precise?

References

1. **IETF.** *Remote ATtestation procedureS (RATS) Architecture*. RFC 9334, Jan. 2023. https://www.rfc-editor.org/rfc/rfc9334
2. **IEEE Standards Association.** *IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity*. IEEE Std 802.1AR-2018, Aug. 2018. https://standards.ieee.org/ieee/802.1AR/6995/
3. **Trusted Computing Group.** *TPM 2.0 Keys for Device Identity and Attestation*. Version 1.1. https://trustedcomputinggroup.org/resource/tpm-2-0-keys-for-device-identity-and-attestation/
4. **IETF.** *Bootstrapping Remote Secure Key Infrastructure (BRSKI)*. RFC 8995, Nov. 2021. https://www.rfc-editor.org/rfc/rfc8995
5. **FIDO Alliance.** *FIDO Device Onboard Specification*. Review Draft 02, Version 2.0, Apr. 2, 2026. https://fidoalliance.org/specs/FDO/FIDO-Device-Onboard-RD02-v2.0-20260402/FIDO-Device-Onboard-RD02-v2.0-20260402.html
6. **National Highway Traffic Safety Administration.** *VIN requirements and recall-program interpretation*. https://www.nhtsa.gov/interpretations/ncc-240821-001-letterwmi-foreign-built-vehicles-ford-565final
7. **U.S. Food and Drug Administration.** *Unique Device Identification System: UDI Basics and Benefits*. https://www.fda.gov/medical-devices/unique-device-identification-system-udi-system/udi-basics
8. **U.S. Department of Defense.** *Item Unique Identification (IUID)*. https://www.acq.osd.mil/asda/dpc/ce/ds/unique-id.html
9. **National Institute of Standards and Technology.** *Foundational Concepts in Trusted IoT Device Network-Layer Onboarding: Enhancing Internet Protocol-Based IoT Device and Network Security*. NIST IR 8350, Nov. 2025, doi:10.6028/NIST.IR.8350. https://doi.org/10.6028/NIST.IR.8350
10. **IETF.** *Uniform Resource Names for Device Identifiers*. RFC 9039, June 2021, doi:10.17487/RFC9039. https://doi.org/10.17487/RFC9039
11. **IETF.** *Randomized and Changing Media Access Control (MAC) Addresses: Context, Network Impacts, and Use Cases*. RFC 9797, June 2025, doi:10.17487/RFC9797. https://doi.org/10.17487/RFC9797
12. **Amazon Web Services.** *AWS IoT Core: X.509 client certificates*. https://docs.aws.amazon.com/iot/latest/developerguide/x509-client-certs.html
13. **Microsoft.** *Azure IoT DPS: Roll X.509 Device Certificates*. https://learn.microsoft.com/en-us/azure/iot-dps/how-to-roll-certificates
14. **Microsoft.** *Certificate renewal in Azure IoT Hub certificate management (preview)*. Updated Apr. 15, 2026. https://learn.microsoft.com/en-us/azure/iot-hub/concept-certificate-renewal
15. **National Institute of Standards and Technology.** *Crypto Agility*. CSRC project. https://csrc.nist.gov/Projects/crypto-agility
16. **National Institute of Standards and Technology.** *Migration to Post-Quantum Cryptography*. NCCoE project guidance, updated 2026. https://pages.nist.gov/nccoe-migration-post-quantum-cryptography/
17. **CISA and international partners.** *Secure by Demand: Priority Considerations for Operational Technology Owners and Operators When Selecting Digital Products*. 2025. https://www.cisa.gov/resources-tools/resources/secure-demand-priority-considerations-ot-owners-and-operators
18. **National Institute of Standards and Technology.** *Guide to Operational Technology (OT) Security*. NIST SP 800-82 Rev. 3, Sept. 2023. https://csrc.nist.gov/pubs/sp/800/82/r3/final
19. **ETSI.** *Intelligent Transport Systems (ITS); Security; ITS communications security architecture and security management*. ETSI TS 102 940 V1.3.1. https://www.etsi.org/deliver/etsi_ts/102900_102999/102940/01.03.01_60/ts_102940v010301p.pdf
20. **ETSI.** *Intelligent Transport Systems (ITS); Security; Pseudonym Change Management*. ETSI TR 103 415 V1.1.1. https://www.etsi.org/deliver/etsi_tr/103400_103499/103415/01.01.01_60/tr_103415v010101p.pdf
21. **IETF RATS Working Group.** *Direct Anonymous Attestation for the RATS Architecture*. draft-ietf-rats-daa-09, work in progress. https://datatracker.ietf.org/doc/draft-ietf-rats-daa/

Research Paper 002 | Todd Kirton | August 2026

This paper examines an open engineering question and does not advocate a specific commercial implementation.