MONITORING TRUTH & FLEET RECONCILIATION SERIES · EC-WP-1100

Missing Is Not Zero

Six ways a renewable monitoring platform reports something that is not true, and the design rules that stop each one.

A field taxonomy of the ways a fleet monitoring platform reports an absence as a value, drawn from a 2026 assessment of a live PV and storage fleet. Twenty-plus verified defects sorted into six structural classes and answered with five design rules. Two of the defects were in our own platform.

The Measurement Layer · Reference One of Eight in the Series

energycomplianceinc.com **OCTOBER 2026**

Disclaimer & Legal Notice

This publication is the work product of Energy Compliance, Inc. and is provided for general information and educational purposes only. It does not constitute legal advice, regulatory guidance, engineering advice, or formal interpretation of any NERC Reliability Standard, FERC order, Department of Energy directive, RTO or ISO protocol, or state commission rule.

This publication may not be used or cited as authority in any audit, enforcement, regulatory, or legal proceeding.

On regulatory citations

The body of this reference carries no requirement text, no requirement number, no standard version and no effective date. That is deliberate. The subject is how a monitoring platform represents measured reality, a question that comes before, and stands apart from, which requirement is being satisfied. Standard families are named in the text only where the connection to a specific finding is material. For orientation, the Sources and Authorities section at the back lists the standards and NERC data requests behind those families as they stood on October 17, 2026. Nothing here is a statement of what any standard requires. A reader who needs operative language must get it from the current enforceable source for the relevant registered entity and audit period.

On the findings

Every defect described here was measured against a live production monitoring platform. The observations are anonymized. No client, site, site code, counterparty, vendor, product, asset tag, or person is identified, and figures are reported at a level of aggregation that does not permit re-identification. Several of the defects are in RenewOps, the platform Energy Compliance builds. Those are marked as ours and written up on the same terms as the rest.

On currency

A measurement is a statement about one system at one moment. The platforms described were under active remediation when it was written. A figure that was true when measured may not describe the same system today, and it does not describe any other system at all. Where a matter was open, unresolved, or unverified at the time of writing, it is identified as such. Energy Compliance, Inc. does not represent that any condition described here persists, or that any condition absent here does not exist elsewhere.

Non-affiliation

NERC, FERC, the United States Department of Energy, the Public Utility Commission of Texas, ERCOT, PJM, and the names of Regional Entities are referenced here for descriptive purposes only. Energy Compliance, Inc. is not affiliated with, sponsored by, or endorsed by the North American Electric Reliability Corporation, the Federal Energy Regulatory Commission, the Department of Energy, the Public Utility Commission of Texas, ERCOT, PJM Interconnection, L.L.C., or any NERC Regional Entity. RenewOps is a product of Energy Compliance, Inc.

Energy Compliance, Inc. assumes no responsibility or liability for errors, omissions, or outcomes resulting from the use of this publication. Compliance obligations depend on facts and circumstances unique to each registered entity, and the reader should consult qualified counsel and subject-matter experts when making compliance decisions.

Foreword

This professional reference is one of a series Energy Compliance, Inc. publishes for registered entities and the people who run their compliance programs.

I have spent more than thirty years on every side of the bulk electric system. I have operated control centers as a Reliability Coordinator, Transmission Operator, and Power System Operator. I have audited grid facilities and signed off on findings as a senior compliance auditor. I have worked enforcement matters from inside the regulator's process. For the last several years I have advised registered entities directly through the firm I founded.

Here is the thing about a screen. In a control room the screen is not a report on the system. For the eight hours you sit in front of it, the screen *is* the system. You act on it. You hand over on it. You write it into the log, the log becomes the record, and the record becomes the evidence. Everything downstream inherits whatever the screen got wrong, and nothing downstream is equipped to notice.

That is the subject of this series. Not whether a fleet is performing, but whether the number on the screen is a measurement of the fleet at all.

The series came out of an assessment. In 2026 we pointed a defect battery at a live production monitoring stack running a multi-gigawatt utility-scale PV and storage fleet and wrote up everything it found: twenty-plus verified defects, each with a source and a timestamp. What I did not expect was the shape of the result. The defects were not a

list of unrelated bugs. They were six recurring structures, and every one of them was a system reporting an absence as a value. A missing irradiance reading rendered as zero. A denominator that shrank as equipment failed, so a half-dead site scored a perfect hundred. An ingest timestamp wearing an event timestamp's label. A reading four hundred days old displayed as current. A revenue meter an order of magnitude above nameplate rendered live, because nothing in the system had a nameplate to check it against. A voltage band seeded from a plant's own history, indistinguishable on screen from a schedule issued by the entity of record.

None of that was a vendor defect. I want to be careful here, because the easy version of this series is a hit piece on monitoring platforms, and that version would be wrong. A monitoring platform displays what it is given. It was built to display. Nobody ever asked it to reconcile what it shows against what a contract requires, what a standard demands, or what physics permits. Nobody built that layer. The gap is not a failure of any product. It is a category that does not yet exist in most fleets.

Two of the defects in the original assessment were in our own platform. We wrote those up the same way and shipped the report with them in it, and they appear in these references marked as ours. I mention it because it is the only credential in this series that is hard to fake. Any vendor can show you a findings list from someone else's product. A vendor who has never found a fault in their own has not looked.

The body of these references carries no regulatory citations, and the Disclaimer at the front explains why. They are also not marketing material dressed up as content. I wrote them in the same voice I use in a control room or in front of a Regional Entity audit team: direct, tied to evidence, and honest about what a measurement can and cannot tell you.

This first reference is the taxonomy itself. It goes first because the six classes are the vocabulary the rest of the series uses, and because the taxonomy is the part that transfers. The specific defects belong to one fleet. The classes belong to everybody. I have not yet seen a monitoring stack of any size come back clean on all six.

Energy Compliance, Inc. exists because most of the consulting offered to registered entities is structured for billable hours rather than for outcomes. Every engagement is overseen by a senior practitioner, and we do not bring five people to a meeting that needs one. We automate the work that should be automated and apply senior judgment to the work that requires it. If that matches what you are looking for, the back of this reference tells you how to reach us.

If not, the reference still belongs to you. Take what is useful. And keep asking the only question that matters about a number on a screen: what would have to be true for this to be wrong, and would I be able to tell?

Rob Smith

Founder and Principal, Energy Compliance, Inc.

Bloomington, Minnesota

How to Use This Reference

This reference is one of eight in the Energy Compliance Monitoring Truth and Fleet Reconciliation Series. The series is organized layer by layer. Each reference stands on its own and assumes none of the others, at the cost of a little repeated framing where two layers meet.

EC-WP-1100 · The Measurement Layer: Missing Is Not Zero (this reference)
The six-class failure taxonomy and the five design rules that answer it. Every paper that follows takes one class apart on a live system.

EC-WP-1101 · The Alarm Layer: The Board That Deleted Its Own Alarms
Alarm management treated as an information system: load, priority, latching, and the monitor that stops. Seven failure modes, each with the arithmetic that exposes it.

EC-WP-1102 · The Operator Layer: What the Incoming Operator Cannot See
Situational awareness as a testable property of the record rather than a quality of a dashboard. Turnover, attribution and identity, measured five ways.

EC-WP-1103 · The Rules Layer: A Rules Engine That Measures Its Own Usefulness
The not-needed rate, and how a rules engine earns the right to keep a rule. It is the measurement that breaks the one-way growth of an alarm catalog.

EC-WP-1104 · The Procedure Layer: A Machine May Draft It
Switching order integrity when a machine prepares the draft. Five integrity requirements, each with the failure that motivates it, and the one field a drafter must never write.

EC-WP-1105 · The Contract Layer: Six Ways to Compute 99%
Availability constructs reconciled against executed agreements. Computability as a per-site position (ready, partial, blocked or undefined) with the blocker named.

EC-WP-1106 · The Evidence Layer: Displayed Is Not Citable
Provenance as the property that separates a number you can show from a number you can cite. The tautology trap, the seeded schedule, and the obligation nobody could mark done.

EC-WP-1107 · The Platform: The Reconciliation Layer
RenewOps itself: what the platform does, what it deliberately refuses to produce, and what it does not claim.

Where the findings come from

Every finding described here was observed in a live production monitoring stack during a 2026 engagement on a multi-gigawatt utility-scale PV and storage fleet. None of them is hypothetical. All of it is anonymized, and no operational record is reproduced. What travels is the *mechanism*: the shape of each failure, the reason it survives undetected, and the test that finds it. That is the part worth having, and it is the part that carries over to your fleet.

Where a figure appears (eleven days, 620 of 653, a third of sensors) it is the observed figure from the engagement, reproduced without the context that would identify its

source. Where we could not verify something, the text says so rather than rounding it into a claim.

Two of the twenty-plus findings were in RenewOps, the platform Energy Compliance builds. Both appear here, marked as ours.

How to read a figure in this series

Read every number with its source, and with its measurement date where the text gives one. This series does not make claims about systems it has not measured. Where a value could not be verified, it is marked as deferred with the specific question that would resolve it, rather than softened into an estimate.

What these tests cannot tell you

Every diagnostic in this series tests the *record*, not the plant. A clean result means the platform's picture of the fleet is internally consistent, sourced and current. It does not mean the equipment is healthy, the revenue is correct, or the program is compliant. The reverse also holds. A failed test is evidence that a number cannot be defended, not evidence that the asset underneath is in trouble. Several of the worst findings in the original assessment sat on top of plants that were running perfectly well. Confusing the two is the most common misreading of this work, and it produces false alarm and false comfort in equal measure.

For this reference specifically: the self-assessment protocol in Chapter 11 tells you whether your platform can tell an absence from a zero, a stale value from a current one, and a derived threshold from an issued one. It cannot tell you what any of that is costing you. Putting a number on the exposure takes the contract terms, the settlement history and the audit period, and a monitoring platform holds none of the three.

Reader's license

The detection tests in Chapters 3 through 8 are written to be run. They are in PostgreSQL syntax and every one of them parses. Adapt the table and column names to your schema and point them at your own historian or reporting database, read-only. If you find something, I would be glad to hear what it was, with nothing attached to the asking.

The findings in this reference are drawn from an assessment of one fleet. They are offered as a taxonomy to test your own fleet against, not as a statement about it.

Executive Summary

Renewable monitoring platforms were built to display data. They were not built to be cross-examined. That is a reasonable design goal right up until someone asks a question with money or a finding attached.

At that moment a display tool has nothing useful to say. It was never asked to hold the difference between a number it measured, a number it modeled, a number it inherited from a batch that stopped updating months ago, and a number that is simply missing data rendered in the shape of a zero.

In one assessment we found twenty-plus verified defects in a fully deployed, commercially supported monitoring stack that was working exactly as designed. Sorted by mechanism rather than by symptom, they fall into six classes. Each class is a different way of committing the same error: **the system reported an absence, an assumption or an artifact as though it were a measurement.**

Failure class	The error
Substitution	A missing value is written as a real one, almost always zero, and every calculation downstream consumes it as measurement.
Denominator drift	The denominator of a derived metric moves with the fault it exists to detect, so the metric improves as the plant degrades.
Clock collapse	Two different questions, *when did this happen* and *when did we last see it*, are answered with one timestamp field.
Staleness masquerade	An old value renders exactly like a current one because the display leaves out the part of the timestamp that would give it away.
Plausibility void	A physically impossible value is rendered live because nothing in the system holds a bound it could have been checked against.
Provenance erasure	A value that was inferred, seeded or assumed renders exactly like one issued by the entity of record, so neither can safely be cited.

Each class gets a chapter, and Chapters 3 through 8 share one structure: the mechanism, why the failure survives when more obvious ones do not, the observed case, a detection test you can run against your own data this week, and the design rule that prevents it. Read them in any order.

> **The eleven-day zero**
>
> Had those irradiance sensors gone properly dark (nulls, no rows, a dead link) somebody would have caught it inside an hour. The failure survived eleven days because it looked healthy.

The common thread

All six classes survive for the same structural reason: **monitoring systems are built to alarm on the presence of a bad condition, not on the absence of a good one.** A sensor that goes dark trips a staleness rule. A sensor that keeps reporting a wrong value trips nothing. From the system's point of view nothing is wrong. Data is arriving on cadence, in range, over a healthy link.

The correction is not a better alarm. It is a change in what the screen is willing to say. Five design rules, stated in Chapter 9, cover all six classes:

1. Missing renders as unknown, never zero.
2. Every tile carries its own denominator, on its face.
3. Every number shows its provenance and its age.
4. When the system cannot compute, it says so and names what would let it.
5. Refusals are recorded and visible.

None of these is hard to implement. All of them are hard to *hold*, because each one makes the product look worse in a demonstration. A dash in a tile reads as a broken feature. A zero reads as a working one. Most platforms choose to look like they are working.

Who should read which chapter

If you are	Start at
Control-room or O&M lead	Chapter 5 (clock collapse) and Chapter 6 (staleness). These two decide whether a quiet board is information or silence.
Asset owner or portfolio lead	Chapter 4 (denominator drift) and Chapter 10 (the audit consequence). These decide whether the availability number you report can be defended.
Compliance or regulatory lead	Chapter 8 (provenance erasure) and Chapter 10. These decide which of your signals are citable and which only look citable.
Engineering or data lead	Chapter 3 onward, in order, then Chapter 11 for the self-assessment protocol.

1 The Failure That Looks Like Health

For eleven consecutive days, a fleet we were assessing reported measured irradiance of exactly zero at every site.

Not missing. Zero.

The rows arrived on their normal cadence the entire time. Row counts were nominal. No staleness check fired, because nothing was stale. No communications alarm raised, because the links were healthy. No tile went gray, because nothing was absent for a monitoring system to notice. There was simply a wrong number, present and punctual, at every site, for eleven days.

Every performance-ratio and insolation-quality calculation downstream consumed those zeros as measurement. The result was eleven days of numbers that were precise, internally consistent, confidently rendered, and false.

Here is the part worth sitting with. Had those sensors failed properly (nulls, no rows, a dead link, a flatlined heartbeat) somebody would have caught it within the hour. Monitoring stacks are built to catch exactly that. **The failure survived because it looked like health.**

1.1 Why This Is Not a Vendor Defect

It is tempting to file this under poor software and move on. That would be an expensive mistake, because it leaves the reader believing a better vendor solves it.

The platform in question was fully deployed, commercially supported, and behaving exactly as specified. It was asked to display the value in a field. The field contained zero. It displayed zero. Nothing in its specification, and nothing in the specification of any comparable product we have assessed, requires it to hold an opinion about whether that zero was measured or manufactured.

That question, *is this number the kind of number it appears to be*, is not a feature the industry has built for. It is a category. Every finding in this document belongs to it.

1.2 The Distinction This Document Rests On

A monitoring platform answers *what is the value?* An evidence system answers *what kind of value is this, how old is it, what is it divided by, where did it come from, and would it survive being cited?* In my experience nearly every fleet has the first and almost none has the second. The gap between them is where every serious problem I have found in this work actually lives.

1.3 Three Systems That Have Never Spoken

Widen the frame and the gap has a shape. Every renewable fleet runs three systems, and they do not talk to each other.

The monitoring stack knows what the plant is doing, in five-minute intervals and enormous detail, and it is usually very good at that.

The contract knows what the plant is *required* to do: the availability construct, the guarantee, the exclusions, the measurement window, the termination trigger. It lives as a PDF in a document management system and nothing reads it.

The compliance program knows what the standards demand and what evidence would satisfy them. It lives in a spreadsheet, a shared drive, and one person's memory.

Every serious problem sits in the gaps between those three. A guarantee nobody can compute. Evidence that was never captured because nobody told the monitoring system it was evidence. An obligation that goes overdue because the field recording completion existed in the schema and nothing in the application could write to it.

None of that is a monitoring problem, and you cannot buy your way out of it with a better historian. It is a reconciliation problem.

1.4 What This Document Is For

It is not here to sell you anything, and it does not argue that your platform should be replaced. It probably should not be. It exists to give you a vocabulary for a class of defect that currently has no name in this industry, and a set of tests that will tell you within a day or two whether you have it.

In our experience everyone does. The useful question is *which of the six*, and how far downstream each one has already traveled.

2 A Taxonomy, and Why Symptom-Sorting Fails

Twenty-plus findings sorted by symptom produce a punch list. Sorted by mechanism they produce six classes, and the six classes tell you where to look next.

That matters more than it sounds. A punch list is finite and closes. A taxonomy generalizes. Once you can recognize denominator drift in a capability score, you will find it in a soiling ratio, a peer-comparison coverage figure and a fleet availability rollup without being told to look.

2.1 The Six Classes

Class	Mechanism	Tell
Substitution	An absent measurement is written into the record as a real value. Zero is the usual substitute because it is what an uninitialized numeric variable holds in common programming languages.	Exact zeros clustered by time or by site, with plausible neighboring columns.
Denominator drift	A derived metric's denominator is defined over the set of currently healthy assets, so it shrinks whenever an asset fails.	Metric improves while the underlying plant degrades. Best sites score worst.
Clock collapse	Event time and observation time share one column, so neither question can be answered reliably.	Many rows sharing an identical timestamp to the second.
Staleness masquerade	A display drops the coarse part of a timestamp, usually the date, so an old value cannot be told from a fresh one.	Time of day shown without a date. Age not displayed anywhere.
Plausibility void	No bound exists anywhere in the system against which a value could have been checked, so an impossible value passes through as live.	Values orders of magnitude off. Missing nameplate or rating reference rows.
Provenance erasure	Values of different epistemic status (measured, modeled, inferred, issued) are stored in one column and rendered identically.	No source column. Seeded defaults that look exactly like authoritative values.

2.2 How the Classes Interact

They are not independent. In practice they compound, and the compounding is what makes them expensive.

Substitution feeds denominator drift: a zero-filled sensor is often excluded from a peer set, which quietly shrinks the comparison population. Plausibility void feeds substitution: without a bound, a garbage value is stored rather than refused, and the next process to meet it may replace it with a default. Provenance erasure makes the other five harder to diagnose, because once a value's epistemic status is gone from the record, the only way to establish it is forensic.

Figure 1 · How one sensor failure propagates

A pyranometer degrades and begins reporting zero rather than failing open. **Substitution:** the zero enters the record as a measurement. **Plausibility void:** no bound exists that would flag daytime irradiance of zero at a site generating at half nameplate, so nothing refuses it. **Denominator drift:** the affected site drops out of a peer-referenced comparison and the reference set shrinks without any screen saying so. **Provenance erasure:** the resulting performance ratio is stored in the same column as ratios computed from healthy sensors, with no marker to tell them apart. A month later, a report is built on that column. Nobody involved did anything wrong.

2.3 What the Taxonomy Deliberately Excludes

This document is not about outages, sensor drift, calibration error, network reliability or the ordinary difficulty of running plants. Those problems are well understood, they have owners, and the tooling for them is mature.

Every class here is about **a system's representation of its own knowledge**: what it claims to know, how confidently, and whether that claim survives inspection. That is a narrower subject, and as far as I can tell nobody in this industry has been made responsible for it.

A reasonable objection, answered early

"Our engineers know which sensors are unreliable." Very likely true, and it does not help. The knowledge lives in people, not in the record. It leaves when they leave, the month-end report cannot see it, and nobody can produce it when someone asks for the basis of a reported figure eighteen months later. For every purpose that matters at audit or in a dispute, a defect that is known but unrecorded is undetected.

3 Substitution: The Absence Written as a Value

A measurement that does not exist is written into the record as a number that does. The number is almost always zero, for a reason that has nothing to do with engineering.

3.1 Mechanism

Zero is what an unassigned numeric variable holds in much of the software that moves plant data. In Go, a numeric variable declared without a value is 0. In Java, a numeric field or array element that nobody sets is 0. A database column is different: in PostgreSQL, a column with no declared default receives NULL. So when an acquisition path cannot get a value (the sensor is faulted, the tag is unmapped, the poll timed out, the transform threw) what lands in the row depends on the code in between. A pipeline that carries nulls end to end writes NULL. A pipeline that passes the reading through a variable that was initialized and never assigned writes 0. I see the second far more often than the first.

The consequence is asymmetric, and the asymmetry is the whole problem. NULL announces itself. Compare it to anything and the answer is unknown, not true or false. Aggregates skip it, so an average is taken only over readings that exist and a count of the column tells you how many there were. Zero does none of that. It participates. It sums, it averages, it divides, it plots. It is a full member of every calculation it enters, and it says nothing.

3.2 Why It Survives

Every detection mechanism a monitoring stack has is looking for the wrong thing. Staleness checks look at row timestamps, and the rows are arriving. Communications alarms watch link state, and the link is up. Range checks look for values outside physical bounds, and zero is inside every physical bound for a quantity that can legitimately be zero at night. Row-count monitors see a full complement.

There is a second reason. Zero is a *plausible* reading for most of the quantities involved. Irradiance really is zero at night. Power really is zero when a plant is off. Curtailed megawatts really are zero most of the time. The legitimate case camouflages the substitute.

> **Missing is not zero**
>
> Missing does not mean zero. It means we do not know. Those are different claims, and a system that cannot tell them apart cannot be cross-examined about anything.

3.3 Observed Case: Eleven Days of Fleet-Wide Zero-Fill

The measured-irradiance feed wrote exact zeros at every site in the fleet for eleven consecutive days. Rows arrived on the normal cadence throughout. No alarm caught it. A reconciliation did. On a large number of site-days, measured insolation was recorded as exactly 0.00 while metered generation at the same site on the same day was substantial, hundreds of megawatt-hours in the clearest cases.

That pair of facts is impossible. A site cannot generate at scale under zero irradiance. The pair had been sitting in the database, side by side, for eleven days, and nothing in the platform was built to notice that two of its own columns contradicted each other.

Two more instances of the same class turned up on the same fleet.

Where	What zero actually meant
Curtailment capability column	Thousands of rows carrying maximum possible output of 0 at a site that was generating. The real meaning was *no capability signal is available at this site.* Every curtailment loss calculation built on it returned zero loss, correctly, from a false premise.
Point-of-interconnection voltage	A smaller number of rows carrying 0 kV on an energized connection. The real meaning was *no voltage read was obtained.* Zero volts at an energized POI is not a marginal reading. It is a physical impossibility, and it was stored and displayed without comment.

3.4 Detection Test

Substitution is the easiest of the six to find, because it leaves an arithmetic fingerprint: an exact zero next to a value that makes it impossible. Run the contradiction, not the range check.

Test 1: irradiance zero while the plant generates

Any row returned is a substitution candidate, not a measurement.

```
SELECT site_id, reading_date, insolation_kwh_m2, generation_mwh
FROM daily_performance
WHERE insolation_kwh_m2 = 0 AND generation_mwh > 0
ORDER BY reading_date DESC;
```

Test 2: exact-zero density by day

A real quantity produces a smooth distribution near zero. Substitution produces a spike at zero and a gap just above it.

```
SELECT date_trunc('day', ts) AS d, COUNT(*) AS rows_total,
COUNT(*) FILTER (WHERE value = 0) AS exact_zero,
COUNT(*) FILTER (WHERE value > 0 AND value < 1) AS near_zero
FROM measurement_history
WHERE signal_role = 'irradiance_poa'
GROUP BY 1 ORDER BY 1 DESC;
```

Reading the second test. A genuine quantity that spends time near zero, such as irradiance at dawn and dusk, produces many near-zero values for every exact zero. A substituted column does the opposite: a large exact-zero count with almost nothing in the band just above it. That ratio is the tell, and it holds across signal types.

Run the same shape against every numeric column that feeds a reported figure. In our experience it takes an afternoon, and it has found at least one instance in every fleet we have looked at.

3.5 The Design Rule

Missing renders as unknown, never zero. Enforce it at three layers, because enforcing it at one is how it comes back.

Ingest. A value that could not be obtained is written as NULL, never defaulted. If the pipeline cannot express that, the pipeline is the defect.

Storage. Add a database constraint or trigger that refuses the known-impossible combination outright: a dead-zero state on a channel that cannot legitimately be zero in the current operating context. Refuse it at write time. Do not clean it up later.

Display. Null renders as a dash with a reason beneath it. Never a number, and never a blank cell. A blank cell reads as *fine*. A dash reads as *we do not have this*, which is the truth.

The display half of that rule gets abandoned first, and for aesthetic reasons rather than technical ones. A wall of dashes looks like a broken product to anyone who does not know what they are looking at. Holding the rule takes a decision, made once, that the screen will be honest even when honesty is unflattering.

4 Denominator Drift: The Metric That Rewards the Fault

A derived metric defines its denominator over the set of currently healthy assets. When an asset fails it leaves the numerator and the denominator at the same instant, and the metric improves.

4.1 Mechanism

Every derived metric is a ratio, and every ratio is an argument about what the right comparison population is. That argument gets made once, usually early, usually by one engineer under time pressure. Then nobody revisits it, because the number it produces looks reasonable, and a number that looks reasonable never gets audited.

Drift happens when the population is defined by asset state rather than by asset existence. *Expected output summed across generating inverters* drifts. *Expected output summed across installed inverters* does not. The two definitions differ by one word and behave in opposite ways under fault.

4.2 Why It Survives

The metric never looks broken. It stays inside its normal band, moves smoothly, and responds to real changes in plant behavior. It is not noisy, it does not spike, it does not go null. It is quietly answering a different question from the one everyone believes it answers.

You cannot see the failure from inside the metric. Staring at the score will not reveal it. You find it only by comparing the score against a fact the score does not contain, which nobody does, because the score exists precisely so that nobody has to.

4.3 Observed Case: The Capability Score at 100

Site capability was scored against expected output computed as the sum of possible power *over generating inverters*. These were the results.

Inverters generating	Output	Capability score	Flagged
40 of 83	46.3 MW	100 (Good)	None
50 of 75	108.6 MW	100 (Good)	None
316 of 329	73.5 MW	100 (Good)	None
360 of 375	224.7 MW	30 (Poor)	Yes

Read the first and last rows together. A site running **48% of its inverters** scored a flawless 100 with no flags. A site running **96% of its inverters**, a genuinely mild underperformance, scored 30 and was flagged for attention.

The metric built to surface lost capability got arithmetically better every time capability was lost, and it was steering the operations team's attention almost exactly backwards. Nobody had made a mistake in any ordinary sense. Somebody chose a denominator once, years earlier, and nobody ever cross-examined it.

Three more instances turned up on the same fleet.

- **Soiling ratio computed from surviving sensors.** A third of the fleet's soiling sensors were faulted. At two sites, three of five sensors were down and the published site ratio came from the two survivors, with nothing on the screen indicating a reduced population. The worst measured soiling loss fleet-wide was under three percent. I consider that implausibly low for the season and geography, and it is exactly what you would expect if the most-soiled sensors were the ones that had failed.
- **Peer-comparison coverage presented as fleet-wide.** An inverter health deviation metric covered 24% of installed inverters. Four large sites had zero units scored. The screen presented the result as a fleet view.
- **A fleet MW rollup with no denominator at all.** Fleet output was summed across whatever sites happened to be publishing, and the count of publishing sites appeared nowhere. The same tile read across a range of site counts within a single afternoon, and no viewer could have known.

4.4 Detection Test

Drift is found by adversarial recomputation. Compute the metric a second way, against a population that cannot move, and compare.

Recompute against the static population and diff

A large positive drift_pct means the denominator is shrinking with the fault. The LEFT JOIN matters: a site with no inverters generating must stay in the result, not drop out of it.

```
WITH installed AS (SELECT site_id, COUNT(*) AS n_installed FROM asset_registry WHERE asset_type = 'inverter' AND commissioned IS TRUE GROUP BY site_id),
reporting AS (SELECT site_id, COUNT(*) AS n_generating FROM asset_state_now WHERE asset_type = 'inverter' AND state = 'generating' GROUP BY site_id)
SELECT i.site_id, i.n_installed, COALESCE(r.n_generating, 0) AS n_generating,
ROUND(100.0 * COALESCE(r.n_generating, 0) / i.n_installed, 1) AS pct_online,
s.capability_score,
s.capability_score - ROUND(100.0 * COALESCE(r.n_generating, 0) / i.n_installed, 1) AS drift_pct
FROM installed i LEFT JOIN reporting r USING (site_id) JOIN capability_score_now s USING (site_id)
ORDER BY drift_pct DESC;
```

Reading it. Any site where capability_score materially exceeds pct_online has a denominator that is not the installed base. A site at 48% online scoring 100 returns a drift of +52, which is not a subtle signal.

The same shape works for any ratio. Identify the population the metric *claims* to cover, count it from the registry, count what actually contributed, and print both.

4.5 The Design Rule

Every tile carries its own denominator, on its face. Not in a tooltip, not in a footnote, not one click away. On the tile, in the same visual unit as the number.

The moment a qualifier sits one interaction away from the number it qualifies, somebody screenshots the number without it and pastes it into a board deck. That is not hypothetical. It is the normal life cycle of every figure your platform produces.

The second half of the rule matters just as much. **The denominator is the installed population, not the healthy one.** Where a metric genuinely must exclude assets, show the exclusion count beside it: *"of 83 installed, 43 excluded: not generating."*

5 Clock Collapse: Two Questions, One Timestamp

A control room needs two different times for every alarm. Most systems store one, and the one they store is the less useful of the two.

5.1 Mechanism

The two questions are not variations on a theme. They are independent, and an operator needs both at once.

Question	Answered by	Tells you
When did this happen?	**Onset**: the moment the condition first asserted, written once and never updated.	Whether anything new is occurring. Governs a latching board.
When did we last confirm it?	**Last observation**: the moment the ingest process most recently saw the condition still present, rewritten on every sweep.	Whether the feed is alive. Governs trust in the board itself.

Collapse happens when one column serves both. Because the ingest sweep rewrites its column on every pass, the meaning that survives is *observation*, and onset is destroyed. The system keeps the less informative of the two and loses the one operators actually reason with.

It gets worse. The sweep usually re-stamps every alarm it sees, *including the ones it is in the middle of closing out*. So the column is not even a reliable observation time for the alarm. It is a heartbeat of the process, painted onto every row it touched.

5.2 Why It Survives

The collapsed column looks correct, and it looks correct in the most reassuring way possible: it is always fresh. A screen sorted by it always appears current. Nothing looks stale, nothing looks stuck, and the display gives every impression of being closely tied to reality.

The failure only shows when two screens using different columns sit side by side. Even then it usually gets misdiagnosed, because the screen that is *right* is the one that looks wrong.

5.3 Observed Case: 620 Alarms, One Timestamp

An operations lead reported three times in one week that the alarm board appeared frozen while the sequence-of-events display was updating constantly. Here is what the live table showed.

Measurement	Value
Alarms standing on the board	653
Distinct *last-observation* values across them	32
Alarms sharing a single identical last-observation value	620
Distinct *onset* values across the same alarms	96
Minutes since the newest onset	39.4
Minutes since the newest confirmation	1.3

The board was correct the entire time. The last two rows were true together: the feed was alive, confirmed 1.3 minutes earlier, *and* nothing new had happened in 39 minutes. That pair is what a healthy plant looks like on a quiet afternoon.

The bucketed board stamped onset, so it showed ages of one hour, three hours, ten hours, eleven days, twenty-eight days. That is correct for a latching board. The sequence-of-events display stamped last observation, so twenty-five consecutive rows read the same second. Those were not twenty-five events. They were one sweep, painted onto twenty-five alarms that had been standing for hours. Neither screen said which clock it was using, so the operators concluded the board was dead and trusted the wrong one.

> **A calm board**
>
> A calm board should be information. Too often it is silence with a timestamp on it.

5.4 The Second-Order Failure: Counting Closed Items as Live

Collapse rarely stays in the display layer. Because the sweep keeps the column fresh on rows it is closing, any "asserted in the last 24 hours" filter built on that column counts resolved alarms as active. On the same fleet:

Priority	On the board (state-based)	24-hour filter	Of which already closed
Critical	32	32	0
High	230	391	170
Medium	320	478	155
Low	27	23,904	23,877
Info	52	13,901	13,849

Critical agreed exactly, and that detail is what let the defect live so long. Anyone spot-checking the numbers checked Critical, found agreement, and moved on. A critical alarm asserts once and stays asserted, so the sweep has nothing to re-stamp. Below Critical, alarm types that raise and auto-resolve constantly refreshed the column every cycle. The filter accumulated **just over 38,000 closed alarms reported as live**, and those fed map pins, operator queues, communications-health panels and protection registers.

5.5 Detection Test

Test 1: timestamp clustering

Concentration means collapse.

```
SELECT COUNT(*) AS alarms_standing, COUNT(DISTINCT last_seen_at) AS distinct_last_seen,
COUNT(DISTINCT first_seen_at) AS distinct_onset,
MAX(last_seen_at) AS newest_confirmation, MAX(first_seen_at) AS newest_onset
FROM alarms WHERE state = 'active';
```

Test 2: are closed items reaching the live filter?

```
SELECT priority,
COUNT(*) FILTER (WHERE state = 'active') AS on_board,
COUNT(*) FILTER (WHERE last_seen_at > now() - interval '24 hours') AS in_24h_filter,
COUNT(*) FILTER (WHERE last_seen_at > now() - interval '24 hours' AND state <> 'active') AS
closed_but_counted
FROM alarms GROUP BY priority ORDER BY closed_but_counted DESC;
```

Reading test 1. If distinct_last_seen is far smaller than the alarm count, that column is a process heartbeat, not an alarm attribute. If distinct_onset is close to the alarm count, onset is intact and the fix is a display and query change, not a data-loss problem. If both are small, onset was never captured and the repair goes deeper.

Reading test 2. Any nonzero closed_but_counted is a live defect. Inventory every consumer of that filter before you change it. Repointing the query without that inventory moves the problem instead of fixing it.

5.6 The Design Rule

One column, one question, and both labeled on the face of the screen.

Store onset and last observation separately. Never let a sweep write to onset. Label the columns in the interface as Onset (ET) and Last confirmed (ET), not "time," "updated" or "last seen," because each of those can mean either. Then publish the pair as a first-class fleet measurement: *feed confirmed 1.3 min ago · newest onset 39 min ago*. Once that pair is on the wall, a stopped feed can no longer hide behind a quiet room.

6 Staleness Masquerade: Old Rendered as Current

The data is correctly stored, correctly timestamped and correctly retrieved. The display drops the part of the timestamp that would tell you it is fourteen months old.

6.1 Mechanism

A display formats a timestamp as time of day. On a screen that refreshes every few seconds, showing seconds and leaving out the date is a defensible choice, until a value stops updating. From that moment the display looks exactly like a live one, and the older it gets the more confidently wrong it becomes.

This is the purest of the six classes, because the underlying data is entirely correct. Nothing was substituted, no denominator moved, no column was overwritten. The defect lives only in the rendering, which is why it survives code review and database audit alike.

6.2 Why It Survives

Every automated check misses it. The record is well formed. The timestamp is accurate. A query returns the right value with the right time. Only a human looking at a screen can see the failure, and by accident the failure is built to be invisible to a human looking at a screen.

Habituation helps it along. Operators see the same card in the same place every shift. A card that has read the same value for a year draws no attention. It becomes furniture.

6.3 Observed Case: The Wall of Stale State Cards

Voltage-regulation state cards across the fleet displayed a time of day and no date. The cards read as current. Checked against the underlying state table:

Card displayed	Actual timestamp	Age at review
8:53:42 AM	June 27, previous year	424 days
7:32:31 AM	February 13	193 days
10:10:31 PM	May 13	103 days
9:31:57 AM	June 4	82 days
2:49:42 PM	July 22	34 days
4:59:51 AM	July 27	29 days
4:44:11 PM	August 12	13 days

Card displayed	Actual timestamp	Age at review
10:32:37 AM	August 14	11 days

Eight cards on one wall spanned fourteen months of staleness, and every one of them presented as the current state of a regulating asset. The eleven-day card and the 424-day card were visually identical.

The fix is one line of formatting code. It earns a chapter because the defect had persisted for more than a year in a monitored, supported, actively used system, and it would have persisted indefinitely. Nothing in the system could report it, and nothing in the operators' routine could notice it.

Two more instances turned up on the same fleet.

- **A soiling station printing a clean result at 27 days old.** The card read *measured 1.000, 0.0% loss*, a perfectly plausible figure, from a sensor that had last reported nearly a month earlier. Age appeared nowhere.
- **A performance feed seven days stale with no indication.** Actual generation had stopped landing a week before review. Every downstream ratio kept rendering against the last values received, and the screen offered no age.

6.4 Detection Test

You cannot find staleness masquerade in the data, because the data is fine. You find it by computing age for every displayed quantity and ranking it. The system almost certainly has no such report, which is why the defect exists.

> **Age of the newest row per feed, per site**
>
> *Sort descending and read the top.*
>
> SELECT feed_name, site_id, MAX(observed_at) AS newest_row,
> now() - MAX(observed_at) AS age,
> EXTRACT(epoch FROM (now() - MAX(observed_at)))/86400 AS age_days
> FROM feed_observations GROUP BY feed_name, site_id
> HAVING now() - MAX(observed_at) > interval '2 hours'
> ORDER BY age DESC;

Scope it per feed, not per site. The most common gap we find is a staleness rule defined at site level. As long as *any* feed from a site is arriving, the site counts as healthy, and an individual channel can be dead for a year inside a site that reports as live. Site-level staleness is close to useless. Channel-level staleness is the control that matters.

Then do the manual half. It takes twenty minutes and is worth more than the query. Walk the screens and list every timestamp displayed anywhere in the product. For each one, write down

whether it shows a date and whether it shows an age. Every entry with neither is a live instance of this class.

6.5 The Design Rule

Every displayed value carries its age, and every timestamp carries its date.

Show the age, not only the timestamp. *"As of 10:32 · 11 days ago"* does work that *"10:32:37 AM"* cannot, because it does not ask the reader to do arithmetic against a date they may not have noticed.

Past a defined threshold the value stops rendering as a value. It becomes *"last known: 11 days"* in a visibly different treatment. A last-known value is often still useful operationally. What is never acceptable is a last-known value wearing the costume of a live one.

7 Plausibility Void: Impossible Rendered Live

A value arrives on time, over a healthy link, in the right column, and it is physically impossible. Nothing refuses it, because nothing in the system holds a bound to check it against.

7.1 Mechanism

Monitoring stacks validate three things well: that data arrived, that the link is up, and that the value parses as the right type. They rarely validate a fourth: that the value is physically possible for *this asset*.

That fourth check needs something the other three do not: a per-asset reference such as voltage class, nameplate rating and expected operating band. Generic range checks cannot supply it. A 200 MW reading is unremarkable on a 300 MW site and impossible on a 20 MW one. Without the per-asset reference there is no bound, and without a bound there is nothing to refuse.

The void is usually a data-model gap rather than a logic gap. The check is missing because the reference table it needs is incomplete, and the reference table is incomplete because nothing ever failed loudly enough to force somebody to fill it in.

7.2 Why It Survives

Implausible values are often *excluded* from the screens where a human would catch them and *included* in the aggregates where nobody can. A corrupt reading four orders of magnitude high does not sit on a trend line where an engineer would spot it. It sits inside a fleet sum, one contributor among twenty, and its influence is invisible.

This class also comes with a diffusion of responsibility. Site engineers look at their own site and see nothing wrong, because the corrupt tag is not one they watch. Fleet operators look at the rollup and see a plausible total. The person who would notice is whoever compares the two, and in most organizations that person does not exist.

7.3 Observed Case: One Meter, 39% of the Fleet

A revenue meter on a storage site published values orders of magnitude above the site's physical capability. The platform rendered them LIVE, with link state reported as OK.

Observation	Value
Voltage displayed on a medium-voltage connection	808.28 kV
Power displayed on a 100 MW asset	1,180.83 MW

Observation	Value
24-hour voltage range	0.000–9,262.42 kV
24-hour power range	0.066–13,991.94 MW
Historical rows beyond plausible bounds, all one site	4,629
Fleet rollup at time of review	3,015.2 MW across 21 sites
Contribution from this single meter	1,180.8 MW (39.2%)

Two-fifths of the reported fleet output came from one corrupt instrument, and the fleet tile showed a number that looked entirely ordinary.

The root cause was a data-model gap, not a logic failure: **the site had no row in the nameplate table.** There was no rating anywhere in the system against which 1,180 MW could have been recognized as impossible. One trust view did flag the site. The view that governed the operational display did not, and it reported the readings as trustworthy.

One thing we checked and rejected, because it matters

The obvious hypothesis is a register scaling artifact, a 16-bit rollover producing a clean multiple. We tested it. Only 61 of 298 recent samples landed on a 65,536 boundary, which is far too few. **Do not apply a correction factor to values like these.** A correction factor applied to a non-systematic corruption manufactures plausible wrong numbers, and those are worse than obvious wrong numbers. Fix the tag mapping at source, and until it is fixed, refuse the values.

7.4 A Quieter Instance: Mixed Units in One Column

Power factor across the fleet, in a single snapshot, held values expressed as percent (99.98, 99.96), as per-unit (0.03, 0.19), and as signed percent (−99.87, −99.44). All of them sat in the same column and rendered identically. Every fleet average computed over that column was arithmetic on three incompatible units. No individual value was implausible under *some* convention, which is exactly why nothing caught it.

7.5 Detection Test

Plausibility testing needs a reference. Start by finding out where you do not have one. That gap is usually the finding.

Step 1: which sites can we not check at all?

Every row returned is an asset with no plausibility reference.

```
SELECT s.site_id, s.site_name
FROM registry_sites s LEFT JOIN site_nameplate n ON n.site_id = s.site_id
WHERE n.site_id IS NULL;
```

Step 2: for sites we can check, find the impossible

1.15 times nameplate is a deliberately generous bound.

```
SELECT h.site_id, h.observed_at, h.mw, n.nameplate_mw,
ROUND(h.mw / NULLIF(n.nameplate_mw, 0), 2) AS times_nameplate
FROM telemetry_history h JOIN site_nameplate n USING (site_id)
WHERE h.mw > n.nameplate_mw * 1.15
ORDER BY times_nameplate DESC;
```

Step 3: unit consistency within a column

More than one bucket populated means mixed units.

```
SELECT site_id,
COUNT(*) FILTER (WHERE ABS(power_factor) <= 1.0) AS looks_per_unit,
COUNT(*) FILTER (WHERE ABS(power_factor) > 1.0) AS looks_percent,
COUNT(*) FILTER (WHERE power_factor < 0) AS signed_values
FROM electrical_history
WHERE observed_at > now() - interval '7 days'
GROUP BY site_id;
```

Step 1 is the important one, and it is the step most teams skip. The question is not only "which values are impossible" but "for how much of my fleet is the question unanswerable?" A site with no reference row is not passing the plausibility check. It is exempt from it, permanently and silently.

7.6 The Design Rule

Every signal role carries a plausibility rule, and every refusal is recorded.

Define the bound where the meaning lives: on the signal role, derived from voltage class and nameplate, not in the display layer. Then hold three lines.

Refuse at ingest; do not clean up later. An implausible value is rejected before storage, with an audit row naming the value, the bound and the reason. A corrupt value that reaches storage will eventually reach a report.

A missing reference is a finding, not an exemption. An asset with no nameplate row sits on a gap list until somebody supplies one. It does not quietly opt out of validation.

Never silently correct. If a systematic scaling error is confirmed (genuinely confirmed, not assumed from one clean example) fix the mapping at source. A correction factor applied to an unconfirmed pattern manufactures plausible wrong numbers, and plausible wrong numbers are the most expensive thing any system can produce.

8 Provenance Erasure: Inferred Rendered as Issued

Two values sit in the same column. One was issued by the entity of record. The other was inferred from the plant's own history because no issued value was available. On screen they are identical. Only one of them can be cited, and nobody can tell which.

8.1 Mechanism

Values in an operational database differ in kind, not only in size. A setpoint may be *measured*, *modeled*, *inherited* from a prior configuration, *inferred* from the plant's own behavior, or *issued* by an external authority. Those are five epistemic statuses with five different evidentiary weights.

A typical schema stores all five in one numeric column with no companion field recording which is which. Once that happens the distinction is gone from the record, and inspection cannot recover it. A seeded band and an issued schedule are the same data type holding similar numbers.

This is the most consequential of the six classes, because it is the one that turns a data problem into an evidence problem. The other five make numbers wrong. This one makes correct numbers **uncitable**. That is a different and worse failure. Recomputing does not fix it, and you discover it at the moment someone asks you to produce the basis for a figure.

> **Inferred is not issued**
>
> A setpoint band the platform inferred from your own plant history is not a schedule your operating entity issued. On most screens the two look identical. Only one is citable, and if the screen does not say which, you cannot safely cite either.

8.2 Why It Survives

Seeding is a reasonable engineering decision that becomes a problem later, and in a different department. When a platform is stood up across a fleet, some sites will not have an issued schedule on day one. Seeding a plausible band from the historical median is the right call. It makes the screen usable and it harms nothing.

The defect is not the seeding. The defect is that nobody marked the seeded value as seeded. Eighteen months later, when a compliance lead needs to show operation within an issued schedule, there is no way to separate the sites where that can be shown from the sites where it cannot.

8.3 Observed Case One: Five of Twenty Schedules Citable

Twenty voltage schedules were present in the platform. On review, **five** had been issued by the operating entity of record. The other fifteen had been seeded from each site's own historical median.

To the platform's credit, the seeded values had been labeled *indicative only*. That is the design rule this chapter argues for, implemented correctly. The finding is what the label revealed: three-quarters of the fleet's schedules were not evidence. On an unlabeled platform that fact could only have been found by reconstructing the record document by document.

One site made the operational stakes concrete: a regulation loop setpoint of 164.5 kV against a metered reading of 34.7 kV and a schedule of 34.1 kV. A setpoint of a different order from the quantity it regulates is a configuration error. It had gone unremarked because no screen compared the three.

8.4 Observed Case Two: The Evidence Source That Did Not Exist

At every site carrying automatic voltage regulation, the service-status field was NULL across all rows, and the matching display read "--" on every card in the fleet.

This is provenance erasure in its most complete form. The question is not whether the displayed value is trustworthy. There is no value. The voltage-control obligation that turns on that status runs on a clock: a Generator Operator has 30 minutes to tell its Transmission Operator about an AVR status change, unless the status is restored within those 30 minutes. Nothing anywhere in the platform stood behind that clock. The right move is to state that plainly in the compliance narrative and name the source that would supply it, rather than discover it during an audit.

8.5 Observed Case Three: Sixty-Five Intervals That Could Not Compute

A recurring-obligation calendar showed **65 of 65** intervals reading *cannot compute: last performed not recorded*. The completion fields existed in the schema. Nothing in the application could write to them.

So the calendar could display obligations but could not record their performance. A compliance calendar that cannot record completion is not a calendar. It is a list. And a next-due date computed from a blank last-performed date is not conservative. It is arbitrary.

8.6 Detection Test

> **Test 1: does a provenance column exist at all?**
>
> *If not, that is the finding.*
>
> SELECT column_name, data_type FROM information_schema.columns
>
> WHERE table_name = 'voltage_schedules' ORDER BY ordinal_position;

Test 2: where a source column exists, how much is authoritative?

```
SELECT COALESCE(source_type, 'UNRECORDED') AS provenance, COUNT(*) AS n,
ROUND(100.0 * COUNT(*) / SUM(COUNT(*)) OVER (), 1) AS pct
FROM voltage_schedules GROUP BY 1 ORDER BY n DESC;
```

Test 3: compliance-relevant fields that are entirely empty

A 100% null rate is not sparse data. It is an absent evidence source.

```
SELECT COUNT(*) AS rows_total, COUNT(service_status) AS rows_populated,
COUNT(*) - COUNT(service_status) AS rows_null
FROM regulation_state;
```

Test 4: recurring obligations that cannot compute a next-due date

```
SELECT obligation_id, standard_family, interval_months, last_performed_at,
CASE WHEN last_performed_at IS NULL THEN 'cannot compute' ELSE 'computable' END AS status
FROM compliance_obligations WHERE recurrence = 'interval'
ORDER BY status, standard_family;
```

Test 1 is usually decisive. If the schema has no provenance column, the answer is not that provenance is poor. Provenance was never captured, and every value in the table has unknown epistemic status until someone reconstructs it from outside documents.

8.7 The Design Rule

Every compliance-relevant value carries its provenance on its face, and the unavailable case is stated rather than blanked.

Store status with the value. A source field, an authority and a date obtained, mandatory on write. A value without provenance is refused, not accepted with a null source.

Render status with the value. Issued values carry an issued marker. Inferred values carry *indicative only, not evidence*. The reader never has to ask which one they are looking at.

Distinguish the three empties. *No evidence source exists* is not the same as *not yet obtained*, and neither is the same as a blank cell. A blank reads as fine. Say on the screen which of the three it is.

Completion is a record, not a state. Capture who performed it, when, and how many days early or late. Require a written reason to reopen it and keep that reason permanently. Refuse an unattributed entry and refuse a future date. Anything less is a checkbox, and a checkbox is not evidence.

9 Five Rules That Close All Six Classes

None of these is hard to implement. All of them are hard to hold, because each one makes the product look worse in a demonstration.

Rule	Closes
1 · Missing renders as unknown, never zero	Substitution
2 · Every tile carries its own denominator	Denominator drift
3 · Every number shows its provenance and its age	Staleness masquerade; provenance erasure
4 · When it cannot compute, it says so and says why	Clock collapse; plausibility void
5 · Refusals are recorded and visible	All six. It is the control that keeps the other four honest.

9.1 Rule One: Missing Renders as Unknown, Never Zero

Enforce it at ingest (null, never defaulted), at storage (a constraint that refuses the known-impossible), and at display (a dash with a reason, never a number and never a blank). Enforcing it at one layer is how it returns. Each layer catches what the others miss.

The display half goes first, and for aesthetic rather than technical reasons. A dash reads as a broken feature to anyone who does not know what they are looking at. Holding the rule means deciding once, explicitly, that the screen will be honest even when honesty is unflattering. Then you defend that decision the first time somebody senior asks why a tile is empty.

9.2 Rule Two: Every Tile Carries Its Own Denominator

On its face, in the same visual unit as the number. Not in a tooltip, not in a footnote, not one interaction away.

This is not pedantry. The normal life cycle of any figure your platform produces is that somebody screenshots it and pastes it into a deck. Whatever falls outside the crop does not survive. A denominator one click away will be absent from every decision the number ever informs.

Define the population statically. *Of 83 installed* is a denominator. *Of those currently generating* is a moving target, and sooner or later it will flatter exactly the situation you built the metric to catch.

9.3 Rule Three: Every Number Shows Its Provenance and Its Age

Four provenance states cover nearly everything: **measured** (an instrument read it), **modeled** (computed from other quantities), **inferred** (derived from the asset's own history because nothing better was available), and **issued** (provided by an external authority of record). Only measured and issued values are evidence without qualification.

Age travels with provenance because it modifies it. A measured value four seconds old and a measured value four hundred days old are not the same claim. A display that renders them identically is making a false statement about its own knowledge.

9.4 Rule Four: When It Cannot Compute, It Says So and Says Why

The rule has three parts, and the third is what makes it useful rather than merely honest.

1. **Say that you cannot.** Not a zero, not a blank, not an unexplained dash.
2. **Say why.** *Coefficients not populated. Reference asset has no rating. Feed last confirmed 11 days ago.*
3. **Name what would resolve it.** The specific input, document or decision that turns "cannot" into "can." Without it, a refusal is an excuse. With it, a refusal is a work order.

A good auditor holds their own findings to the same discipline. An auditor who writes *"unable to determine"* without stating what would have determined it has produced nothing anyone can act on.

9.5 Rule Five: Refusals Are Recorded and Visible

This is the rule nobody ships, and the one to examine first if you are evaluating a platform, ours included.

Every value the system declined to compute or display goes into a ledger any user can open, with its reason. A panel prints *"0 of 1 evaluable"* instead of a comfortable zero. A database trigger rejects a dead-zero state instead of storing it. A count states its exclusions instead of absorbing them.

Rule five keeps the other four from decaying. Rules one through four are invisible when they work (a correctly rendered dash looks like nothing at all) so they erode quietly under delivery pressure. A visible refusal ledger makes the discipline observable, and that makes its absence noticeable.

9.6 The Economic Argument, Stated Once

Each rule trades a small, immediate, visible cost against a large, deferred, invisible one. The small cost is an emptier screen and a longer conversation in a demonstration. The large one is a report built on a number nobody can defend. The person building the product pays the first. Whoever is holding the file when someone asks a hard question pays the second, and that is usually a different person, usually much later. That asymmetry is why the rules are hard to hold, and it is the only reason they need writing down.

10 What Each Class Costs When It Meets a Data Request

Every class in this document is survivable while nobody asks. The cost arrives on somebody else's schedule (a dispute, an audit, a diligence request, a lender's question) and by then the record is what it is.

Class	What happens when it is examined
Substitution	The reported figure cannot be reproduced from source data, because the source holds values the reporter did not know were manufactured. The reconstruction is forensic and the outcome is a restatement.
Denominator drift	The counterparty computes the same metric from the installed base and gets a materially different number. Both parties are computing correctly from the same data, which makes the dispute long rather than short.
Clock collapse	Event chronology cannot be established. In an event analysis or a sequence-of-events reconstruction, that is the difference between a defensible narrative and a set of assertions.
Staleness masquerade	An operating condition is asserted for a period during which no observation existed. The assertion is not so much wrong as unsupported, and unsupported is the harder position to defend.
Plausibility void	A single corrupt instrument is shown to have contributed materially to reported fleet figures. Every figure derived from that rollup is then in question, including the ones that were fine.
Provenance erasure	The most expensive. Evidence cannot be produced because its status was never recorded, and unlike the other five, recomputing does not fix it. The underlying operation may have been perfect.

10.1 The Contractual Dimension

On the assessed fleet, sixteen sites guaranteed availability, and the number was computed **six structurally different ways**. These were not six formulas for one idea. They were six

constructs with different denominators, exclusions and measurement windows. Every solar site guaranteed 99%. None meant the same thing by it.

On several sites the contractual equation depended on coefficients the agreement left *to be mutually agreed upon*. Nobody ever populated them. Neither party could have computed the guarantee both had signed, and neither had tried. Two further exhibits referenced attachments that were missing from the executed copy.

Against that, one site terminated on any miss below its guarantee with no cure period, and that site's availability construct was one of the ones that could not be computed.

> **The asymmetry worth internalizing**
>
> The party that finds a computability gap first controls how it is characterized. Found internally, it is a remediation project on your own timeline. Found by a counterparty during a dispute, it is a credibility problem attached to a number you have been reporting for years. The gap is identical. The position is not.

10.2 The Compliance Dimension

Provenance erasure is where a monitoring problem becomes an evidence problem, and the transition deserves a precise statement.

A monitoring platform is not an evidence system. It is not required to be one, and most are not sold as one. But operational data is routinely relied on as the raw material for compliance demonstration: voltage regulation, protection system maintenance intervals, model validation inputs, generating availability reporting. The assumption underneath that reliance is that the values the platform holds are what they appear to be.

These are the three failures I would look for first, in any fleet.

- **Setpoints and schedules with no recorded source.** If it cannot be shown to have been issued by the entity of record, it is not evidence of operating to an issued schedule, whatever the plant actually did.
- **Status fields that are null across every row.** A uniformly empty compliance-relevant field is not sparse data. It is an evidence source that does not exist, and it is far better identified now, with a plan, than during an audit without one.
- **Recurring intervals with no completion record.** An interval whose last-performed date was never captured cannot compute a next-due date, and a program that cannot compute next-due cannot show that it performed on schedule.

Consistent with the notice at the front of this document, the body names no requirement, version or effective date here. Which requirements attach to which of these gaps depends on registered function, asset configuration and audit period, and that analysis belongs in a deliverable prepared against the current enforceable source. The Sources and Authorities section lists where to start.

11 A Self-Assessment Protocol

Two days of work, no vendor required. Run the stages in order. Each one narrows what the next has to look at.

11.1 Stage 1: The Screen Walk (Two Hours, No Database Access)

Print or screenshot every operational screen. For each displayed number, answer four questions in a spreadsheet. Do not fix anything yet.

1. What is this number divided by, and does the screen say so?
2. How old is this number, and does the screen say so?
3. If the source stopped feeding, what would this show?
4. Is this measured, modeled, inferred or issued, and does the screen say?

Question three is the highest-yield question in this document. Any tile whose answer is *"the same as it shows now"* is a confirmed instance of substitution or staleness masquerade, established without touching a database.

11.2 Stage 2: The Four Queries (Half a Day)

Query	Finds	From
Exact-zero density	Substitution	Chapter 3, test 2
Feed age ranking, per channel	Staleness masquerade	Chapter 6
Timestamp clustering on alarms	Clock collapse	Chapter 5, test 1
Missing reference rows	Plausibility void	Chapter 7, step 1

Run them against production, read-only. All four are cheap. None needs a schema change or a maintenance window.

11.3 Stage 3: The Contract Read (One Day, and the One People Skip)

Take one executed agreement. Find the availability construct. Then answer these in writing.

1. What exactly is the denominator, in the contract's own words?
2. Which exclusions apply, and can you identify each one in your data?
3. Is every coefficient the equation needs actually populated somewhere?
4. Does the exhibit reference an attachment, and is that attachment in the executed copy you hold?
5. Does the number you report to this counterparty use this construct, or your internal one?

Question five is where the money is. In our experience the two are often different, nobody has ever quantified the difference, and nobody in the organization knows there are two numbers.

11.4 Stage 4: Write the Register

One row per finding: what was observed, where, when, which class, and the specific question that would resolve it. Mark each finding **confirmed** or **deferred**. Never "probably," never "appears to." A deferred finding with a precise question is more useful than a confident-sounding finding that is actually a guess. The discipline of choosing between those two words will improve the register more than any other single practice.

> **What good looks like at the end of two days**
>
> A register of ten to twenty-five findings, each with a source and a timestamp, sorted into the six classes, with a named owner and a resolving question for each. That document is worth more than any dashboard, and it did not require buying anything.

11.5 What to Fix First

Order matters, and the intuitive order is wrong. Fix in this sequence.

1. **Anything feeding an external report.** Contractual availability, regulatory submissions, lender reporting. These have counterparties and deadlines.
2. **Anything feeding an operational decision.** Capability scores, dispatch, work-order prioritization. These misdirect attention every day they persist.
3. **Display-only defects.** Real, cheap to fix, and able to wait a week. A stale card nobody acts on costs less than a wrong number in a monthly report.

The instinct is to start with the display defects because they are quick. Resist it. The reported figures are where the exposure lives.

12 What RenewOps Does About This

The five rules in Chapter 9 are not aspirations. Each one is a behavior you can look at. This chapter shows what they looked like in RenewOps on a live production instance between August 24 and 28, 2026, including the rule RenewOps was failing. The figures describe that instance on those dates, not a specification, a roadmap, or any other fleet.

RenewOps is the reconciliation layer Energy Compliance builds. It reads from the monitoring stack a fleet already runs, checks what it finds against the plant's ratings, the executed agreements and the documents of record, and declines to publish what it cannot defend.

12.1 Rule One: Missing Renders as Unknown, Never Zero

Three populations of false zeros were converted to unknown on August 25, 2026: rows in the curtailment history carrying a maximum possible power of zero, the measured-insolation column feeding the performance-ratio calculation, and a telemetry voltage column. Every one of those rows had been a real absence wearing a number.

The curtailment flag is a harder case, and the one worth studying. Where no determination was made, the flag is set to **unknown**, not to false. A false all-clear during a real curtailment is worse than no reading, because an operator acts on it and a settlement analyst later reconciles against it.

The one that reads as a bug and is not. An infinite insulation-resistance reading at nine sites now renders blank. Infinity there means *no fault path to measure*, and on screen it had been rendering as an extremely healthy value.

Deferred, and stated as such. As of the August 2026 measurement, the eleven days of irradiance already written as zero were still written as zero. Stopping new false zeros and remediating historical rows are two different decisions, and the second one was open. Those rows fed everything downstream of them.

12.2 Rule Two: Every Tile Carries Its Own Denominator

Site availability publishes as an **unweighted mean, named as unweighted**, with the per-asset minimum and the count below threshold beside it. No site rollup was invented, and the reason deserves a plain statement. Whether an agreement averages by asset count or weights by nameplate is a term of that agreement. It is not a display choice, and a platform that picks one silently has substituted its own construct for the contract's.

Every DC-loss row carries a coverage note stating how many paired hours it rests on. A day whose inverter coverage swings more than twenty percent **labels itself UNEVEN COVERAGE**, and a comparison bucket requires comparable coverage on both sides before it will compare at all. Alarm parent rows carry the full child count and the priority mix.

The largest open item against this rule. An authoritative fleet definition exists, with a separate flag for each thing a tile might really be counting. As of August 26, 2026, **no tile cited it**, and screens still stated fleet counts that differed from one another. The definition is the hard part, and it was done. Putting the qualifier on the face of each tile is the easy part, and it was not.

12.3 Rule Three: Every Number Shows Its Provenance and Its Age

Every state stamp carries the full date and an explicit age in days. One feed read **796 days**. Before the fix the card printed a time of day and nothing else, which concealed feeds up to 424 days stale, and it fooled the reviewer who was in the middle of documenting the defect. That is why the fix routes every stamp in the product through one formatter instead of repairing the cards one at a time.

Values resolve to the **latest non-null reading per signal, each carrying its own as-of time**, rather than to the newest row. The difference is not cosmetic. Reading the newest row made one site publish "0 of 352 units reporting" when the units were reporting on a slower cadence than the row that arrived last.

Two battery sites carry an explicit **ROLLUP LIVE, UNITS SILENT** verdict. The site-level number updated every fifteen minutes while every one of its units (276 at one site, 42 at the other) had stopped answering days earlier. The rollup is not wrong at its own source. It is simply not evidence of what a reader would assume, and the verdict says so on the row.

Where the plausibility gate refuses a reading, **the raw value is kept beside the blank.** The record of a bad reading is the evidence that the gate is needed.

12.4 Rule Four: When It Cannot Compute, It Says Why

Where	What it prints instead of a number
Curtailment tiles	"--" with the reason on the face: *not computable: no directed-curtailment history*; *needs a real curtailed MW and a real price*.
Twenty historical availability rows	Blank, with the reason on the row. The denominator was wrong, so the online count was taken over the same wrong population and the percentage cannot be corrected. **It is not estimated.**
Sixty-five recurring intervals	*Cannot compute: last performed not recorded*, rather than a manufactured next-due date or a silent slide into overdue.
Four sites in a DC comparison	Excluded, each with its own stated reason: partial plant measured against a whole-plant meter; coverage swinging across the window; no DC rating held.
A day's event history	The fetcher halves its window on hitting a response cap, recursively, and **returns an error rather than writing a partial day that looks whole.**

12.5 Rule Five: Refusals Are Recorded and Visible

The plausibility gate refused **367 samples at one site in twenty-four hours** on the day it landed, with zero false positives anywhere else in the fleet. The worst raw values it caught that day were around 13,700 MW and 9,280 kV, on a plant whose rating makes both impossible.

It runs three independent tests. The first is a per-site power ceiling set from the plant's own rating. The second is an absolute bulk-system voltage ceiling plus a per-site band derived from the voltage schedule of record. The third is a shape test that rejects a twenty-four-hour span wider than one and a half times its own center.

The refusal that refuses itself. Where a site's own voltage schedule is suspect, RenewOps **declines to use that setpoint as a banding input** and lists the site in a coverage view instead. Six sites carried no assertable band at the time of measurement, and they were stated as open per-site questions rather than defaulted to something plausible. A plausibility check that bands against a number it does not trust is not a check.

The tracker wind channel is gated to a physically possible band, with **every rejected sample named in an audit view.** All 162 controllers at one site reported 758 m/s, about Mach 2.2 at sea level, and wind stow is decided off that signal.

Two alarm-catalog promotions are **deliberately held, with the reason held as data** rather than as a comment in somebody's ticket. A priority override the classifier cannot read is **refused on write**, rather than accepted and stored where it would silently do nothing. A publish refuses unless four separate assertions pass: child counts reconcile, duplicate keys read zero, every dependent view rebuilds with its permissions re-asserted, and both layers complete their refresh.

12.6 The Rule the Product Did Not Pass

A gate exists that should bar any site reporting under ninety-five percent of its inverters from receiving a GOOD capability grade. The code is deployed and confirmed present in the running bundle. **It does not fire.** On August 25, 2026, one site showed GOOD 100 with 50 of 75 inverters, 66.7 percent. The cause was unknown on that date, and the next step was instrumenting the two inputs in the browser.

The same score was still computed per browser session, per operator, and not persisted. An unauditable score is not evidence of anything. Two operators looking at the same site can hold two different numbers with no way to reconcile them.

Both belong here rather than in a release note. A chapter on design rules that left out the rule the product is failing would be an instance of the behavior this reference spends eleven chapters arguing against.

> **The trade this product makes**
>
> A platform that refuses is more useful than one that guesses, and it is harder to sell. That is why the refusal ledger is a feature rather than an apology.

Glossary

Term	Meaning in this paper
Clock collapse	Storing event time and observation time in one column, so neither can be answered reliably. See Chapter 5.
Computability state	Whether a contractual figure can be computed from the executed agreement: *ready*, *partial*, *blocked* or *undefined*, with the specific blocker named.
Denominator drift	A derived metric whose denominator is defined over healthy assets, so it shrinks with the fault. See Chapter 4.
Indicative only	A label for a value derived from the asset's own history rather than issued by an authority. Useful operationally; not evidence.
Inferred	Provenance state: derived from the asset's own behavior because no authoritative value was available.
Issued	Provenance state: provided by an external entity of record, with source and date captured. Citable.
Last observation	The moment an ingest process most recently confirmed a condition still present. Rewritten every sweep. Indicates feed liveness, not event recency.
Measured	Provenance state: read by an instrument.
Modeled	Provenance state: computed from other quantities.
Nameplate	The rated capacity of an asset as held in the platform's reference data. Without it, no per-asset plausibility bound exists. See Chapter 7.
Not-needed rate	How often operators judge a rule's output not worth surfacing. The measure that lets an alarm set shrink rather than only grow. See EC-WP-1103.
Onset	The moment a condition first asserted. Written once, never updated. The correct basis for a latching board.
Peer referencing	Comparing like assets against each other rather than against a modeled expectation. Degrades honestly when a shared reference fails.
Plausibility rule	A per-signal bound derived from voltage class and nameplate, against which an arriving value can be refused.
POI	Point of interconnection: where the plant connects to the transmission system.
Provenance	The epistemic status of a value (measured, modeled, inferred or issued) recorded alongside it.

Term	Meaning in this paper
Refusal ledger	A visible record of every value a system declined to compute or display, with the reason.
Staleness masquerade	A display that omits date or age, rendering an old value exactly like a current one. See Chapter 6.
Substitution	Writing an absent measurement into the record as a real value, usually zero. See Chapter 3.
Verified / Deferred	The only two permitted states for an assertion. Verified means confirmed against a primary source with provenance captured. Deferred means not confirmed, stated as not confirmed, with the question that would resolve it.

How Energy Compliance Can Help

Everything in this reference can be run by your own team in two days. If you would rather have a second set of eyes on it, or you have already found something and need to know what it means for a contract or a compliance program, this is where we come in.

Monitoring Truth Assessment

We point our defect battery at your live monitoring stack and hand you the findings, each with a source, a timestamp and the question that resolves it, whether or not you buy anything afterward. Where we cannot verify something, it comes back marked deferred with the specific question, not softened into a maybe. Nobody buys a reconciliation layer before they know they need one, and the only honest way to find out is to look.

Fleet Exposure Read

Your availability constructs reconciled against your executed agreements. You get a per-site computability position (ready, partial, blocked or undefined) with the specific blocker named for every site that cannot compute.

Evidence Provenance Review and gap assessment

Your compliance-relevant signals traced to source and sorted into citable and not, with the seeded, the derived and the self-referential separated from the issued. Where a gap touches a reliability obligation, we tie it to the current enforceable standard for your registered functions and give you a mitigation path that will stand up at audit.

WattOps Compliance

WattOps Compliance is EC's NERC compliance evidence platform. Once you know which of your signals are evidence, it is where that evidence lives. We will run one standard at one facility through WattOps at no cost so you can see it on your own data. CIP standards are excluded from that offer. Details are at wattopscompliance.com.

RenewOps

RenewOps is EC's renewable operations monitoring and analytics platform, and it includes a Data Center module. Chapter 12 shows how it handled the five rules on a live fleet, including the one it was failing.

Pick the one number on your screens you would least like to defend, and call me about it.

Website: energycomplianceinc.com · Email: rsmith@complianceforenergy.com · Phone: 763.438.4427

About the Author

Rob Smith is the Founder and Principal of Energy Compliance, Inc., with more than thirty years in the industry on every side of the system. He began in control-room operations as a Reliability Coordinator, Transmission Operator, and Power System Operator across RTO/ISO and utility control centers. On the regulator side, he was a senior compliance auditor and subject matter expert for NERC Reliability Standards. He audited grid facilities, evaluated mitigation adequacy, participated in risk-based oversight of utility mitigation activities, and supported the development of violation notifications and settlements in FERC-directed enforcement actions. He conducted a regulatory audit in the Sultanate of Oman against its Sector Law and Grid Code.

Education: MBA, Energy Management, University of Mary; MSL, Compliance Law, Fordham Law University; BAS, Energy Management, Bismarck State College; BAS, Mechanical Engineering, Metallurgy minor, University of Florida; BAS, Criminology, St. Petersburg College (State Law Enforcement Certification); AAS, Power Plant Technology, Bismarck State College; AAS, Electrical Transmission System Technology, Bismarck State College.

Certifications: NERC Lead Auditor training; previously NERC-certified at the Reliability Coordinator level; 1st Class A Boiler Engineer License, Minnesota.

Auditing teaches one habit that never leaves: before you believe a number, ask what it would look like if it were wrong.

The views expressed in this publication do not represent the views of NERC, FERC, or any Regional Entity.

About Energy Compliance, Inc.

Energy Compliance, Inc. is an independent regulatory compliance and advisory firm serving the energy sector, with a focus on NERC, FERC and RTO/ISO compliance and a particular concentration in the ERCOT and Texas PUCT markets. The firm helps registered entities and prospective registrants through the full compliance lifecycle: registration, program design, evidence development, RSAW production, mitigation, and audit defense.

The approach is research-analyst-first. Every conclusion is tied to an authoritative source, every narrative is backed by evidence, and every deliverable is built to survive regulator scrutiny. Every engagement is overseen by a senior practitioner with regulator-side experience. We do not staff for billable hours. We staff for outcomes. Where automation can replace manual work, we build the automation. Where senior judgment is required, the senior is in the room.

That discipline extends to a portfolio of compliance technology, including WattOps Compliance and RenewOps, the reconciliation layer described throughout this series.

To discuss how any of this applies to a specific fleet, or to request other references from the Energy Compliance library, visit energycomplianceinc.com, email rsmith@complianceforenergy.com, or call 763.438.4427. Energy Compliance, Inc. is based in Bloomington, Minnesota.

Sources and Authorities

The body of this reference cites no requirement. The sources below are the current primary authorities behind the standard families and reporting obligations it names, and the technical references for statements about software behavior. All were checked on October 17, 2026.

NERC Reliability Standards (voltage regulation, protection maintenance, model validation)

- NERC, Reliability Standard VAR-002-4.1, Generator Operation for Maintaining Network Voltage Schedules; status "Mandatory Subject to Enforcement," effective September 26, 2017 (Docket No. RD17-7-000). Requirement R3 requires a Generator Operator to notify its Transmission Operator of an AVR, power system stabilizer, or alternative voltage controlling device status change within 30 minutes unless the status is restored within that time. https://www.nerc.com/standards/reliability-standards/var/var-002-4.1
- NERC, Reliability Standard VAR-002-4.1 (standard text). https://www.nerc.com/globalassets/standards/reliability-standards/var/var-002-4.1.pdf
- NERC, Reliability Standard PRC-005-6, Protection System, Automatic Reclosing, and Sudden Pressure Relaying Maintenance; status "Mandatory Subject to Enforcement," effective January 1, 2016 (Docket No. RD16-2-000); applies to Generator Owners for BES generating Facilities. https://www.nerc.com/standards/reliability-standards/prc/prc-005-6
- NERC, Reliability Standard MOD-026-2, Verification and Validation of Dynamic Models and Data; FERC order issued February 19, 2026 (Docket No. RD26-3-000); effective April 1, 2026, with phased compliance dates of April 1, 2027 for Requirements R1 and R7 and April 1, 2029 for Requirements R2 through R6. The February 19, 2026 order also approved MOD-032-2, IRO-010-6, TOP-003-8 and MOD-033-3 (Docket Nos. RD26-1-000, RD26-2-000 and RD26-3-000). https://www.nerc.com/standards/reliability-standards/mod/mod-026-2
- NERC, Project 2020-06, MOD-026-2 Implementation Plan (retires MOD-026-1 and MOD-027-1 immediately before MOD-026-2 takes effect). https://www.nerc.com/globalassets/standards/projects/2020-06/final-ballot---mod-026/2020-06-mod-026-2-implementation-plan-clean_102025.pdf

NERC generating availability reporting

- NERC, Rules of Procedure, Section 1600, Requests for Data or Information. https://www.nerc.com/globalassets/who-we-are/rules-of-procedure/nerc-rop.pdf

- NERC, 2026 GADS Solar Data Reporting Instructions (reporting required for registered Generator Owners operating solar plants of 20 MW or more of Plant Total Installed Capacity with commercial operation on or after January 1, 2010; data collected under a Section 1600 Data Request). https://www.nerc.com/globalassets/programs/rapa/gads/solar/2026-gads-solar-data-reporting-instructions.pdf

Software behavior and physical constants

- PostgreSQL Global Development Group, PostgreSQL Documentation, Section 5.2, Default Values (a column with no declared default receives NULL). https://www.postgresql.org/docs/current/ddl-default.html
- PostgreSQL Documentation, Comparison Functions and Operators (comparisons with NULL yield null; IS NULL predicates). https://www.postgresql.org/docs/current/functions-comparison.html
- PostgreSQL Documentation, Aggregate Functions (aggregates compute over non-null input values; count is the exception for empty input). https://www.postgresql.org/docs/current/functions-aggregate.html
- The Go Programming Language Specification, "The zero value" (numeric variables without explicit initialization are set to 0). https://go.dev/ref/spec#The_zero_value
- Oracle, The Java Language Specification, Java SE 21, Section 4.12.5, Initial Values of Variables. https://docs.oracle.com/javase/specs/jls/se21/html/jls-4.html#jls-4.12.5
- NASA Glenn Research Center, Beginner's Guide to Aeronautics, Speed of Sound (about 761 mph at sea level on a standard day; Mach number defined as the ratio of speed to the speed of sound). https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/speed-of-sound-interactive/

© Energy Compliance, Inc. · EC-WP-1100 · Status as of October 17, 2026