MONITORING TRUTH & FLEET RECONCILIATION SERIES · EC-WP-1101

The Board That Deleted Its Own Alarms

Alarm management as an information system: load, priority, latching, and the monitor that stops.

Seven failure modes in a live alarm subsystem, each with the arithmetic that exposes it. From 7,839 rows on the board to 1,175. From 17 unrendered Critical alarms to 32 rendered. From a suppression rate reported as 14.2% to a measured 80.9%. An analysis of what an alarm system is actually doing to the attention of the people who watch it.

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 the absence of regulatory citations

The body of this series carries no regulatory requirement text, requirement number, standard version, or effective date, and it cites no order, protocol section, or rule. The omission is deliberate. The subject of the series is how a monitoring platform represents measured reality. That question comes before, and stands apart from, which requirement is being satisfied. Regulatory standard families are named only where the connection to a specific finding is material, and they are named as families in the text. Where one is named, the Sources and Authorities page at the back identifies the version in effect on the review date so a reader can go to the source. Nothing here should be read as a statement of what any standard requires. A reader who needs operative language must obtain 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 described 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 are under active remediation. 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. The external references in this edition were checked against their sources on October 17, 2026.

Non-affiliation

NERC, FERC, the United States Department of Energy, the Public Utility Commission of Texas, ERCOT, PJM, the International Society of Automation (ISA), the American National Standards Institute (ANSI), 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., ISA, ANSI, 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, and I want to be careful here. 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. It was never asked to reconcile what it displays 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 that is deliberate; the reasoning is on the previous page. They are also not marketing material dressed as content. I wrote them the way I talk in a control room or in front of a Regional Entity audit team: direct, grounded in evidence, and honest about what a measurement can and cannot tell you.

Alarm management is where I expect the most disagreement. Every operations group I have worked with already believes its board is under control, and most of them have a number that says so. This reference is about how that number is computed. A suppression rate reported at 14.2 percent and measured at 80.9 percent is not a rounding difference. It is two different questions wearing one label.

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. 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 apply it well. 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 small amount of repeated framing where two layers meet.

EC-WP-1100 · The Measurement Layer: Missing Is Not Zero
The six-class failure taxonomy and the five design rules that answer it. This is the foundational reference for the series; 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 (this reference)
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. This is the measurement that breaks the monotonic-growth ratchet in 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 is treated 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. It is the capability reference for the layer the other seven show a need for.

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 is a test of the *record*, not of the plant. A clean result means the platform's representation 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 underlying asset is in trouble. Several of the worst findings in the original assessment sat on top of plants that were running perfectly well. Conflating the two is the most common misreading of this work, and it produces both false alarm and false comfort.

Specific to this reference: the arithmetic here tells you what your alarm system is doing to your operators' attention. It does not tell you whether the alarms are the right alarms. A board that is well loaded, correctly prioritized, and honestly rated can still be watching the wrong things. That is a rationalization question. It needs plant engineers in the room, not a query.

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.

About This Document

Every finding here was measured against a live production platform on the date stated. None of it is recalled, modeled or illustrative.

All of it is anonymized. No client, site, site code, vendor, product, asset tag or person is identified. Where a figure appears it is the observed figure, reproduced without the context that would identify its source. Some of the defects described are in RenewOps, the platform Energy Compliance builds; those are marked **ours** and written up the same way as the rest.

Regulatory standard families are named only where the connection to a finding is material. No regulatory requirement text, requirement number, standard version or effective date appears in the body of this document. Verify any regulatory position against the current enforceable source for the relevant registered entity and audit period before relying on it.

One external benchmark is still deferred. Chapter 3 computes alarm load against a per-operator hourly threshold of 60. That threshold is the value configured in the platform under assessment, not a published standard. The current edition of the principal US alarm-management standard is ANSI/ISA-18.2-2016, *Management of Alarm Systems for the Process Industries*. Trade articles that describe ISA-18.2 report much lower average rates: about 6 alarms per hour as very likely to be acceptable and about 12 as the maximum manageable. One article states those figures per operator console; the other captions them as frequencies for an alarm system. We have not read the figures in the standard's own text, which ISA sells. **Deferred: the benchmark must be confirmed against the published ANSI/ISA-18.2-2016 text before any exceedance figure in Chapter 3 is cited externally.** The arithmetic in that chapter is correct against the configured threshold and is presented as such.

Executive Summary

An alarm board is an information system, and it can lie

The most expensive alarm defect we have found did not produce a flood, a false positive, or a missed condition. It produced a board that looked calm, correct and well managed while it silently deleted every alarm the platform raised itself.

Over one overnight assessment and the remediation that followed, an alarm subsystem went from 7,839 rows on the board to 1,175, from 17 unrendered Critical alarms to 32 rendered ones, and from a suppression rate reported as 14.2% to a measured 80.9%. Every one of those movements corrected something the board had been asserting incorrectly, in some cases for more than a year. Chapters 2 through 7 take the subsystem apart, one or two failure modes to a chapter, each with the arithmetic that exposes it and the design rule that prevents it.

Measure	Before	After
Rows on the operator board	7,839	1,175
Of those, state events carrying no condition	6,965 (89%)	48 (held deliberately)
Critical alarms present	17	32
Critical alarms rendered to the operator	0	32
Platform-generated alarms surviving one ingest cycle	0 of 843	16 standing
Suppression rate as published	14.2%	80.9% (measured)

- **Chapter 2: the deletion defect.** A sweep closed every alarm absent from an upstream array, which is every alarm the platform computes itself. 843 raised, 800 destroyed, none seen.
- **Chapter 3: load and suppression.** Computing load honestly, and why a rate measured over the wrong window is worse than none.
- **Chapter 4: priority resolution.** A classifier read a browser-posted blob ahead of a 249-code reviewed catalog. The resolution order that fixes it.
- **Chapter 5: states are not conditions.** Removing 89% of a board without losing an alarm.
- **Chapter 6: key stability and latching.** Why an unstable key makes acknowledgment meaningless, and how alarms that never clear stop re-latching.
- **Chapter 7: the monitor that stops.** A stopped evaluator looks like a quiet plant.

1 What an alarm is supposed to be

An alarm is a claim about the plant that obliges a person to do something. Anything that does not meet that definition is a different kind of object wearing an alarm's clothes.

The definition has three parts, and each maps to a failure class that occupies a chapter of this document.

Part	Means	Fails as
A claim about the plant	The condition is real and currently true. Not a status change, not a heartbeat, not the poller's opinion.	State events on the board (Ch. 5)
That obliges action	There is something a person should do. If the answer is "nothing," the object is information, and it belongs in a log.	Flood and suppression (Ch. 3)
By a person	A specific person, with a response, whose action is recorded against their identity and can be reconstructed.	Attribution failure (Ch. 6, and EC-WP-1102)

The four objects that get confused with alarms

- **State events.** "Sunrise: Daylight Hours." A true statement about the world that requires nothing of anybody. On one assessed board this single event had latched **2,092 times**.
- **Conditions.** A real state that persists and clears on its own. One example is an inverter work-state transition with a 97.4% self-clear rate and a median duration of 0.56 minutes. Real, and not an alarm.
- **Derived observations.** "Live production 14% below forecast so far today." A computed judgment, useful, and not a plant condition.
- **Diagnostics.** Instrument health. Important, and addressed to a different person on a different timescale.

Each of the four is legitimate. Putting any of them on an operator's alarm board produces the flood. The flood makes the board unreadable, and an unreadable board is how a trip-coil supervision failure sits at High priority for twenty-six days.

The measurement that settles the argument

Whether an object belongs on the board is not a matter of opinion. It is measurable from the object's own history, with two numbers.

Calculation 1 · Board eligibility

```
-- Self-clear rate and median duration decide the class.
SELECT event_name,
COUNT(*) AS occurrences,
COUNT(*) FILTER (WHERE cleared_without_action) AS self_cleared,
ROUND(100.0 * COUNT(*) FILTER (WHERE cleared_without_action) / NULLIF(COUNT(*),0), 1) AS self_clear_pct,
PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY EXTRACT(epoch FROM (cleared_at - raised_at))/60) AS median_minutes,
COUNT(*) FILTER (WHERE state = 'active') AS active_now
FROM alarm_history
WHERE raised_at > now() - interval '90 days'
GROUP BY event_name
ORDER BY occurrences DESC;
```

Reading it. A high self-clear rate with a short median duration is transient chatter. It resolves before a human could act, so a human acting on it is impossible by construction. A low self-clear rate with a long median duration is a genuine standing condition. On one assessed catalog, a trip-coil supervision code returned a self-clear rate of **0.0%** with a **median duration of 26 days**. That is the signature of a real defect nobody was closing, and it sat at High priority beneath a flood of transients.

The rule this yields. An object belongs on the operator board only if a person can act on it before it resolves itself, and only if the action is one the operator is actually authorized to take. Everything else belongs in a log, a report, or a different person's queue. Moving it there is not suppression. It is filing.

2 The board that deleted its own alarms

Every alarm the platform raised itself was destroyed within thirty seconds of being raised, for months, and stamped as though the plant had returned to normal.

The defect

An ingest sweep reconciled the local alarm store against the array of active keys the upstream monitoring platform handed it on each cycle. Simplified, the predicate was:

The sweep, as it stood

```
UPDATE alarms
SET active_upstream = false,
state = CASE WHEN state = 'acked' THEN 'archived' ELSE 'auto_resolved' END,
archived_at = now(),
archive_reason = 'cleared_at_source'
WHERE state IN ('active_unacked','acked')
AND active_upstream = true
AND last_seen_at < now();
```

Read the predicate carefully. It does not say *this alarm cleared at source*. It says *this alarm is not in the array I was just handed*. Those are different claims, and for one category of alarm the difference is catastrophic: **anything the platform computes itself is, by construction, never in that array.**

What it cost

Alarm generator	Rows ever raised	On the board	Closed as "cleared at source"
Voltage band excursion	575	0	548
Voltage schedule deviation	262	0	246
Regulation state change	5	0	5
Storage state-of-charge low	1	0	1
Total	843	0	800

Not one of the 843 had ever survived a single thirty-second ingest cycle. The voltage-band and voltage-schedule alarms matter most here, because they are the compliance-relevant ones, in the standard family governing operation to a voltage schedule. **794 of them were raised and destroyed, each stamped with a reason asserting the plant had returned to normal. No operator ever saw one.**

> **Ours**
>
> This defect was in RenewOps. It is published here for the same reason it was published in the assessment report. The mechanism is general and will exist in any system that reconciles a local store against an upstream list. A vendor who describes only other people's defects is writing a sales document, not describing a method.

How it was found, and why it nearly was not

Not by an alarm about alarms. By an assertion. A storage state-of-charge monitor raised a low-charge alarm at 7.81% against a documented latch threshold of 10%. Thirty seconds later the row read auto_resolved. The monitor was correct; the sweep had eaten its output.

The general point reaches well beyond alarms: **the only reason this was findable is that something wrote down what it expected to happen and then checked.** Without the documented threshold, the disappearance would have read as normal behavior.

The fix, and the two second-order problems it exposed

The repair itself is one clause. Derive an origin on the alarm row (a key carrying the upstream event-start timestamp is upstream; anything else was minted locally) and scope the sweep with AND origin = 'upstream'. Two consequences surfaced immediately, and both are instructive.

Consequence one: unstable keys

Three of the four local generators embedded epoch milliseconds in the alarm key, so every evaluation minted a fresh row. That was harmless while the sweep deleted them all. The moment they survived, it became unbounded growth. Worse, it made acknowledgment meaningless, because the next evaluation would create a row nobody had acknowledged.

A key is not an identifier if it changes when the underlying condition does not. Chapter 6 covers this properly.

Consequence two: the sweep had been hiding dead monitors

The sweep was quietly cleaning up after evaluators that had stopped. Remove it and a stopped monitor's last alarm stands forever, asserting a condition nobody is checking. In one observed instance a storage monitor stopped at 07:50. At 11:29 the board still showed a site at "6.7%, at or below 10%" while the plant actually read 19.86%.

The answer is not to reinstate the deletion. It is a heartbeat, plus a per-alarm freshness label that marks an unconfirmed alarm as unconfirmed rather than clearing it. That is Chapter 7.

The design rule

A reconciliation predicate must state the claim it is actually making.

If the predicate is "not present in the upstream list," the reason code written must be *not present upstream*, never *cleared at source*. The two are different facts, and the query that produced the second does not support it.

Then: **scope every sweep to the population it has authority over.** A process reconciling against an external source has authority over rows that originated from that source and none over anything else. Origin is not metadata; it is the boundary of the process's mandate.

And: **never let a cleanup process double as a health check.** If removing a sweep exposes stale rows from dead evaluators, the sweep was concealing an availability defect, not preventing one.

3 Load, suppression, and the arithmetic of an operator's burden

Suppression rate is one of the most quoted numbers in alarm management and one of the easiest to compute wrongly. A suppression rate measured over the wrong window is worse than no number at all, because people believe it.

The observed error

A view published a fleet suppression rate of **14.2%**, with 2,992 alarms per hour reaching the board. Both figures were wrong, and wrong in the same way. The view counted suppression-tagged rows across a 24-hour window, but **the trigger that writes the suppression tag had only existed for part of that window.** Every hour before the trigger existed added rows to the denominator and none to the numerator.

Measured over a window lying entirely after tagging began:

Measure	As published	Measured correctly
Suppression rate	14.2%	80.9%
Alarms per hour reaching the board	2,992	1,131

The corrected figures are better news and a worse finding. Better, because suppression was working far more effectively than reported. Worse, because 1,131 per hour is still an unmanageable load. And the published number had been understating the one control that was actually helping. That is exactly the kind of error that gets a working control switched off.

Calculation 2 · Suppression rate, computed honestly

```
-- The window must start at the LATER of (now - 24h) and the moment
-- the tagging mechanism began writing. Publish the window with the rate.
WITH w AS ( SELECT GREATEST(now() - interval '24 hours', (SELECT tagging_began_at FROM alarm_pipeline_meta)) AS win_start )
SELECT (SELECT win_start FROM w) AS window_start,
EXTRACT(epoch FROM (now() - (SELECT win_start FROM w)))/3600 AS window_hours,
COUNT(*) AS arrivals,
COUNT(*) FILTER (WHERE archive_reason = 'state_event') AS suppressed,
ROUND(100.0 * COUNT(*) FILTER (WHERE archive_reason = 'state_event') /
NULLIF(COUNT(*),0), 1) AS suppressed_pct,
ROUND(COUNT(*) FILTER (WHERE archive_reason IS NULL) / NULLIF(EXTRACT(epoch FROM (now()-(SELECT win_start FROM w)))/3600,0), 0) AS to_board_per_hour
FROM alarms, w
WHERE created_at >= w.win_start;
```

The rule embedded in that query is the one to take away: **publish the window alongside the rate.** A percentage with no stated measurement window is not a measurement. It is a rumor. The corrected view emits window_hours and a plain-language window_note on every row, so a reader can see that a rate computed over four hours is not comparable to one computed over twenty-four.

Load per operator: the number that actually matters

Fleet alarms per hour is a system property. Alarms per operator per hour is an ergonomic one, and it is the only version that predicts whether the board gets read.

Calculation 3 · Per-operator load and headroom

```
load_per_operator_hr = alarms_to_board_per_hour / operators_on_shift
exceedance_factor = load_per_operator_hr / configured_threshold_per_hr
-- Observed, after suppression was measured correctly:
-- alarms_to_board_per_hour = 1,131
-- configured_threshold_per_hr = 60
-- 1 operator: 1,131 / 1 = 1,131/hr -> 18.9x threshold
-- 3 operators: 1,131 / 3 = 377/hr -> 6.3x threshold
```

Deferred, and it matters

The 60-per-hour figure above is the threshold configured in the assessed platform. It is *not* a published standard value. Trade articles describing ANSI/ISA-18.2 report far lower sustained rates, about 6 per hour as very likely acceptable and about 12 as the maximum manageable, stated per operator console in one article and for the alarm system in the other. Before any exceedance factor from this chapter is quoted externally, the benchmark must be read from the current ANSI/ISA-18.2-2016 text and the arithmetic redone against it. The exceedance factors above are correct against the configured threshold and are presented on that basis only.

Two things follow regardless of which benchmark is eventually used. First, tripling the operators divides the load by three and does not change its order of magnitude. Staffing is not the lever. Second, the entire value of the suppression control shows up in one subtraction: without it, the load on the board would have been more than five times higher.

Where the load actually comes from

Board composition after remediation, from one assessed fleet:

Priority	Rows	Share	Comment
Critical	32	2.7%	The population the board exists for
High	193	16.4%	Actionable, triaged
Medium	837	71.2%	One transient work-state code is 51% of this bucket alone
Low	35	3.0%	
Info	78	6.6%	
Total	1,175	100%	

Seventy-one percent of the board sat at Medium, and half of that bucket was a single transient code with a 97.4% self-clear rate and a median duration of 0.56 minutes. By the test in Chapter 1, a condition that resolves itself in thirty-four seconds cannot be acted on by a person and so is not an alarm. But it was correctly classified as a real condition rather than a state event, so it could not simply be swept.

That is a genuine design tension, and I would rather name it than paper over it: **a real condition that no human can act on in time still does not belong on an operator's board.** The correct destination is a condition register with trending. Not the alarm queue, and not the bin.

The daylight problem: a worked example of a gate that must exist

An alarm catalog promoted an inverter under-production code to High on the rationale *"inverter producing nothing while the sun is up."* The rationale is sound. The implementation had no way to know whether the sun was up.

Measured shortly after 21:00 local, in full darkness, **582 of these were standing across two sites**, with 637 fleet-wide. Promoting them to High would have put six hundred night-time alarms above the thirty-two Critical alarms the remediation had just made visible.

The promotion was **held at Medium with the reason recorded as data** in a catalog-holds view. It was neither applied nor silently dropped. That is the right handling of a decision that cannot yet be made correctly: the analysis is preserved, the blocker is named, and the unsafe change does not ship.

Calculation 4 · The daylight gate

```
-- Promote a sun-dependent code only when irradiance supports the premise.
-- 50 W/m2 is a deliberately conservative floor: comfortably above night,
-- comfortably below any level at which real generation is expected.
SELECT a.alarm_id, a.site_id, a.event_name, i.poa_wm2,
CASE WHEN i.poa_wm2 IS NULL THEN 'hold - no irradiance reference'
WHEN i.poa_wm2 < 50 THEN 'hold - dark'
ELSE 'promote' END AS gate_decision
FROM alarms a
LEFT JOIN LATERAL ( SELECT poa_wm2 FROM site_irradiance_history WHERE site_id = a.site_id
AND observed_at <= a.raised_at ORDER BY observed_at DESC LIMIT 1 ) i ON true
WHERE a.event_name IN (SELECT event_name FROM alarm_catalog WHERE requires_daylight IS
TRUE);
```

Note the three-way outcome. **No irradiance reference is a hold, not a promote.** A gate that defaults to "promote" when it cannot evaluate its own premise is not a gate. And a site with no irradiance reference is a gap on a list, not an exemption. It is the same principle as the missing nameplate row in EC-WP-1100.

4 Priority resolution, and who is allowed to decide

A priority classifier read a JSON blob posted by the browser, then a regex, then returned "Low." A 249-code reviewed catalog existed and was never consulted. A governed override table, with a named approver and a written reason, was written by two functions and read by none.

The defect, stated as an authority problem

This is not primarily a code defect. It is a question of *who decides* the priority of an alarm, and the resolution order answered it wrongly.

The system had three sources of priority, and they carry very different authority:

- **An approved override.** A deliberate decision by a named approver with a written reason, recorded. The highest authority available.
- **The reviewed catalog.** 249 codes worked through by engineers, with rationale. Considered judgment, applied consistently.
- **A blob posted by the client.** Whatever the browser happened to send.

The resolution order in force was, in effect, the reverse of that list.

The correction

Resolve in the order authority actually runs

```
-- First match wins; the client blob is the fallback, never the override.
CREATE FUNCTION fn_alarm_priority_of(p_code text, p_event text, p_payload jsonb)
RETURNS text LANGUAGE sql STABLE AS $$
SELECT COALESCE(
(SELECT priority FROM alarm_priority_override WHERE event_name = p_event AND approved_at IS NOT NULL ORDER BY approved_at DESC LIMIT 1), -- 1. approved override
(SELECT priority FROM alarm_catalog WHERE event_name = p_event), -- 2. catalog by event
(SELECT priority FROM alarm_catalog WHERE code = p_code), -- 3. catalog by code
alarm_classify_row(p_payload) -- 4. legacy fallback
);
$$;
```

What surfaced when authority was restored

Critical alarms moved from 17 to 32. The fifteen that appeared had been sitting at lower priorities:

New Criticals	What they were	Catalog evidence
10	Trip-coil supervision failure across three sites	Self-clear 0.0%, median duration 26 days
4	Inverters stopped by emergency stop at one site	None recorded
1	Inverter major fault, unit offline	None recorded

Trip-coil supervision is the circuit that confirms a breaker can still be tripped. Ten of those had been standing at High, beneath a Medium bucket of transient chatter, with a measured median duration of twenty-six days and a self-clear rate of zero. That combination has exactly one interpretation.

Two catalog decisions deliberately held

A key too coarse to act on

One catalog entry was keyed on an event name that covered three materially different conditions: a feeder breaker open, a generic breaker open, and *a high-voltage breaker open with the main power transformer isolated*. Applying the catalog's suggested priority would have moved the third off the Critical row along with the first two.

Held, with the reason recorded as data. The correct fix is a finer key, not a coarser decision. Shipping the coarse decision while waiting for the finer key would have removed a transformer isolation from the operator's Critical view.

The daylight promotion

Covered in Chapter 3. Held at Medium pending the gate.

The design rule

Every derived attribute needs a documented resolution order. The order must run from highest authority to lowest, with the first match winning.

Two constraints keep it honest. **A client-supplied value can never outrank a reviewed one.** Anything the browser posts is an input, not a decision. And **an override function must refuse a scope its own classifier cannot read.** The corrected implementation rejects a site-scoped override outright rather than storing a record that will never be consulted, because a governed decision that silently does nothing is worse than a refused one.

Finally: **a held decision is recorded with its reason, in a view anyone can query.** Otherwise the analysis is lost and the same question gets reopened from scratch in six months.

5 States are not conditions

Eighty-nine percent of one operator board consisted of objects that asserted nothing wrong. Removing them removed no information and no alarm.

The single largest improvement available to most alarm boards is not tuning, rationalization, or staffing. It is removing objects that were never alarms.

Board object	Asserts	Belongs
State event	A transition occurred. Nothing is wrong. *Sunrise. Mode changed. Breaker closed as commanded.*	Event log, sequence-of-events
Condition	A state persists and will clear on its own. Real, trendable, rarely actionable inside its own lifetime.	Condition register with trending
Alarm	A condition requiring a person to act, now.	The operator board

On the assessed board, 6,965 of 7,839 rows were state events. One of them, a daily sunrise transition, had latched **2,092 times**. The remediation swept them and installed a BEFORE INSERT trigger so they land as archived with an explicit reason instead of latching. That trigger is the important half: **a sweep without a gate is a chore that runs forever.**

> **An honest note on suppression**
>
> "Suppression" is a word that makes safety-minded people uncomfortable, and it should. What is described here is not suppression in the sense of hiding a real condition. It is *filing*: putting an object where it belongs, with a recorded reason, where it remains queryable. Every swept row in the assessed system kept its full history and an explicit archive reason. Nothing was deleted. If your suppression mechanism deletes, it is not suppression, and Chapter 2 is about what that costs.

Calculation 5 · Classify the board before touching it

```
-- Run this BEFORE any suppression work. It sizes the prize
-- and, more usefully, it tells you what is NOT the prize.
SELECT COALESCE(c.class, 'UNCATALOGUED') AS class,
COUNT(*) AS rows_on_board,
ROUND(100.0*COUNT(*)/SUM(COUNT(*)) OVER (), 1) AS pct,
COUNT(DISTINCT a.event_name) AS distinct_codes
FROM alarms a
LEFT JOIN alarm_catalog c ON c.event_name = a.event_name
WHERE a.state IN ('active_unacked','acked')
GROUP BY 1 ORDER BY rows_on_board DESC;
```

Read the UNCATALOGUED row first. Codes on the board that no catalog describes are, by definition, codes nobody has decided about. On one assessed board every uncatalogued row turned out to be the platform's own derived observations. They had never been through the review that governed everything else, because they did not exist when the review was done.

6 Key stability, latching, and why acknowledgment can stop meaning anything

An alarm key is an identity claim: *this is the same condition you saw before.* When the key changes and the condition does not, acknowledgment silently stops working.

Unstable keys

Three alarm generators embedded epoch milliseconds in the key, so every evaluation cycle minted a new row for an unchanged condition. The consequences compound in a specific order, and only the last one is visible:

1. Row count grows without bound while the condition stands.
2. An operator acknowledges a row. The next evaluation creates a fresh unacknowledged row for the same condition.
3. The horn sounds again. The operator acknowledges again.
4. Within a shift, operators learn that acknowledgment does not work. They stop trusting the acknowledgment mechanism generally, including on alarms where it works fine.

Step four is the expensive one: a behavioral failure produced by a data-model defect. Trust in a control is not per-alarm. It is global. Operators treat a mechanism that fails visibly on some alarms as unreliable on all of them.

Calculation 6 · Key stability test

```
-- A stable key produces ONE row per standing condition.
-- rows_per_condition well above 1 = the key is minting duplicates.
SELECT split_part(alarm_key, ':', 1) AS generator,
site_id,
COUNT(*) AS rows_open,
COUNT(DISTINCT alarm_key) AS distinct_keys,
ROUND(COUNT(*)::numeric / NULLIF(COUNT(DISTINCT split_part(alarm_key,':',2)),0), 1) AS
rows_per_condition
FROM alarms
WHERE state IN ('active_unacked','acked')
GROUP BY 1,2
HAVING COUNT(*) > COUNT(DISTINCT split_part(alarm_key,':',2))
ORDER BY rows_per_condition DESC;
```

The interim remediation collapses a repeat arrival for the same <generator>:<site> onto the open row. That is a compensating control, and it was documented as one, with its removal condition written down: *remove it the day those generators emit a stable key.* A compensating control with no removal condition attached becomes permanent architecture by default.

The opposite failure: alarms that stop re-latching

The mirror image is more dangerous, because it removes alarms rather than duplicating them.

Bulk archive operations, such as a baseline reset or a maintenance clear-down, archived thousands of rows wholesale. Alarms that had *never actually cleared* kept their alarm key. On the next ingest cycle the reconciler saw a key it already knew about, so the condition never re-latched.

The consequence, measured on one assessed board, was a turnover screen reading **"0 critical, oldest critical: none"** while the underlying plant carried multiple standing critical conditions. They included breaker trips and, at one site, an isolated main power transformer.

The rule this produces. A bulk archive must either verify that each condition has genuinely cleared at source, or invalidate the keys it archives so the condition can re-latch on the next cycle. Silently keeping keys for conditions that were never resolved turns an administrative action into permanent blindness. And nobody sees the blindness, because the screen reports zero.

Standing alarm age: the measurement nobody publishes

Of 2,126 alarms on one assessed board, 2,096 were genuinely still asserting. But **257 had been asserting for more than a week**, and the tail is where the findings live:

Days asserting	Priority	What it was
458	**Critical**	A breaker status point
307	**High**	Breaker trip-coil supervision
307	**High**	Breaker trip-coil supervision, second unit
306	**High**	Breaker spring-charge supervision
68	**Critical**	Four feeder breakers, trip coil and relay trip

A breaker status Critical asserting for fifteen months is either a defect nobody closed or an instrument that has been lying for fifteen months. Both are findings, and the board cannot tell you which.

Age must be computed from the **source** clock, meaning the timestamp the upstream system assigned when the condition began. The local row's creation time will not do, because it resets whenever the local store is rebuilt. Where the source clock is unavailable and a local fallback is used, **the row is labeled as using a fallback**, because an age computed from the wrong clock is a number someone will quote.

7 The monitor that stops

A stopped evaluator produces exactly the same board as a plant with nothing wrong. In my view every alarm system in this industry rests on the assumption that silence means health, and almost none of them test it.

This is the alarm-domain instance of the general failure described in EC-WP-1100: a system that cannot distinguish *nothing is wrong* from *nothing is arriving*. In the alarm domain it has a sharper edge, because the alarm board is the control that is supposed to catch everything else failing.

Observed case

A storage state-of-charge monitor stopped at 07:50. At 11:29, nearly four hours later, the board still displayed a site at "6.7%, at or below 10%." The plant actually read 19.86% and had recovered hours earlier.

The alarm was not wrong when it was raised. It became wrong, silently, and nothing in the design could stop it being displayed as current.

The three-layer answer

Layer	What it does	Failure it catches
Evaluator heartbeat	Each monitor writes a timestamp on every completed pass, whether or not it raised anything. A monitor that has not written in more than its own interval is *down*.	The evaluator stopped
Scheduler health	A view over the job scheduler reporting last success and last failure per job.	The evaluator was never invoked
Per-alarm freshness	Every standing alarm carries the time its condition was last independently confirmed. Past a threshold it renders as *unconfirmed*.	This specific claim is no longer supported

The third layer is the one almost always missing, and it carries a design decision worth stating explicitly: **an unconfirmed alarm is labeled, not cleared.**

Clearing it asserts that the condition ended. Nothing supports that, because the monitor stopped and nothing knows what the condition is doing. Leaving it displayed as current asserts that the condition is confirmed, which is just as unsupported. The only defensible rendering is the third one: *this alarm stands, and we have not been able to confirm it for 3h 39m.*

Calculation 7 · Monitor health and alarm freshness

```
-- Layer 1: which evaluators have missed their own interval?
SELECT monitor_name, expected_interval_minutes, last_completed_at,
ROUND(EXTRACT(epoch FROM (now()-last_completed_at))/60, 1) AS minutes_since,
CASE WHEN now() - last_completed_at > (expected_interval_minutes * interval '2 minutes') THEN
'DOWN' ELSE 'ok' END AS health
FROM monitor_heartbeat ORDER BY minutes_since DESC;
-- Layer 3: standing alarms whose claim is no longer confirmed.
SELECT a.alarm_id, a.site_id, a.event_name, a.priority, a.last_confirmed_at,
ROUND(EXTRACT(epoch FROM (now()-a.last_confirmed_at))/60, 1) AS unconfirmed_min,
CASE WHEN now() - a.last_confirmed_at > interval '30 minutes' THEN 'UNCONFIRMED - do not
treat as current' ELSE 'confirmed' END AS freshness
FROM alarms a
WHERE a.state IN ('active_unacked','acked')
ORDER BY unconfirmed_min DESC;
```

The guard ledger, and a caution about documentation

One assessed platform's operator documentation described a control in precise terms: *"a database trigger refuses any low-state-of-charge row that carries no measurement or a dead-zero publish, and audits every refusal; see the guard ledger."*

There was no trigger, no monitor, and no ledger table. The ledger panel rendered a raw database error where the audit trail should have been.

The control was then built for real: refusal trigger, audit ledger, a monitor with the documented thresholds, and a state view. On its first pass it correctly refused and recorded a genuine dead-zero publish at one site, with state-of-charge reported as 0.00% and state-of-health and available charge both absent. That is exactly the case the control was described as handling. It had been passing through unrefused for as long as the documentation had been claiming otherwise.

The caution

Documentation that describes a control is not evidence that the control exists. It is worse than silence, because it stops anyone looking. Any control that appears in operator-facing documentation should have a query that demonstrates it, and that query should run before the documentation ships. **This one is ours.**

8 Ten questions for your own alarm system

Every one of these surfaced a real defect on the system described in this document. None requires a vendor.

#	Question	What a bad answer means
1	What percentage of your board is state events rather than conditions?	Above 20% and I would assume nobody is reading the board. On one assessed system it was 89%.
2	Does any alarm your platform computes itself survive an ingest cycle?	If a reconciliation sweep runs against an upstream list, verify it directly. Do not assume.
3	What is your suppression rate, and over what window was it measured?	A rate without a stated window is not a measurement. Check when the tagging mechanism began.
4	What decides an alarm's priority, and in what order?	If a client-supplied value can outrank a reviewed catalog, the catalog is decorative.
5	How long has your oldest standing Critical been asserting?	If the answer is in months, you have an unclosed defect or a lying instrument.
6	Does acknowledgment survive the next evaluation cycle?	If keys embed a timestamp, it does not, and your operators already know it.
7	After a bulk archive, can an uncleared condition re-latch?	If keys are retained, no. The board will report zero and be wrong.
8	If an evaluator stopped, how would the board look different?	If the answer is "it wouldn't," you have no monitor health.
9	Does any promotion rule depend on a premise the system cannot evaluate?	A daylight-dependent rule with no irradiance reference promotes a night-time flood.
10	Is every control described in your operator documentation demonstrable by query?	Documented-but-absent is worse than absent. It stops people looking.

What good looks like. Every row on the board is a condition a person can act on. The suppression rate is stated with its window. The priority resolution order is documented. Standing-alarm age is published and bounded, monitor health is visible, and every described control can be demonstrated. None of that requires new software. Most of it requires deciding that the board will tell the truth even when the truth is unflattering.

9 What RenewOps does about this

The reconciliation layer. It reads your monitoring stack, checks it against ratings, agreements and the documents of record, and declines to publish what it cannot defend.

This chapter walks the seven failure modes and the RenewOps alarm layer built to answer each. Everything below was measured on a live production instance between 24 and 26 August 2026 and is dated where the figure matters.

One definition of "on the board"

The board carries two states and no others: **unacknowledged-active** and **acknowledged**. Acknowledging an alarm no longer archives it. An alarm that has been acknowledged and is still asserting carries an explicit **ACK'D · STILL ASSERTING** badge. "I have seen this" and "this is over" are two different statements, and the board had been recording them as one.

The audible feed covers unacknowledged-active alarms only. An acknowledged alarm goes silent and stays visible. That is the whole point of an acknowledgment, and a horn wired to the same population as the board would not do it.

Status is not an alarm, and cannot latch again

The board went from **7,839 rows to 1,175**. Of the 6,965 status and informational rows, 48 were held deliberately and the rest were filed off the board. One daylight status event had been appearing on an operator's board **2,092 times**.

The sweep is the smaller half of that fix. The larger half is a rule installed at ingest so those rows **never latch again**. A new arrival of that class lands archived with an explicit archive reason attached, instead of landing on the board and waiting for the next cleanup. A board that has to be swept is a board that will need sweeping again.

Criticals went from seventeen present and none rendering, to thirty-two rendering.

Priority resolves in the order authority actually runs

Effective priority resolves through three tiers, in the order the authority genuinely runs. An **approved override** comes first. The **reviewed cause catalog** comes next, matched by event name and then by code. Only then does a classifier fallback apply. A priority posted by the browser can no longer contradict a cataloged code, which it previously could and did.

The write it refuses. A site-scoped priority override the classifier cannot read is **rejected at the point of writing**. Accepting it, storing it, and letting it silently do nothing would produce an override register that looks populated and governs nothing. An override that cannot take effect is worse than one never made, because somebody will later cite it.

A vocabulary check reports any priority word outside the known set. Without it, a new value arriving from upstream sorts quietly to the bottom of every board and nobody is told.

Held, with the reason held as data

Two catalog promotions are deliberately not applied, and the reason for each sits in a holds view rather than in a ticket:

- One catalog key covers three different breaker conditions, including an isolated main transformer. Promoting the key would have moved that condition *off* Critical.
- 582 instances of a second code were standing after dark. Promoting a night-time flood to High would have buried the thirty-two Criticals underneath it.

The daylight gate that would resolve the second case is designed and not built. It is carried as an open item, not a completed one. A hold with its reason recorded is a decision. A hold with no reason recorded is an omission, and whoever inherits it will reverse it.

Rollup that discards nothing

A grouped alarm takes the **worst priority among its children plus the full child count**. Nothing is discarded. The parent row also carries a **priority-mix qualifier on its face**. An operator who reads "High (High, Medium)" knows immediately that the group is not uniform. Five parent rows currently merge disagreeing priorities, and each says so, with asset counts and standing hours beside it.

A fix that was withdrawn before it ran. A proposed deduplication would have kept three assets at High *or* eleven at Medium and discarded the other. It was withdrawn. Under-reporting both count and severity is worse than staleness, because staleness at least announces itself in the timestamp. Nothing about the discarded rows would have.

Suppression measured against its own window

Suppression rate is published **with the window it was measured over**. The window runs from the later of twenty-four hours ago and the moment suppression tagging began, and its length is printed beside the figure. That correction moved a reported 14.2 percent to a measured **80.9 percent**. The number had not changed. The denominator had.

At 80.9 percent suppressed, 1,131 alarms per hour were still reaching the board.

Deferred. Alarm load is computed against a threshold of sixty alarms per operator-hour, a value configured in this platform. It has not been checked against the text of ANSI/ISA-18.2-2016, and it is not stated here as a standard.

The monitor that stops

The system publishes its own heartbeat, its monitor health, and the health of its scheduled jobs. Per-alarm freshness **labels an unconfirmed alarm rather than clearing it**. Two clocks are named separately: minutes since the newest onset, and minutes since the newest confirmation. A stopped feed cannot hide behind a calm board when the board publishes how long it has been since anything confirmed anything.

This matters because of what was found underneath it. A close-out sweep had been destroying every alarm RenewOps raised itself. **548 voltage-band and 246 schedule-deviation alarms were raised and deleted**, each stamped as though the plant had returned to normal, and not one was ever seen by an operator. The sweep now touches only alarms that originated upstream. On 25 August 2026, for the first time in the platform's life, sixteen voltage alarms stood across nine sites. Twelve were acknowledged within the hour.

The alarm system was not failing to detect. It was detecting correctly and then deleting the evidence on a schedule, and every screen downstream reported a quiet fleet.

Acknowledge what you say you are acknowledging

A bucket acknowledgment acknowledges the whole bucket server-side and returns the count actually changed. It refuses an unattributed call and an unrecognized priority, never writes an archive stamp, and writes its own bulk-acknowledgment record in the same transaction. The confirmation dialog shows the true bucket size. Before the fix, "Ack all (1,707)" acknowledged the hundred rows the browser had loaded and reported success.

What remains open

Item	Position as of 26 August 2026
Cause catalog disagreement	131 of 249 cataloged codes disagree with the priority currently in use, including a breaker trip-coil monitor sitting at High. Documented, not yet reconciled.
Override store empty	The governed override path exists and refuses bad writes. Nothing has used it. Applying an override today writes an audit trail and changes nothing until the classifier reads it.
Platform-generated flood	Forecast-deviation alarms embed a percentage in the description, so every distinct percentage is a distinct code and none can match the catalog. A nightly closure job stops them accumulating; the key shape is still wrong.
Attribution	Content is guarded well. Identity is a string the caller supplies. Who acknowledged, overrode, or held is not independently verified.

Figures in this chapter measured 24–26 August 2026 on a live production instance. They describe that instance on those dates and are not a claim about any other fleet.

Glossary

Term	Meaning in this paper
Acknowledgment	An operator's recorded statement that they have seen an alarm. It does not mean the condition is over.
Alarm	A claim about the plant that obliges a specific person to act, now. The definition this paper tests every board object against.
Alarm key	The identifier that tells the system an arrival is the same condition it saw before. A stable key yields one row per standing condition.
ANSI/ISA-18.2	*Management of Alarm Systems for the Process Industries*, the alarm-management standard published by the International Society of Automation and approved by ANSI. The 2016 edition is current.
Bulk archive	An administrative action that archives many alarm rows at once, such as a baseline reset or a maintenance clear-down.
Compensating control	An interim measure that offsets a known defect. In this paper it carries a written removal condition.
Condition	A real state that persists and clears on its own. Real and trendable, but rarely actionable inside its own lifetime.
Condition register	A trended record of conditions, kept off the operator board.
Daylight gate	A check that holds a sun-dependent promotion unless measured irradiance supports the premise that the sun is up.
Derived observation	A computed judgment, such as production against forecast. Useful, but not a plant condition.
Diagnostic	Instrument or system health, addressed to a different person on a different timescale from an operator alarm.
Evaluator heartbeat	A timestamp each monitor writes on every completed pass, whether or not it raised anything.
Exceedance factor	Per-operator alarm load divided by the threshold it is measured against.
Ingest sweep	A process that reconciles the local alarm store against the list of active alarms supplied by an upstream system.
Latching	An alarm row appearing on the board and staying there until it is cleared or archived. Re-latching is a new row for a condition that returns.
Per-alarm freshness	The time since a standing alarm's condition was last independently confirmed. Past a threshold the alarm is labeled unconfirmed, not cleared.

Term	Meaning in this paper
Priority resolution order	The documented sequence of sources a system consults to set an alarm's priority, highest authority first, first match wins.
RenewOps	The reconciliation platform built by Energy Compliance, Inc.
Self-clear rate	The share of an alarm code's occurrences that cleared without any action.
Standing alarm age	How long an alarm has been asserting, computed from the source system's clock.
State event	A record that a transition occurred, with nothing wrong. It belongs in the event log.
Suppression rate	The share of arrivals filed off the board with a recorded reason, over a stated measurement window.

How Energy Compliance Can Help

This reference is about what an alarm board does to the attention of the people who watch it, and about the one category of alarm nobody saw: the voltage-schedule alarms a platform raised and destroyed for months. That is an operations problem first. It is also an evidence problem, because the alarm record is what a Generator Operator reaches for when asked how it maintained the voltage schedule the Transmission Operator issued.

Where to start

An assessment points our defect battery at your live monitoring stack and hands you the findings, each with a source, a timestamp, and the question that resolves it, **whether or not you buy anything afterward.** Nobody buys a reconciliation layer before they know they need one, and the only honest way to find out is to look.

Monitoring Truth Assessment

The full defect battery against your live monitoring stack. Every finding comes back with a source, a timestamp, and the question that resolves it. Where we cannot verify something, it comes back marked deferred with the specific question, not softened into a maybe.

Evidence Provenance Review

Your compliance-relevant signals traced to source and sorted into citable and not. The seeded, the derived, and the self-referential are separated from the issued. For an alarm subsystem, that means establishing whether the voltage-band and schedule-deviation alarms you would cite in an audit were ever rendered to an operator at all.

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 each site that cannot compute.

Compliance program support and audit defense

Where the alarm record feeds a compliance obligation, we produce the evidence package, write the RSAW narrative, manage the data request, and sit on your side of the table when the Regional Entity asks how you know the number on the screen was true. Every engagement is overseen by a senior practitioner with regulator-side experience.

WattOps Compliance

WattOps Compliance is EC's NERC compliance evidence platform (wattopscompliance.com). If the voltage-schedule record in Chapter 2 is the kind of evidence you would have to produce,

this is where EC manages it. EC will run one standard at one facility through WattOps at no cost, CIP standards excluded, so you can see the result before you commit to anything.

RenewOps

RenewOps is EC's renewable operations monitoring and analytics platform, with a Data Center module. The alarm layer described in Chapter 9 is part of it. Contract terms are parsed to structure and signals resolved to roles. Every tile carries its own denominator, its own provenance, and its own age, and it refuses to compute, visibly, when it cannot.

Talk to us

Run the ten questions in Chapter 8 against your own board first. If any answer surprises you, or if you cannot get an answer at all, send me what you found and we will tell you what we would look at next. No pitch, no slide deck.

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 the Sultanate's Sector Law and Grid Code.

He founded Energy Compliance, Inc. to deliver senior regulator-side compliance authority without the layered staffing and billable-hour overhead of larger firms.

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.

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. Its focus is NERC, FERC, and RTO/ISO compliance, with 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 two platforms: WattOps Compliance, the firm's NERC compliance evidence platform, and RenewOps, the renewable operations monitoring and analytics platform described throughout this series.

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

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

Rigorous Compliance. Defensible Programs.

Sources and Authorities

Measured findings

- All system measurements in this reference (row counts, rates, durations, timestamps and priorities) are Energy Compliance first-party observations of live production platforms, anonymized under the Disclaimer & Legal Notice, and not available from a public source.

Alarm management

- American National Standards Institute webstore, ANSI/ISA-18.2-2016, *Management of Alarm Systems for the Process Industries*, listed as "Most recent" (accessed October 17, 2026). https://webstore.ansi.org/standards/isa/ansiisa182016
- ISA, *InTech*, "ISA18 alarm management standard updated" (May/June 2016), reporting ANSI approval of the 2016 edition. https://www.isa.org/intech-home/2016/may-june/departments/isa18-alarm-management-standard-updated
- Marcus Tennant, "Implementing Alarm Management per the ANSI/ISA-18.2 Standard," *Control Engineering* (September 2013), Figure 5, captioned as recommended alarm frequencies for an alarm system (about 6 per hour average very likely acceptable; about 12 maximum manageable). Secondary source. https://www.yokogawa.com/us/library/resources/media-publications/implementing-alarm-management-per-the-ansi-isa-182-standard-control-engineering/
- Donald G. Dunn and Nicholas P. Sands, "Solve common problems in industrial automation alarm systems," Automation.com (February 13, 2017) (fewer than six alarms per hour per console very likely acceptable; more than 12 about the maximum manageable). Secondary source. https://www.automation.com/article/solve-common-problems-industrial-automation-alarm-systems

NERC Reliability Standards

- NERC, Reliability Standard VAR-002-4.1, *Generator Operation for Maintaining Network Voltage Schedules*, Mandatory Subject to Enforcement, effective September 26, 2017, per the NERC VAR listing (accessed October 17, 2026); successor VAR-002-5 (Project 2021-02) not yet listed as approved or enforceable. https://www.nerc.com/pa/Stand/Reliability%20Standards/VAR-002-4.1.pdf

© Energy Compliance, Inc. · EC-WP-1101 · Status as of October 17, 2026