

YOUR POLICY DOES NOT EXECUTE ITSELF. BUILD EVIDENCE-PRODUCING RUNBOOKS.

The policy says the control will operate. The procedure describes the process. The GRC task turns green. None of those records necessarily tells an operator what to do now, against which population, with which permissions, what failure means, or what evidence must exist afterward.

PRACTITIONER POSITION

PCI DSS does not generally require an organization to name an artifact a runbook. It does require applicable policies and operational procedures to remain documented, current, in use, and known to affected parties, and it expects defined-frequency and event-driven activities to be performed and evidenced as applicable. My practitioner position is that a runbook is the operational layer between governance and execution. Policy establishes management intent. Standards define mandatory rules and configurations. Procedures describe how a process operates. A runbook binds one recurring or event-driven control activity to its exact trigger, subject, population, owner, permissions, steps, decision points, success state, failure path, escalation, and emitted evidence. The runbook is not the control, and its output is not automatically sufficient assessment evidence. But without that execution contract, the organization is often asking tribal knowledge, calendar reminders, GRC tickets, and assessment-season reconstruction to impersonate operations.

THE FIELD PATTERN

WHAT THIS LOOKS LIKE IN PRACTICE

The organization has a PCI policy, hardening standards, process documents, recurring GRC tasks, ticket queues, cloud consoles, SIEM alerts, vulnerability tools, and an evidence repository. The six-month network security control review still depends on one engineer remembering which accounts, firewalls, security groups, inherited rules, exceptions, and prior changes belong in the population. The quarterly access review begins from a spreadsheet copied from last quarter. A failed scan produces email but no bounded investigation record. When the assessor asks what happened, the team reconstructs screenshots, tickets, approvals, and timestamps after the fact. The control may have operated. The operating system cannot reliably tell the story.

SIGNALS DETECTED

WHAT SHOULD MAKE YOU STOP

- A control owner can cite the policy and GRC task but cannot identify the current operator instructions.
- Different qualified operators perform the same activity with different populations, steps, decisions, or outputs.

- The runbook documents the happy path but says nothing about failed scans, unauthorized rules, inappropriate access, missing evidence, or missed occurrences.
- The operator is told to retain evidence without a defined output schema, authoritative source, storage location, retention rule, or claim relationship.
- A calendar reminder supplies the cadence, but the applicable PCI DSS requirement or Targeted Risk Analysis is not referenced.
- A screenshot-and-click guide explains one interface but omits the control objective, decision logic, system boundary, population, and alternate tooling path.
- The current runbook names a departed employee, obsolete tool, retired account, changed architecture, or superseded approval path.
- An automated job emits logs but nobody has defined which result, deviation, approval, or exception still requires accountable human action.
- The runbook is considered tested because its author can execute it.
- Evidence collection starts months after execution and depends on reconstructing who did what from several systems.
- The assessment request list has become the de facto operating procedure because no maintained execution layer exists.
- A missed activity is closed by completing the next occurrence without preserving the missed occurrence, corrective action, recurrence prevention, and bounded assessment effect.

WHY IT MATTERS

THE COST OF LEAVING IT UNEXPLAINED

Treat the runbook as an execution contract, not a new documentation tier created for its own sake. The useful operating chain is Requirement to Policy to Standard to Procedure to Runbook to Execution to Evidence to Assessment. Each layer has a distinct job. Requirement language establishes the external obligation. Organizational policy and standards establish internal authority and rules. Procedures describe the process. The runbook tells a qualified operator how one activity is executed in the current environment. The resulting records support later evaluation; they do not predetermine it.

Name the execution gap. A documented control is not necessarily a reliably executed control. The gap appears when work is late, inconsistent, incomplete, performed from memory, bounded to the wrong population, silently fails, or leaves no attributable result. Staff absence, turnover, on-call pressure, incidents, technology change, acquisitions, tool migration, and assessment requests expose the gap because the usual operator can no longer supply the missing context orally.

Build the runbook around the occurrence. Give it a stable ID and version; related requirement and organizational authority; purpose; scope and exclusions; trigger; required frequency when applicable; owner and backup; roles and permissions; prerequisites; systems and tools; authoritative inputs; ordered actions; decision points; expected results; failure conditions; escalation criteria; evidence outputs; storage and retention destination; completion verification; exceptions; linked tickets, changes, and incidents; last review; and change history. Not every field needs a paragraph. Every material boundary needs an answer.

Do not confuse a runbook with a screenshot collection. Interface captures and command examples may be useful, but they age quickly and rarely explain why the step exists, which population is complete, which decision follows each result, or what to do when the expected screen is absent. A qualified substitute should understand the objective and decision logic well enough to use the current authorized interface safely, not merely replay a historical mouse path.

Make evidence an output of execution. Replace perform the task and remember evidence later with perform the task and emit an attributable occurrence record. A mature network-ruleset review, for example, should yield the reviewer, timestamp, authoritative population, systems and rules reviewed, inclusion and exclusion rules, exceptions, dispositions, approvals, linked changes, retained supporting objects, limitations, and next trigger. The evidence package begins while the work is happening, not when the assessment inbox opens.

Define the output contract before selecting the evidence repository. A GRC platform, ticketing system, SIEM, cloud console, vulnerability platform, identity system, change system, or evidence automation service may hold or reference the record. The platform does not decide what the occurrence must prove. The runbook should identify the authoritative source, required fields, timestamps, actors, population, result states, exceptions, immutable references where appropriate, sensitivity boundary, downstream consumers, and reopening conditions.

Encode time without inventing it. Runbooks may be event-driven, continuous, daily, weekly, monthly, quarterly, every six months, annual, after significant change, or governed by a Targeted Risk Analysis. Reference the authoritative requirement and the applicable TRA where one determines frequency. Record the trigger and due-window semantics. Do not let a convenient reminder, inherited ticket template, or slowly drifting operational habit become the source of the PCI cadence.

Write the failure path with the same care as the happy path. An unauthorized rule, inappropriate privilege, failed vulnerability scan, unresolved malware alert, failing ASV scan, unauthorized payment-page change, unavailable evidence object, or missed scheduled occurrence must lead somewhere. Define immediate containment or preservation actions, notification, accountable decision authority, ticket or incident creation, correction, evidence, recurrence prevention, retest, and which claims or conclusions reopen. A control without a defined failure path is only partially operationalized.

Preserve missed occurrences as history. PCI SSC FAQ 1572 explains that a missed defined-frequency activity cannot be retroactively performed or backdated. Corrective action and successful later performance matter, but they do not erase the missed period. The runbook should therefore create honest late, missed, failed, blocked, incomplete, corrected, and retested states rather than allowing one current green occurrence to rewrite the operating record.

Use trigger, action, decision, evidence, escalation as the minimum execution grammar. A failing scan triggers investigation; the operator validates scope and result, decides whether the finding is confirmed or disputed under the defined process, preserves the scan and analysis, opens remediation or dispute work, and escalates when severity, age, scope, or uncertainty crosses a defined threshold. The same grammar can govern access removal, key rotation, log review, payment-page change alerts, TPSP status review, and incident response without pretending those activities are operationally identical.

Keep human capability in the system. Runbooks reduce key-person dependence; they do not eliminate the need for trained and authorized people. Name the required knowledge, approval boundary, separation of

duties, and stop conditions. An operator must know when the runbook applies, when it does not, when a decision exceeds their authority, and when an apparent shortcut could change the payment-security claim.

Treat runbooks as living assets. Review them when architecture, systems, tools, provider services, roles, responsibilities, frequencies, sources, incidents, failures, assessment observations, or PCI DSS requirements change. Version them, review them, test them with someone other than the author, change them deliberately, and retire them when obsolete. The best cold test removes the usual operator and asks another qualified person to produce the expected bounded result without oral rescue.

Automate in layers. Level zero is tribal knowledge. Level one is a documented procedure. Level two is an operational runbook with trigger, owner, decisions, and outputs. Level three intentionally generates attributable evidence. Level four instruments timestamps, populations, results, tickets, approvals, and exceptions. Level five integrates execution, observation, evidence production, failure handling, and escalation. Higher maturity does not mean every decision is machine-made. It means automation performs the repeatable work while accountable people retain the decisions and professional judgments that automation cannot own.

Apply the Runbook Test. If the person who normally performs the control disappeared tomorrow, could another qualified and authorized person determine what triggered the activity, which systems and population apply, what to do, what success looks like, what failure looks like, where to escalate, and what evidence must exist afterward? If not, the process still contains tribal knowledge. If the substitute can execute it but a cold reviewer cannot reconstruct the occurrence, the runbook still has an evidence-engineering gap.

The goal is not more compliance documentation. The goal is repeatable, observable, attributable, evidence-producing operations. An assessor still evaluates applicability, performs required testing, challenges the population and evidence, and exercises professional judgment. The assessed entity gains something more durable than an easier evidence request: an operating system that already knows what happened, when it happened, who acted, what failed, what changed, and which record proves the bounded fact.

QUESTIONS

QUESTIONS THAT EXPOSE THE WACK

- 01 Which recurring or event-driven PCI activity still depends on one person's memory?
- 02 What exact requirement, organizational instruction, system boundary, and occurrence does the runbook operationalize?
- 03 What authoritative event or clock triggers execution, and what makes the population complete for that occurrence?
- 04 Which knowledge, roles, permissions, prerequisites, tools, and source records does a substitute operator require?
- 05 Where are the decision points, and which decisions exceed the operator's authority?
- 06 What is the expected result, and how is completion independently verified?

- 07 What failure states can occur, and what containment, escalation, correction, recurrence prevention, and retest follow each one?
- 08 What evidence is emitted as a natural output, and which facts, clocks, actors, populations, results, exceptions, and limitations does it preserve?
- 09 Which system is authoritative for each output, and where are references used instead of copied evidence?
- 10 Does the cadence come from the applicable requirement, an approved TRA, a change trigger, or merely an inherited reminder?
- 11 When was someone other than the author last able to execute the runbook without oral rescue?
- 12 Which architecture, tooling, ownership, provider, failure, or requirement change automatically reopens the runbook?

THE CLUSTER OF WACK

FIND / MAP / EXPLAIN / REDUCE / PROVE

ESCAPE THE WACK

01 / FIND

Choose one recurring or event-driven PCI activity that currently depends on memory, last year's request, or a generic GRC task. Recover the requirement, policy, standard, procedure, existing instructions, population source, completed and failed occurrences, evidence, exceptions, and assessment use.

02 / MAP

Build the execution spine from authoritative trigger through subject, population, roles, permissions, prerequisites, actions, decisions, success, failure, escalation, emitted evidence, storage, consumers, and reopening triggers. Keep policy authority, operator execution, evidence, assessor testing, and acceptance decisions separate.

03 / EXPLAIN

Write the runbook for a qualified substitute. Explain objective and decision logic, not only interface clicks. State explicit non-claims: successful execution does not automatically establish evidence sufficiency, requirement applicability, an assessment result, or recipient acceptance.

04 / REDUCE

Reference authoritative inventories and source records instead of copying them; generate structured occurrence records during execution; automate repeatable collection and routing; preserve human approvals, exceptions, investigations, and judgments as attributable decisions.

05 / PROVE

Remove the usual operator. Have a qualified substitute run a fictional, test, or safely sanitized occurrence, including one injected failure. Give the outputs to a cold reviewer and record the first oral rescue, missing field, ambiguous decision, failed escalation, and retest condition.

THE FIELD LAW

THE LINE WORTH REMEMBERING

A MATURE RUNBOOK DOES NOT TELL THE OPERATOR TO REMEMBER EVIDENCE. IT DEFINES WHAT THE OPERATION MUST EMIT.

Run the Evidence-Producing Runbook Contract on one recurring PCI activity. Remove the usual operator, inject one failure, and prove that execution produces an attributable occurrence record without an assessment-season reconstruction.

TECHNICAL GROUNDING

START WITH THE SOURCE MATERIAL

These references ground the practitioner interpretation. They do not turn the metaphor or method into official PCI DSS terminology.

- WACK-SOURCE-0001 / PCI DSS v4.0.1 / PCI Security Standards Council / [Open source](#)
- WACK-SOURCE-0096 / FAQ 1485: What is the meaning of initial PCI DSS assessment? / PCI Security Standards Council / [Open source](#)
- WACK-SOURCE-0098 / FAQ 1572: Can a compensating control be used when a periodic activity was missed? / PCI Security Standards Council / [Open source](#)
- WACK-SOURCE-0127 / OPS07-BP03: Use runbooks to perform procedures / Amazon Web Services / [Open source](#)

AUTHORITY CLASSIFICATION / PCI_SSC_SOURCE / ASSESSOR_JUDGMENT / PRACTITIONER_POSITION / PRACTITIONER_METHOD / AI_ASSISTED

Practitioner reviewed by Scott Norton on 2026-09-20 against canonical digest

2908b128c013d7fa0e885d3dda5634180337e3c8831bb41a6298e1a0a2071175. Educational content, not a compliance determination.

Independent practitioner resource; not affiliated with or endorsed by PCI SSC. Authorship and authority classification do not determine legal intellectual-property ownership.