SALT

PCI DSS V4.0.1

Securing payment APIs across the full compliance lifecycle

How Salt Security supports PCI DSS v4.0.1, from discovery to runtime enforcement

A SALT SECURITY WHITEPAPER · SEPTEMBER 2026

SALT

What's inside

Highlights

- **PCI DSS v4.0.1 is fully in effect.** All 51 future dated requirements from PCI DSS v4.x became mandatory on March 31, 2025, and the standard's transition period is over.
- **A full lifecycle model.** Salt's technical story has grown from API inventory alone into a complete chain: discover, understand data flows, govern policy, build secure code, monitor runtime behavior, and enforce protection.
- **Requirement 6.4.2 is now the active control** for public facing web application protection, and Salt's runtime detection plus Salt Managed Rules for AWS WAF give it a materially stronger role there.
- **Precise positioning.** Salt strengthens technical controls and produces audit evidence. It does not certify PCI compliance, replace a QSA or ISA assessment, or substitute for required ASV scanning.

01 INTRODUCTION

Payment APIs are the new perimeter

Every card transaction now passes through a chain of APIs: authorization, tokenization, fraud scoring, settlement, and the mobile or web front end the customer actually touches. PCI DSS has always protected cardholder data. What has changed is how much of that data now moves through interfaces that traditional network controls were never built to see.

PCI DSS v4.0.1 is the current published version of the standard, and as of March 2025 its full set of requirements, including 51 that were previously future dated, apply in force. For organizations that store, process, or transmit cardholder data through APIs, that shift raises the bar on what "securing the API" actually needs to cover.

This paper updates Salt Security's earlier guidance on PCI DSS and API security. It corrects a few claims that no longer hold up against the current standard, and it reflects how far the Salt platform has grown since API inventory was the whole story. The goal is a technically accurate, defensible picture of where Salt's platform supports PCI DSS today.

02 WHAT CHANGED

PCI DSS v4.0.1 is fully in effect

PCI DSS v4.0.1 was published on June 11, 2024, as a limited revision of v4.0. It clarified existing language; it did not add or remove requirements. PCI DSS v4.0 was retired on December 31, 2024, which makes v4.0.1 the only current version of the standard.

The transition period is over

PCI DSS v4.0 introduced 64 new requirements, 51 of which were future dated. All 51 became effective on March 31, 2025. For applicable assessments, these are no longer best practices to plan for. They are requirements to meet now.

Requirement 6.4.2 replaced 6.4.1

After March 31, 2025, Requirement 6.4.2 is the effective control for protecting public facing web applications, and 6.4.1 is superseded and reported not applicable. 6.4.2 calls for an automated technical solution that detects and prevents web based attacks, a stronger and more specific bar than the older requirement, and a better match for API focused runtime protection.

Payment page scripts are now a named risk

Requirements 6.4.3 and 11.6.1 also became effective on March 31, 2025. PCI SSC paired them with dedicated guidance on e-skimming: the practice of injecting malicious scripts into payment pages to steal card data as it's entered. 6.4.3 requires that scripts running on payment pages are authorized, that their integrity is assured, and that an inventory with business justification is maintained. 11.6.1 requires a mechanism to detect and alert on unauthorized changes to payment pages and the security relevant HTTP headers a browser receives.

SAQ A changed too, but the underlying requirements didn't go away

PCI SSC's 2025 revision of SAQ A removed 6.4.3 and 11.6.1 from that specific questionnaire and added an eligibility criterion instead: merchants must confirm their site isn't susceptible to script based attacks. That's a change to how a subset of merchants document eligibility. It is not a change to what PCI DSS itself requires.

The standard keeps moving

PCI SSC ran a request for comments on v4.0.1 between June and July 2026 to help shape its next iteration. As of this writing, v4.0.1 remains the current published standard. Treat any PCI mapping, including this one, as dated content to revisit as the standard evolves.

Two specifics worth getting right

Encryption in transit. Requirement 4.2.1 calls for strong cryptography and security protocols to protect cardholder data moving across open, public networks. It doesn't name one universal protocol version as the rule; a current, well configured protocol is how an organization meets that bar.

Software and component inventory. Requirement 6.3.2 asks for an inventory of bespoke and custom software and the third party components built into it, kept current for vulnerability and patch management. An API inventory is valuable supporting evidence toward that requirement, and organizations typically pair it with a broader software and component inventory to cover 6.3.2 in full.

A full lifecycle model for PCI DSS and API security

Securing payment APIs isn't a single control. It's a chain, and PCI DSS v4.0.1 touches nearly every link in it. Salt's platform maps to that chain in six stages.

01 Discover the full attack surface

Salt Surface finds internet exposed APIs and endpoints from outside the organization, before any integration takes place. Salt Connect adds agentless, inside out visibility into cloud providers, API gateways, databases, and hosting platforms, without needing to intercept traffic. Together with continuous API discovery, they surface shadow and zombie APIs that never made it into a formal inventory.

PCI value: supports scope validation, exposes unknown public facing endpoints, and helps confirm the asset inventory work required under Requirements 12.5.1 and 12.5.2.

02 Understand where cardholder data actually flows

Sensitive data discovery classifies which APIs handle cardholder data, maps how it moves, and flags where it appears on public or unauthenticated endpoints. It also validates transport layer protection, surfacing unencrypted or weakly protected paths.

PCI value: strengthens the technical case for Requirement 4.2.1 and gives teams evidence of where cardholder data actually travels, not just where it's assumed to travel.

03 Govern policy continuously

Posture Governance and Policy Hub apply organization specific security policy across the API estate, including out of the box PCI DSS framework support. They detect drift from that policy and export evidence for audit workflows, rather than relying on a point in time review.

PCI value: turns PCI DSS into continuous evidence, supporting Requirements 6.5.1 and 6.5.2 and the posture story auditors ask about.

04 Build secure code before it reaches production

Salt Code extends policy enforcement into AI coding assistants and existing repositories, checking for authentication gaps, sensitive data handling, hardcoded secrets, and risky logging patterns before code ships.

PCI value: one of Salt's strongest PCI alignments, mapping directly to Requirement 6.2.4's call to prevent business logic abuse through API manipulation, and supporting the code review expectations in 6.2.3 and 6.2.3.1.

05 Monitor real runtime behavior

Salt Collect gathers live traffic across more than 70 technologies. Runtime behavioral analysis baselines normal API and user behavior, then flags account takeover, data exfiltration, session abuse, and slow moving attacks that static configuration checks can't see.

PCI value: catches the business logic abuse Requirement 6.2.4 calls out by name, after code ships and attackers start probing it in production.

06 Enforce protection at the edge

Targeted blocking stops malicious API activity in real time, and Salt Managed Rules for AWS WAF extend that protection directly into AWS WAF, covering attack classes like credential brute forcing, SSRF, prototype pollution, excessive GraphQL queries, and JWT anomalies.

PCI value: Salt's strongest connection to Requirement 6.4.2, the automated detection and prevention control now in effect for public facing web applications, where that public facing API traffic sits within the covered architecture.

The Agentic Security Graph correlates all six stages, connecting outside in exposure, configuration, code, and runtime behavior into one picture. It isn't itself a PCI DSS requirement, but it's what makes the rest of this model coherent instead of six disconnected tools.

04 REQUIREMENT MAPPING

Requirement by requirement: where Salt fits

The table below maps Salt's platform against the PCI DSS v4.0.1 requirements where the fit is strongest. "Direct or strong fit" means Salt performs an important technical function the requirement describes. "Strong support" means Salt materially contributes visibility, evidence, or governance, alongside other required controls. "Adjacent" means Salt provides useful context without being a primary control. None of these classifications equal compliance certification. Only a qualified assessor can determine that.

Requirement	What it asks for	How Salt helps	Fit
6.2.4	Prevent business logic and API manipulation attacks	Salt Code enforces secure API design pre-release; runtime detection catches business logic abuse, account takeover, and access control attacks after deployment	**Direct / strong fit**
6.2.1	Develop bespoke and custom software securely	Salt Code applies security and compliance policy in coding assistants and CI/CD workflows before release	**Direct / strong fit**
6.2.3, 6.2.3.1	Code review appropriate to the requirement, with independence and approval controls for manual review	Salt Code reviews existing repositories and returns file and line level findings; it does not by itself satisfy personnel independence or approval requirements for manual review	**Strong support**
6.3.1, 6.3.2	Vulnerability management and software component inventory	Salt Connect, continuous discovery, and Salt Code reveal deployed APIs, ownership, and code level dependencies*	**Strong support**
6.4.2	Automated detection and prevention for public facing web applications	Runtime detection plus Salt Managed Rules for AWS WAF block API specific attack classes in real time	**Direct fit for API surface**

6.5.1, 6.5.2	Secure change management	Salt Code tests changes against policy pre-deployment; Posture Governance flags drift afterward	**Strong support**
12.5.1, 12.5.2	Maintain and confirm PCI scope, including data flows	Salt Surface, Connect, and sensitive data discovery reveal exposed and internal APIs and cardholder data paths	**Strong support**
4.2.1	Strong cryptography for cardholder data in transit	Sensitive data discovery validates transport protection and flags unencrypted paths	**Strong support**
7, 8	Access control, authentication, and account management	Salt identifies authentication and authorization posture gaps and detects abuse of valid credentials	**Strong support**
3.2.1, 10, 11.3, 12.3.1	Data retention, audit logging, vulnerability scanning, targeted risk analysis	Salt contributes exposure, posture, and behavioral context that inform each program	**Adjacent**

*An API inventory is powerful supporting evidence for Requirement 6.3.2. It doesn't replace the software and component inventory the requirement names directly; most organizations still need a complete component inventory alongside it.

05 BOUNDARIES

What Salt does not replace

Part of building a defensible PCI DSS story is being precise about where Salt's platform stops. Salt does not replace:

- A qualified security assessor's or internal security assessor's compliance determination
- Required Approved Scanning Vendor (ASV) scans
- Identity, MFA, or privileged access management systems
- Enterprise patch management processes
- Formal change approval and documented rollback procedures
- The complete PCI DSS audit logging program across every in scope system

Salt strengthens the technical controls and evidence around APIs specifically. The rest of the PCI DSS program still needs the people, process, and complementary tools it has always needed.

06 NEXT STEPS

Getting started

PCI DSS v4.0.1 asks more of API security than the 2024 conversation acknowledged, and most organizations are still catching up to what changed on March 31, 2025. A full lifecycle view, discovering every API, understanding where cardholder data actually goes, governing policy continuously, building securely, watching runtime behavior, and enforcing protection at the edge, gives assessors and internal teams a coherent, evidence backed answer instead of a point in time snapshot.

If you're preparing for an assessment, evaluating your current API inventory against 12.5.1 and 12.5.2, or trying to understand what 6.4.2 means for your public facing applications, Salt's team can walk through your specific architecture and where the platform fits.

Talk to Salt Security about PCI DSS and API security at salt.security

Sources

1. PCI Security Standards Council, "Just Published: PCI DSS v4.0.1," blog.pcisecuritystandards.org
2. PCI Security Standards Council, "Now is the Time for Organizations to Adopt the Future-Dated Requirements of PCI DSS v4.x," blog.pcisecuritystandards.org
3. PCI Security Standards Council, "New information supplement: Payment page security and preventing e-skimming," blog.pcisecuritystandards.org
4. PCI Security Standards Council, FAQ 1593 on Requirement 6.4.2's effective date, pcisecuritystandards.org/faqs/1593
5. PCI Security Standards Council, "Important updates for merchants validating to Self-Assessment Questionnaire A," blog.pcisecuritystandards.org
6. PCI Security Standards Council, "Request for Comments: PCI Data Security Standard (PCI DSS) v4.0.1," blog.pcisecuritystandards.org
7. PCI Security Standards Council, FAQ 1085 on Requirement 4.2.1, pcisecuritystandards.org/faqs/1085
8. PCI DSS v4.0 SAQ D for Service Providers and SAQ C, listings.pcisecuritystandards.org

---