aws

AWS Security Reference Architecture - Payment Card Industry (PCI) Data Security Standard (DSS)

AWS Prescriptive Guidance

AWS Prescriptive Guidance: AWS Security Reference Architecture - Payment Card Industry (PCI) Data Security Standard (DSS)

Table of Contents

AWS Security Reference Architecture - Payment Card Industry (PCI) Data Security Standard (DSS)

Avik Mukherjee, Amazon Web Services

This guidance extends the AWS Security Reference Architecture (AWS SRA) - Core Architecture to help organizations design and operate a multi-account AWS environment that aligns with the Payment Card Industry Data Security Standard (PCI DSS). It is an architecture guide that shows how to apply AWS SRA security best practices in a way that satisfies PCI DSS requirements for workloads that store, process, or transmit cardholder data.

This guide is intended for:

- Security architects designing or extending an AWS multi-account landing zone for workloads under PCI DSS scope
- Compliance engineers mapping AWS controls to PCI DSS requirements during an assessment
- Cloud platform teams building shared security services that must accommodate a cardholder data environment (CDE)
- Qualified Security Assessors (QSAs) and Internal Security Assessors (ISAs) who want to understand how AWS SRA patterns satisfy PCI DSS intent

PCI DSS Primer

The Payment Card Industry Data Security Standard (PCI DSS) is an information security standard maintained by the PCI Security Standards Council (PCI SSC). It applies to any organization that stores, processes, or transmits payment card data, regardless of size, transaction volume, or geography.

The standard was created jointly by Visa, Mastercard, American Express, Discover, and JCB to establish a consistent baseline of security controls that protect cardholder data throughout its lifecycle. Compliance is enforced contractually by the payment brands through acquiring banks and payment processors.

PCI DSS version 4.0.1 (effective December 31, 2024, with a transition deadline of March 31, 2025 for future-dated requirements) organizes its controls into six goals and twelve principal requirements:

Goal	Requirements
Build and Maintain a Secure Network and Systems	1. Install and maintain network security controls
	2. Apply secure configurations to all system components
Protect Account Data	3. Protect stored account data
	4. Protect cardholder data with strong cryptography during transmission
Maintain a Vulnerability Management Program	5. Protect all systems and networks from malicious software (out-of-scope for this guide)
	6. Develop and maintain secure systems and software
Implement Strong Access Control Measures	7. Restrict access to system components and cardholder data by business need to know
	8. Identify users and authenticate access
	9. Restrict physical access to cardholder data (out-of-scope for this guide)
Regularly Monitor and Test Networks	10. Log and monitor all access to system components and cardholder data
	11. Test security of systems and networks regularly
Maintain an Information Security Policy	12. Support information security with organizational policies and programs

In a cloud environment, PCI DSS responsibilities are shared between the cloud service provider (CSP) and the customer. AWS has a Service Provider PCI Report on Compliance (RoC) and operates

under a shared responsibility model for PCI DSS. To meet requirement 12.9, both the RoC report and Shared Responsibility Matrix are available to download within AWS Artifacts.

The architectural patterns and controls described in this guidance apply equally to merchants and service providers, including multi-tenant hosting providers subject to PCI DSS Appendix A1. Organizations operating as service providers will find that the multi-account isolation model with dedicated accounts per customer, OU-level SCPs, per-account encryption keys, and centralized per-account logging directly addresses the logical separation and audit log requirements of Appendix A1. AWS provides the building blocks, customers are responsible for assembling them securely. This guide shows you how to do that using AWS SRA patterns.

AWS SRA Context

The AWS SRA is a holistic, prescriptive security architecture guide that describes how AWS security services fit together across a multi-account AWS environment. It is built around a modular, three-tier web architecture (web, application, and data tiers) and is intentionally designed to be adapted — not every workload needs every service, but AWS SRA provides the full range of options and their architectural relationships.

This PCI DSS deep dive does not replace the AWS SRA — it extends it by:

- Mapping AWS SRA account types to PCI DSS scoping boundaries — showing which accounts are in scope, connected-to, or out-of-scope in a typical payment architecture
- Layering PCI-specific controls onto the existing AWS SRA service configurations — e.g., additional logging granularity, encryption requirements, or network restrictions that go beyond AWS SRA baseline
- Identifying where the standard AWS SRA guidance is necessary but not sufficient for PCI DSS, and providing additional prescriptive guidance to help align with PCI DSS requirement.

Key AWS SRA design principles relevant to PCI DSS include:

- Implement a strong identity foundation − Implement the principle of least privilege, and enforce separation of duties with appropriate authorization for each interaction with your AWS resources. Centralize identity management, and aim to eliminate reliance on long-term static credentials.

- Enable traceability – Monitor, generate alerts, and audit actions and changes to your environment in real time. Integrate log and metric collection with systems to automatically investigate and take action.
- Apply security at all layers – Apply a defense-in-depth approach with multiple security controls. Apply multiple types of controls (for example, preventive and detective controls) to all layers, including edge of network, virtual private cloud (VPC), load balancing, instance and compute services, operating system, application configuration, and code.
- Protect data in transit and at rest – Classify your data into sensitivity levels and use mechanisms such as encryption, tokenization, and access control where appropriate.
- Keep people away from data – Use mechanisms and tools to reduce or eliminate the need to directly access or manually process data. This reduces the risk of mishandling or modification and human error when handling sensitive data.
- Prepare for security events – Prepare for an incident by having incident management and investigation policy and processes that align to your organizational requirements. Run incident response simulations and use tools with automation to increase your speed for detection, investigation, and recovery.

Attachments

To access additional content that is associated with this document, download and unzip the following file:

attachment.zip

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current Amazon Web Services (AWS) product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided "as is" without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

The information within this guide is presented as informational and is for reference only. Customers must manage their own PCI DSS compliance certification. While customers may use AWS services to store, transmit, or process their own cardholder data (CHD), AWS does not directly store, transmit, or process any customer (CHD). This compliance guide is provided as a courtesy to facilitate customers' consideration and review of their cardholder data environment (CDE) created using AWS services as they prepare for their PCI DSS assessments. This document does not replace or change the contents of the AWS PCI Responsibility Summary published with the PCI DSS Attestation of Compliance (AOC).

About the AWS SRA library

This guide is part of a library that provides architectural blueprints and technical guidance for designing and building security architectures on AWS. The library consists of implementation code (AWS SRA code library), a validation tool (SRA Verify), and two complementary categories of guides that cover the core architecture and deep dive architectures.

AWS SRA – core architecture

This guide represents a foundation for the recommended AWS security architecture. It is the starting point that applies to all organizations, regardless of their industry, application type, or any other considerations. This foundation helps you build a strong and scalable architecture on AWS and helps create a strong AWS multi-account security baseline that securely scales as your business grows.

AWS SRA – deep dive architectures

The AWS SRA – core architecture guide is complemented by additional publications that provide architectural patterns aligned to specific security capabilities, application types, and compliance or regulatory requirements. These patterns extend the core architecture and should be used in conjunction with the AWS SRA – core architecture guide.

The following guides provide architectural patterns aligned to specific security capabilities:

- AWS SRA – identity management provides guidance on how to implement a scalable, robust, and centralized identity and access management solution on AWS.
- AWS SRA – perimeter security discusses architecture patterns and AWS services for implementing edge security in a central account or in individual accounts.
- AWS SRA – cyber forensics describes how to configure an AWS Forensics account as a starting point to develop your organization's forensic capabilities and to help improve your security incident response (IR) preparedness.

The following guides provide architectural patterns for specific application types. You might want to focus on these after you build your baseline security architecture:

- AWS SRA – generative AI provides security architectural recommendations for designing and building applications that incorporate generative AI capabilities by using AWS generative AI services.
- AWS SRA – IoT provides security architectural recommendations for designing and building IoT applications on AWS.

In addition, the following guide describes architectural patterns that are aligned with specific compliance or regulatory frameworks:

- AWS Privacy Reference Architecture (AWS PRA) provides a security architecture for applications that process personal data and must support broad privacy compliance requirements such as the General Data Protection Regulation (GDPR), the California Consumer Privacy Act (CCPA), or the Brazilian General Data Protection Law (LGPD). The AWS PRA provides a set of guidelines that are specific to the design and configuration of privacy controls in AWS services.
- AWS SRA – PCI DSS (this guide) provides guidance to help organizations design and operate a multi-account AWS environment that aligns with the Payment Card Industry Data Security Standard (PCI DSS)

We recommend that you start with the AWS SRA – core architecture guide to understand the foundational architecture and then consult the complementary guides to take advantage of advanced functionality and implementations. For more information about this content set, refer to AWS Security Reference Architecture.

AWS Organization Architecture – OU and Account structure

The AWS Security Reference Architecture (AWS SRA) organizes all AWS accounts within a single AWS Organization, structured across purpose-built Organizational Units (OUs). This multi-account design is foundational to PCI DSS compliance because AWS accounts act as hard security, access, and billing boundaries — by default, no access is permitted between accounts. This native isolation is a powerful primitive for PCI DSS scoping and segmentation that goes beyond traditional network-layer controls.

The AWS SRA defines the following core OUs and accounts:

- Org Management account — Root of the AWS organization; central governance via AWS Organizations SCPs and AWS Control Tower
- Security OU – Security Tooling account — Hosts broadly applicable security services (Amazon GuardDuty, AWS Security Hub, AWS Config, Amazon Inspector, Amazon Detective); centralized security monitoring and automated alerting
- Security OU – Log Archive account — Immutable, centralized ingestion and archival of all logs and backups across all accounts and Regions
- Infrastructure OU – Network account — Isolates networking services (VPCs, Transit Gateway, AWS Network Firewall, ingress/egress controls) from application workloads; the gateway between workloads and the internet
- Infrastructure OU – Shared Services account — Hosts services consumed by multiple teams (e.g., AWS IAM Identity Center, Active Directory, messaging services)
- Workloads OU – Application account(s) — Hosts application workloads; isolated per service rather than per team to improve resilience

PCI DSS Scoping for an Account or Infrastructure Component

Under PCI DSS 4.0.1, the scoping process begins by identifying all system components — people, processes, and technologies — that interact with or could impact the security of cardholder data (CHD) or sensitive authentication data (SAD). The PCI Council defines three scoping categories that directly map to how AWS accounts and infrastructure should be classified:

Category 1 — CDE Systems (In-Scope, Full PCI DSS Requirements Apply) A system component is in the CDE if it stores, processes, or transmits CHD or SAD, or if it resides in the same network segment as systems that do. On AWS, this means any account, VPC, subnet, or service that directly handles PANs, CVVs, expiry dates, or track data. Examples include payment processing Lambda functions, Aurora databases storing encrypted PANs, and API Gateway endpoints receiving card data.

Category 2 — Connected-To / Security-Impacting Systems (In-Scope, Applicable PCI DSS Requirements Apply) A system component is "connected-to" if it has connectivity to a CDE system and could impact the security of the CDE — even if it never touches CHD directly. On AWS, cloud-native connectivity (cross-account IAM roles, VPC peering, Transit Gateway routes, shared services) can inadvertently pull infrastructure into scope. A system is security-impacting if it provides security services to the CDE (e.g., a centralized logging account, a shared identity provider, a secrets management service). The AWS SRA's Security Tooling account, Log Archive account, and Network account will typically fall into this category and must be assessed accordingly.

Category 3 — Out-of-Scope Systems A system component is out of scope only if it is fully isolated from the CDE — no connectivity path exists, and it provides no security services to the CDE. On AWS, achieving true out-of-scope status requires demonstrating that no network path, IAM trust relationship, shared service dependency, or management-plane access connects the component to the CDE. This is where the AWS SRA's account-level hard boundaries are most powerful: an account in a separate, non-peered OU with no cross-account roles and no shared services touching the CDE can be credibly argued as out of scope to a QSA.

The key determinants for scoping any AWS account or infrastructure part is answering the question, "If a bad actor could compromise this account or component, could they use it to access cardholder data or compromise the security of it?" if the answer is yes, the component should be considered in scope. Additional detailed considerations are:

- Data flow — Does CHD or SAD traverse this component?
- Network connectivity — Is there a routable path (direct or transitive) to a CDE system?
- IAM and management-plane access — Can a principal in this account assume roles in, or invoke APIs against, CDE accounts?
- Security service dependency — Does the CDE rely on this component for logging, secrets, identity, patching, or key management?
- Shared infrastructure — Is any underlying infrastructure (Transit Gateway, DNS, Active Directory) shared between CDE and non-CDE workloads?

The AWS SRA Recommendations supporting PCI DSS Requirements

The AWS SRA's multi-account, multi-OU architecture directly maps to and accelerates compliance with PCI DSS 4.0 requirements across two critical dimensions: segmentation and resource visibility.

Segmentation

The PCI Council's Guidance for PCI DSS Scoping and Network Segmentation advises the CDE to be isolated from systems that could impact its security. The AWS SRA supports this through multiple layers:

- Account-level segmentation — Placing CDE workloads in dedicated AWS accounts within a PCI Workloads OU creates a hard boundary that is not dependent on network configuration alone (Refer to recommended OU and account structure depicted later in this guide under Account and OU structure) . This directly reduces PCI DSS scope by ensuring out-of-scope systems have no connectivity path to CDE accounts by default.
- SCP-enforced guardrails — AWS Organizations SCPs applied at the OU level prevent CDE accounts from establishing unauthorized cross-account access or enabling unapproved services, supporting Requirement 1.2 (network security control configurations are reviewed and validated).
- Network-level segmentation — the AWS SRA's Network account centralizes ingress/egress controls using AWS Network Firewall, AWS WAF, and security groups, enforcing east-west (internal) and north-south (external) traffic controls aligned with Requirement 1.3 (restrict inbound and outbound traffic to that which is necessary).
- VPC design — Dedicated VPCs with private subnets, no direct internet gateway attachment for CDE tiers, and Transit Gateway-mediated routing through the Network account enforces the segmentation boundary at the network layer.

The updated AWS whitepaper Architecting for PCI DSS Scoping and Segmentation on AWS provides actionable network design patterns and reference architectures for OU and account structures specifically designed for PCI DSS 4.0 compliance.

Resource Visibility

PCI DSS Requirement 12.5.1 requires maintaining an inventory of all in-scope system components, and Requirement 12.5.2 requires that PCI DSS scope is documented and confirmed at least

once every 12 months. Requirement 6.3.2 also requires an accurate, up-to-date Software Bill of Materials (SBOM) or software inventory for all bespoke, custom, and third-party software components. The AWS SRA supports continuous, automated resource discovery through:

- AWS Config (deployed in the Security Tooling account) — AWS Config Continuously records configuration state of all resources across all accounts and Regions, providing a real-time inventory of in-scope system components.
- Amazon Inspector SBOMgenerator - The Amazon Inspector SBOM Generator (Sbomgen) is a tool that produces an SBOM for archives, container images, directories, local systems, and compiled Go and Rust binaries.
- AWS Organizations integration — All accounts are visible and manageable from the Org Management account, ensuring no account or resource falls outside the governance perimeter.
- AWS Systems Manager - AWS Systems Manager Inventory provides visibility into AWS computing environment. Inventory can be used to collect metadata from all managed nodes and stored in a central Amazon Simple Storage Service (Amazon S3) bucket. Using built-in tools, the data can be queried to quickly determine which nodes are running the software and configurations required by organization defined software policy.
- Tags - A tag is a key-value pair applied to a resource to hold metadata about that resource. Users can create and apply Tags to AWS resources using the AWS CLI, API, or the AWS Management Console. Tags can help correctly identify in-scope and out-of-scope resources dynamically without having to manually keep track of every resource in a dynamic environment. It should be noted that tags are not encrypted and should not be used to store sensitive data.

Together, these AWS SRA-recommended services create a continuously validated, automatically documented inventory of the CDE — reducing the manual effort of annual scoping exercises and providing auditors with real-time evidence of compliance posture. Additionally you should also design a strong change management process implemented through Infrastructure as Code (IaC) to ensure you have an accurate inventory of your AWS environment and only authorized changes are introduced.

Workload Account Structure: CDE and Connected-To Accounts

For PCI DSS environment, we need to extend the AWS SRA Workload OU and account structure further, as this is where the CDE is hosted. The AWS SRA's Workloads OU houses application accounts, but for PCI DSS environments, a flat "one Workloads OU" model is not recommended

for controlling the scope. The AWS Architecting for PCI DSS Scoping and Segmentation on AWS whitepaper recommends structuring the Workloads OU — and the broader organization — around the three scoping categories above, using dedicated child OUs and accounts to make scope boundaries explicit, enforceable, and auditable.

The recommended account structure maps as follows:

PCI Workloads OU — CDE Account(s): These accounts host infrastructure that directly stores, processes, or transmits CHD. Each CDE account should be isolated to a single application or payment flow to minimize scope and simplify scoping. Key characteristics:

- Dedicated VPC with private subnets only for CDE tiers (no direct internet gateway attachment)
- All ingress/egress routed through the Network account's inspection VPC via Transit Gateway
- No shared encryption keys, IAM roles, or service endpoints with non-CDE accounts
- AWS KMS Customer Managed Keys scoped exclusively to the CDE account
- SCPs on the PCI Workloads OU deny actions that would expand scope e.g., creating VPC peering to non-PCI accounts, disabling CloudTrail, or attaching internet gateways
- RCP should also be applied to resources to prevent access from non PCI accounts. As an example, you can apply RCP to restrict access to specified AWS services for all principals except those in a specific OU path, which belongs to PCI in-scope OU. For other sample RCPs, refer to resource control policies example on GitHub.

PCI Workloads OU — Connected-To Account(s): These accounts host systems that have connectivity to the CDE but do not directly handle CHD — for example, a tokenization service, a fraud detection engine, or a payment orchestration layer that routes transactions to the CDE. They are fully in scope for applicable PCI DSS requirements and must be treated with the same rigor as CDE accounts for controls like access management (Requirement 7), logging (Requirement 10), and vulnerability management (Requirement 6/11). Key characteristics:

- Connectivity to CDE accounts is strictly controlled — only specific, documented flows are permitted (e.g., via AWS PrivateLink or tightly scoped Transit Gateway route tables)
- No broader internet exposure than what is required for their function
- Inherit the same SCP guardrails as CDE accounts within the PCI Workloads OU

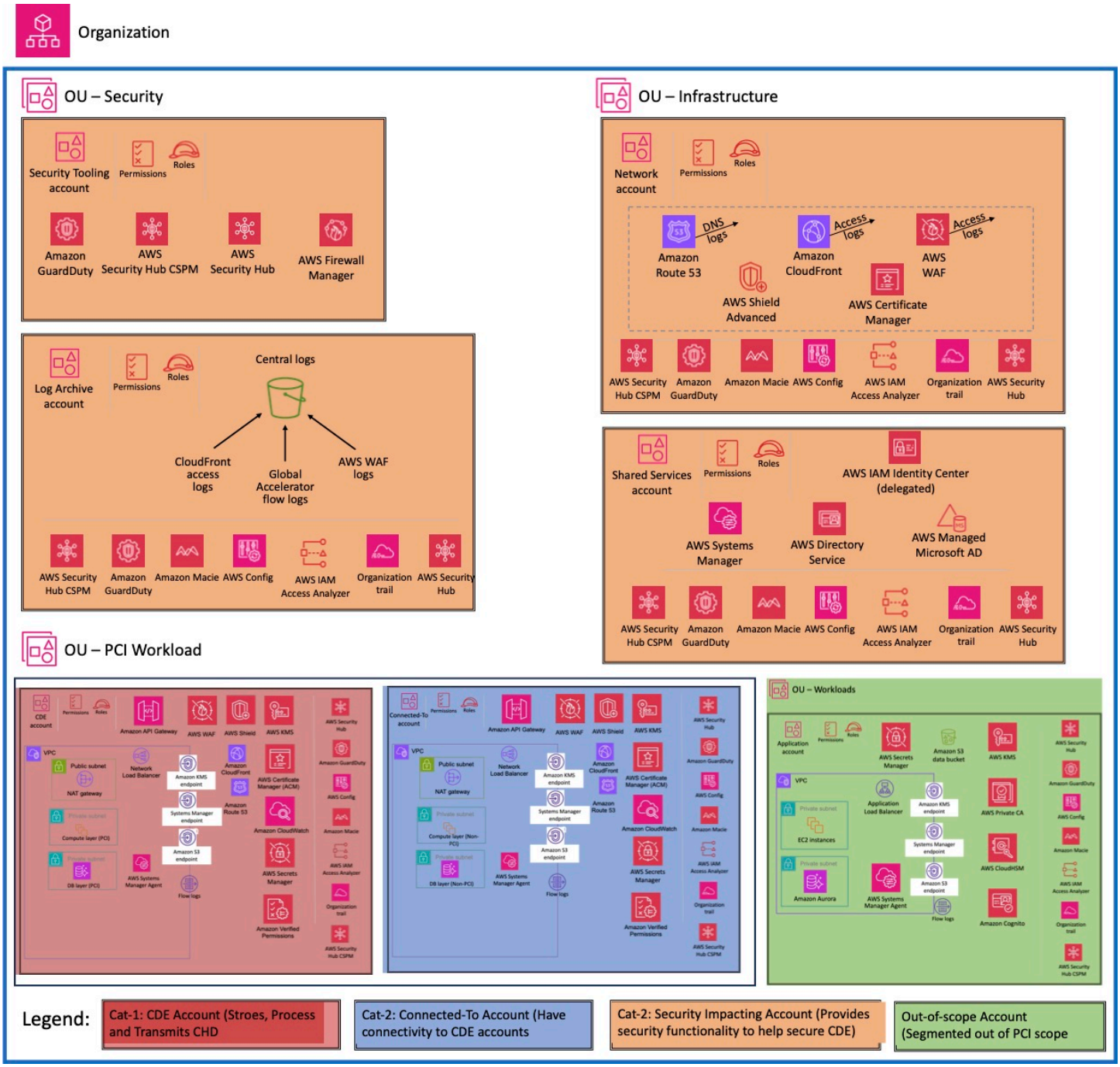
AWS SRA Infrastructure Accounts — Connected-To / Security-Impacting: Several AWS SRA accounts outside the Workloads OU should be included into PCI scope as "security-impacting"

systems. These are AWS accounts that can impact the security configuration of "CDE Accounts" and/Or "Connected-To-Accounts". A primary example of these are accounts that perform Delegated administration, managing AWS service configuration for the entire AWS organization.

AWS SRA Account	Scoping Category	Rationale
Security Tooling	Connected-To (Security-Impacting)	Hosts GuardDuty, Security Hub, Config that monitor CDE accounts
Log Archive	Connected-To (Security-Impacting)	Receives and stores all CDE audit logs; integrity of logs is a PCI DSS Req. 10 control
Network	Connected-To (Security-Impacting)	All CDE ingress/egress transits through this account; Network Firewall rules enforce CDE segmentation
Shared Services	Evaluate Case-by-Case	In-scope if it provides identity (AD/IAM Identity Center), secrets, AWS CloudFormation StackSet delegated administrator, or DNS to CDE accounts; out-of-scope if fully isolated
Org Management	Connected-To (Security-Impacting)	SCPs and organizational governance directly impact CDE account security posture

Non-PCI Workloads OU — Out-of-Scope Accounts: Workloads with no CHD data flow, no connectivity to PCI accounts, and no shared security services with the CDE can be placed in a separate Non-PCI Workloads OU. SCPs on this OU should explicitly deny any action that would create connectivity to PCI accounts (e.g., creating Transit Gateway attachments to PCI route tables, assuming roles in PCI accounts). This SCP-enforced isolation is what allows a QSA to accept these accounts as out of scope.

The following diagram provides a visual representation of different OU types as described in the AWS SRA and their corresponding PCI DSS scope consideration.



Segmentation Boundary Enforcement Across the OU Structure

The power of the AWS SRA's multi-account model for PCI DSS is that segmentation is enforced at multiple independent layers simultaneously — making it resilient to misconfiguration at any single layer:

- Account boundary — No default connectivity between accounts; cross-account access requires explicit IAM trust
- OU-level SCPs — Preventive guardrails that cannot be overridden by account-level administrators
- Network layer — Transit Gateway route tables and Network Firewall rules in the Network account control all inter-account traffic flows
- IAM — Resource-based policies and permission boundaries limit cross-account API access
- Data layer — Account-scoped KMS Customer Managed Keys ensure CDE data cannot be decrypted outside the CDE account. This layered model directly supports the PCI Council's guidance that segmentation must be validated — not just designed — and the AWS whitepaper's recommendation that both network-layer and non-network controls (account boundaries, IAM, SCPs) be documented and presented to QSAs as part of the segmentation evidence package.

Design Consideration: On cloud, logical isolation controls provide a strong segmentation boundary enforcement mechanism. One option is to design a strong Data Perimeter. A data perimeter is a set of permissions guardrails in your AWS environment you use to help ensure that only your trusted identities (PCI In-scope) are accessing trusted resources (PCI In-scope) from expected networks (PCI In-scope). This also helps you prevent your PCI In-scope resources from being accessed by out-of-scope networks/resources.

Inventory

PCI DSS Requirement 12.5.1 mandates that an organization maintain an inventory of all in-scope system components, including a description of function/use for each. Requirement 12.5.2 further requires that this inventory be reviewed at least once every 12 months and updated whenever the environment changes. On AWS, the dynamic, API-driven nature of cloud infrastructure — where resources can be provisioned and deprovisioned in seconds — makes a manual inventory approach both impractical and unreliable. The AWS SRA addresses this through a layered, automated inventory architecture built entirely on native AWS services.

Layer 1 — Continuous Resource Discovery: AWS Config

AWS Config is the foundational service for PCI DSS inventory on AWS. When enabled across all accounts in the organization — which AWS Control Tower enforces automatically upon account enrollment — Config continuously records the configuration state of every supported resource type, generating a Configuration Item (CI) each time a resource is created, modified, or deleted.

For PCI DSS inventory purposes, Config provides three critical capabilities:

- Real-time resource tracking — Config detects and records changes to resources right after they occur, maintaining a point-in-time history of every resource's configuration state. This directly supports the "updated whenever the environment changes" requirement of Requirement 12.5.2.
- Multi-account aggregation — A Config Aggregator deployed in the AWS SRA's Security Tooling (Audit) account collects configuration data from all member accounts and regions into a single pane of glass. This gives the security team a centralized, organization-wide view of all in-scope resources without requiring account-by-account queries.
- Advanced Query — Config's SQL-like query interface allows teams to run structured queries against the current configuration state of all resources across the organization — for example, querying all EC2 instances, RDS databases, Lambda functions, within specific account of VPC. This is directly usable as evidence for a QSA demonstrating inventory completeness.

You can extend this further by pairing the Config Aggregator with Amazon Athena for historical querying and Amazon QuickSight for interactive compliance dashboards — enabling teams to visualize their in-scope resource inventory, track changes over time, and produce audit-ready reports.

Layer 2 — Software and OS-Level Inventory: AWS Systems Manager Inventory, Amazon Inspector

AWS Config tracks infrastructure-level resources (EC2 instances, S3 buckets, RDS databases, etc.), but PCI DSS Requirement 12.5.1 also requires documenting the function and use of each system component — which demands visibility into what is running inside those resources. AWS Systems Manager (SSM) Inventory fills this gap by collecting metadata from managed nodes at configurable intervals (minimum every 30 minutes).

SSM Inventory collects the following metadata types relevant to PCI DSS:

- Applications — Names, publishers, and versions of all installed software (critical for Requirement 6.3 software inventory and vulnerability management)
- Network configuration — IP addresses, MAC addresses, DNS, gateway, and subnet mask per node
- AWS components — EC2 drivers, SSM agent versions, and other AWS-managed software
- Services — Running services, start type, and status (supports identification of unnecessary services per Requirement 2.2.4)

- Windows/Linux updates — Patch status and installed hotfixes (supports Requirement 6.3.3 patching)
- Tags — Tags assigned to each node, enabling PCI scope classification to be reflected in the software inventory
- Custom inventory — Organizations can extend SSM Inventory with custom JSON metadata, for example recording the PCI scoping category, data classification, or owning team for each node

SSM Inventory data can be synced to an S3 bucket via Resource Data Sync and queried using Amazon Athena, enabling cross-account, cross-region software inventory queries. This is particularly valuable for demonstrating to a QSA that all CDE nodes are running only approved, patched software — a direct evidence artifact for Requirements 2.2 and 6.3.

Amazon Inspector SBOM Generator produces SBOM for archives, container images, directories, local systems, and compiled Go and Rust binaries by scanning files that contain information about installed packages. This can be used for Amazon ECR, AWS Lambda and agentless scanning for Amazon EC2. This helps satisfy PCI DSS 4.0 Requirement 6.3.2 mandates that organizations maintain an accurate, up-to-date Software Bill of Materials (SBOM) or software inventory for all bespoke, custom, and third-party software components.

Layer 3 — Tagging Strategy as the Inventory Backbone

None of the above services can distinguish in-scope from out-of-scope resources without a consistent, enforced resource tagging strategy. Tags are the mechanism by which PCI DSS scoping categories are operationalized in AWS — they allow Config queries, SSM Inventory filters, Inspector coverage reports, and Security Hub findings to be scoped to CDE and Connected-To resources specifically.

AWS Organizations Tag Policies enforce consistent tag key naming across all accounts, and AWS Config (e.g., required-tags) can be used to flag any resource that is missing mandatory PCI scope tags — ensuring no resource enters the CDE without being classified.

Recommended Tagging Schema – PCI DSS Inventory

Tag Key	Example Values	Purpose
pci-scope	cde, connected-to, out-of-scope	Identifies PCI DSS scoping category per Req. 12.5.1

data-classification	chd, sad, non-pci	Identifies data type handled by the resource
environment	prod, staging, dev	Supports scope boundary validation
owner	payments-team	Supports accountability and review per Req. 12.5.2
compliance-boundary	pci-cde-prod	Groups resources into named compliance boundaries

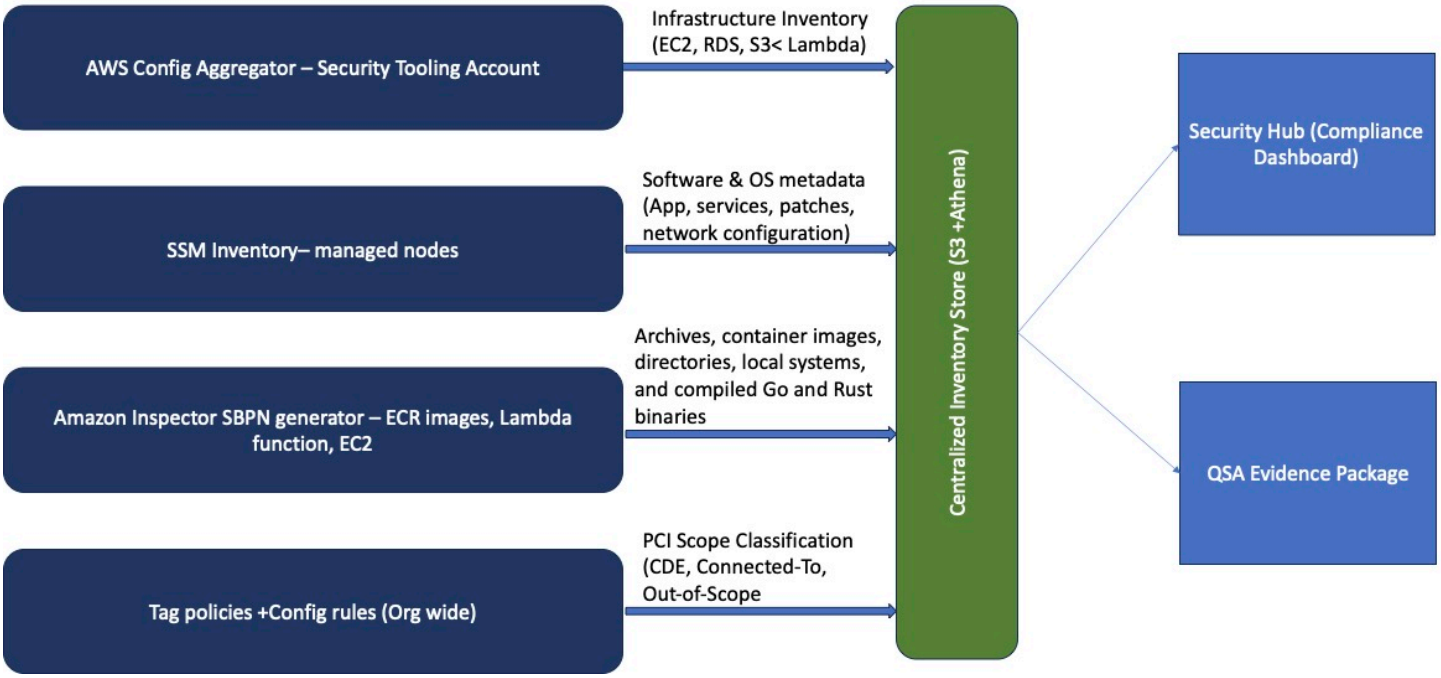
Putting It Together — The Inventory Architecture

The three layers combine into following continuous, automated inventory pipeline:

- AWS Config (with Aggregator) → discovers and tracks all infrastructure-level resources across all accounts in real time
- SSM Inventory → enriches each managed node with software, service, and configuration metadata
- Amazon Inspector → enriches each managed node with software bill of material (SBOM)
- Tag Policies + Config Rules → enforce PCI scope classification on every resource at creation time

Together, these services ensure that the PCI DSS inventory is not a point-in-time spreadsheet exercise but a continuously maintained, automatically updated, and audit-ready record — directly aligned with the intent of PCI DSS 4.0.1's scoping and inventory requirements.

The following diagram depicts the different collections layers for inventory, centralization and output layer architecture.



The flow moves left to right across three stages. On the left, the four collection layers feed in parallel — AWS Config Aggregator discovers all infrastructure resources and change history; SSM Inventory enriches managed nodes with software and OS metadata; Amazon Inspector contributes vulnerability findings and SBOM exports; and Tag Policies with Config Rules enforce PCI scope classification across the organization. In the center, all streams converge into the Centralized Inventory Store (S3 + Athena), which acts as the single queryable source of truth for the complete PCI DSS asset inventory. On the right, the inventory store feeds two outputs —Security Hub for the live compliance dashboard, and the QSA Evidence Package that directly satisfies Requirements 12.5.1,12.5.2 and 6.3.2.

Network Security Controls and Network Intrusion

Network security controls form a key line of defense in protecting cardholder data environments from unauthorized access and malicious activity. PCI DSS Requirement 1 mandates the installation and maintenance of network security controls to restrict network access to and from the cardholder data environment (CDE), while Requirement 11.5.1 requires the prevention/detection and response to network intrusions. These requirements are foundational to PCI DSS compliance because network boundaries define the scope of the CDE, and effective network intrusion capabilities enables organizations to identify and respond to security incidents before they result in data compromise. In cloud environments, software-defined networking and managed security services provide opportunities to implement these controls with greater automation, consistency, and scalability.

Network Security Controls and Network Intrusion within the AWS SRA

The AWS SRA provides foundational guidance for network security controls and intrusion detection that directly supports key elements of PCI DSS:

Use Amazon VPC with security groups and network ACLs to implement network segmentation and restrict access to the CDE. Amazon VPC provides logically isolated environments enabling network security functions and boundary definition, while security groups provide stateful filtering at the resource level and network ACLs provide stateless filtering at the subnet level, creating defense-in-depth network controls. Together, these services support Requirements 1.3 and 1.4 for restricting inbound and outbound traffic between trusted and untrusted networks. You are responsible for configuring security group rules and network ACL rules to permit only necessary traffic to and from the CDE.

Leverage AWS Firewall Manager within the network account to centrally manage security group rules, network ACL rules, and AWS Network Firewall policies across multiple accounts in AWS Organizations. Firewall Manager can identify unused, redundant, or overly permissive security groups and enforce organizational policies that define which security groups are allowed or disallowed, supporting Requirements 1.2.7 and 1.2.8 for semi-annual reviews of network security control rule sets and configurations.

Enable Amazon GuardDuty to manage threat detection by analyzing VPC Flow Logs, AWS CloudTrail event logs, and DNS logs to identify malicious or unauthorized behavior. GuardDuty uses machine learning models, anomaly detection, and integrated threat intelligence to detect threats such as compromised instances, reconnaissance activities, and command-and-control communications, addressing Requirement 11.5.1 for intrusion detection. You are responsible for enabling GuardDuty in all relevant AWS accounts and configuring alerting mechanisms such as Amazon EventBridge rules to notify personnel of suspected compromises.

Integrate VPC Flow Logs with Amazon CloudWatch Logs to capture and analyze network traffic metadata at the VPC, subnet, or network interface level. Flow Logs provide visibility into traffic patterns, source and destination IP addresses, ports, and protocols, supporting both network security control validation and intrusion detection activities (Requirements 1.2.7 and 11.5.1). You can use Amazon CloudWatch Logs Insights or Amazon Athena to query and analyze flow log data for security investigations.

Add Config Rules in AWS Config to continuously monitor network security control configurations and detect changes/drifts from approved network control baselines. Config Rules can validate that security groups do not permit unrestricted access (0.0.0.0/0) on sensitive ports, that network ACLs are configured according to organizational standards, and that VPC configurations align with documented network diagrams, supporting Requirements 1.2.7 and 1.2.8 for configuration reviews.

Deploy AWS Network Firewall in a centralized network account to provide stateful inspection, intrusion prevention, and web filtering at the VPC perimeter. Network Firewall uses the Suricata open-source intrusion prevention engine for stateful inspection and can be configured with domain filtering, intrusion prevention rules, and stateful protocol detection, supporting Requirements 1.3, 1.4, and 11.5.1. AWS manages the underlying infrastructure, while you are responsible for defining firewall policies, rule groups, and logging configurations.

Network Security Controls and Network Intrusion beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

Automate network diagram generation by deploying Amazon Q CLI, combined with the AWS Diagram MCP server and AWS Documentation MCP server, which automatically incorporating AWS

service icons, VPC boundaries, security groups, network segmentation zones (untrusted, DMZ, internal), and data flow paths. This AI-driven, diagram-as-code approach substantially reduces the manual effort traditionally required to create and maintain these PCI-mandated diagrams, supports iterative refinement as environments evolve, and helps ensure diagrams remain current and consistent. By leveraging this tooling, organizations can integrate diagram generation into their CI/CD or compliance workflows, reducing documentation drift and providing audit-ready evidence that demonstrates ongoing compliance with PCI DSS Requirements 1.2.3 and 1.2.4 for maintaining accurate network diagrams and data flow diagrams.

Implement automated data flow documentation by using AWS CloudTrail and VPC Flow Logs as data sources, with Amazon OpenSearch Ingestion pipelines to continuously ingest, transform, and index flow log data into structured data flow indexes. The indexed data documents source, destination, transport protocols, encryption methods, and access controls for each data flow. Leverage OpenSearch Dashboards to visualize network flows in near real-time providing continuous, audit-ready evidence of account data flows supporting Requirement 1.2.4 for accurate data flow diagrams.

Deploy centralized network security control review automation by configuring leveraging AWS Config conformance packs with rules that validate network security control configurations against PCI DSS requirements and . Deploy AWS Security Hub CSPM with the PCI DSS security standard enabled to aggregate findings across accounts. Create Amazon EventBridge rules monitoring for network resource changes and compliance status changes via AWS Config, AWS Security Hub CSPM, and AWS Network Firewall that trigger AWS Lambda functions to generate semi-annual network security control review reports, documenting all security group rules, network ACL rules, and firewall policies with business justifications, supporting Requirements 1.2.7 and 1.2.8.

Enable automated intrusion detection alerting by configuring Amazon GuardDuty findings to publish to Amazon EventBridge, which triggers AWS Lambda functions or Amazon SNS topics to alert security personnel of suspected compromises. Integrate GuardDuty with AWS Security Hub to correlate intrusion detection findings with other security alerts. Configure Amazon CloudWatch alarms to monitor GuardDuty finding counts and severity levels, ensuring personnel are promptly alerted to suspected network intrusions (Requirement 11.5.1).

Implement advanced network traffic inspection by deploying VPC Traffic Mirroring to copy network traffic from critical CDE resources to Amazon EC2 instances running third-party intrusion detection appliances from AWS Marketplace. Alternatively, deploy AWS Network Firewall with intrusion prevention rule groups enabled, configuring Suricata-compatible rules to detect and prevent known attack patterns. Configure firewall logs forward to Amazon S3 and Amazon CloudWatch

Logs for analysis, supporting Requirement 11.5.1 for monitoring all traffic at the perimeter and critical points of the CDE.

Deploy network access path analysis by configuring Amazon VPC Network Access Analyzer to continuously analyze network paths and validate that CDE resources are properly segmented from out-of-scope environments. Create AWS Config custom rules that leverage Network Access Analyzer findings to detect when network paths violate segmentation requirements, triggering Amazon EventBridge rules that alert security teams and initiate remediation workflows, supporting Requirements 1.3 and 1.4.

System Hardening and Secure Configuration

System hardening and secure configuration establish the foundational security posture of all system components within the cardholder data environment. PCI DSS Requirement 2 mandates applying secure configurations to all system components, ensuring that vendor defaults are changed, unnecessary services are disabled, and security parameters are configured to prevent misuse. In cloud environments, infrastructure-as-code, automated configuration management, and continuous compliance monitoring provide opportunities to implement and maintain secure configurations with greater consistency, auditability, and scalability than traditional manual configuration approaches.

System Hardening and Secure Configuration within the AWS SRA

The AWS SRA provides foundational guidance for system hardening that directly supports key elements of PCI DSS:

Using AWS Systems Manager State Manager and AWS Config together enables automated enforcement and validation of configuration standards across all system components. State Manager can apply and maintain desired configurations, while Config Rules continuously verify compliance with defined standards, satisfying Requirements 2.2.1 and 2.2.6 for configuration standards that are consistent with industry-accepted hardening guidelines and system security parameters configured to prevent misuse.

AWS Config Conformance Packs enable deployment of pre-built collections of Config Rules aligned with industry frameworks such as CIS Benchmarks, NIST guidelines, and PCI DSS requirements, providing automated validation that system components are configured according to industry-accepted hardening standards (Requirement 2.2.1).

Using AWS Systems Manager Session Manager for administrative access eliminates the need for bastion hosts, SSH keys, or open inbound ports, while enforcing encrypted connections (HTTPS) for all administrative functions, directly supporting Requirement 2.2.7 for encrypted non-console administrative access.

AWS Security Hub security standards automatically assess resources against CIS AWS Foundations Benchmark, AWS Foundational Security Best Practices, and the PCI DSS security standard,

providing continuous validation that configurations align with industry-accepted hardening standards and identifying configuration drift from secure baselines (Requirement 2.2.1).

Hardened AMIs from AWS Marketplace provide pre-configured baseline images that have been hardened by security professionals according to CIS Benchmarks and vendor hardening recommendations, reducing the time and effort required to implement secure configurations and ensuring consistency across deployments (Requirement 2.2.1).

AWS CloudFormation and AWS CDK enable organizations to define hardened configurations as code, ensuring that every deployment follows approved configuration standards and eliminating manual configuration errors. Templates can be version-controlled, peer-reviewed, and validated before deployment, supporting consistent application of configuration standards across all system components (Requirement 2.2.1).

AWS Organizations SCPs enforce preventive guardrails across all accounts in the organization, ensuring that insecure configurations cannot be deployed regardless of individual account-level permissions. SCPs can restrict the use of unapproved services, enforce encryption requirements, and limit deployments to authorized regions, supporting consistent application of security parameters across the environment (Requirements 2.2.1 and 2.2.6). Service control policy examples - This GitHub repository contains example policies to get started or mature your usage of AWS SCPs

System Hardening and Secure Configuration beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements.

Automate vendor default account remediation: Deploy AWS Systems Manager Automation documents and Run Command to systematically identify and remediate vendor default accounts on EC2 instances and other system components at launch. Combine with AWS Config to continuously detect instances running with known default credentials or default account configurations, triggering automated remediation workflows to change default passwords and disable unnecessary default accounts (Requirement 2.2.2).

Enforce function isolation through deployment architecture: PCI DSS Requirement 2.2.3 mandates that each system component performs only one primary function, or that functions with differing security levels are isolated from one another, with the intent of preventing a compromise

of one function from providing unauthorized access to another. AWS SRA already prescribes AWS CloudFormation, AWS CDK, AWS Config, and security groups as foundational components across the multi account environment. Extending the AWS SRA for Requirement 2.2.3 involves applying these existing AWS SRA prescribed services in a pattern that enforces function isolation through three complementary controls.

- Infrastructure as Code: The AWS SRA prescribes AWS CloudFormation and AWS CDK for consistent infrastructure deployment across accounts. Extend this by defining deployment templates that assign a single primary function to each system component. CloudFormation stacks and CDK constructs structurally separate each function into its own deployment unit with dedicated IAM roles, network configurations, and security groups so that the only permitted communication paths between functions are those explicitly defined in the template. Codifying these patterns ensures every deployment is consistent, repeatable, and auditable, preventing unauthorized addition of secondary functions to any component. Because templates are version controlled and peer reviewed, any deviation from the approved isolation architecture requires a deliberate, auditable change.
- EC2 isolation: Different EC2 instances on the same physical host are isolated from each other as though they are on separate physical hosts. The hypervisor isolates CPU and memory, and the instances are provided with virtualized disks instead of accessing the raw disk devices. Additionally for Windows host you can use Windows Firewall settings with Group Policy Objects (GPO) to block specific applications from accessing the network, ensuring only permitted functionality is allowed on the host.
- AWS Lambda isolation: Lambda functions provide tenant isolation mode to serve multiple end-users or tenants using a single Lambda function, while ensuring that the execution environments used to process invocations for individual tenants remain isolated from one another. You can also use AWS Lambda Micro VM serverless compute environments that provide VM-level isolation with full operating system capabilities (for e.g. installing system packages or mounting filesystems), snapshot-based rapid startup speeds, and fine-grained control over ingress networking (port access, support for HTTP/2, gRPC, WebSockets) and egress networking (public internet access and VPC access).
- Container Based Function Isolation: The AWS SRA multi account structure provides dedicated accounts for workloads with distinct security requirements. Within those accounts, deploy each function (web server, application logic, database) as a separate container using Amazon ECS task definitions or Amazon EKS pod specifications. Each container operates independently with its own runtime boundary, dedicated IAM task role scoped to least privilege, and isolated resource limits. This extends the AWS SRA account level isolation principle down to the workload

layer, limiting the blast radius if any single component is compromised and preventing lateral movement between functions.

- Network Level Segmentation: The AWS SRA prescribes security groups and VPC architecture as core network controls. Extend these by applying security groups that enforce network level isolation between containers serving different primary functions. Each security group permits only the minimum required ports and protocols (e.g., the application container can reach the database container on port 5432, but the web container cannot), ensuring components cannot communicate beyond their defined trust boundaries. Deploy AWS Config, already prescribed by the AWS SRA for continuous compliance monitoring, to validate that deployed security groups maintain the intended isolation boundaries and detect any unauthorized changes to permitted communication paths (Requirement 2.2.3).

Automate removal of unnecessary services and functions: Use AWS Systems Manager State Manager associations to continuously enforce desired-state configurations that disable unnecessary services, protocols, daemons, and functions on EC2 instances. Deploy AWS Config to detect system components running unauthorized services or protocols, and trigger AWS Systems Manager Automation remediation to disable them. For any insecure services that must remain enabled, use AWS Config to validate that additional security controls such as encryption and enhanced logging are in place (Requirements 2.2.4 and 2.2.5).

Implement automated security parameter validation and remediation: Deploy AWS Config, including custom rules where needed, to validate that security parameters are properly configured across all system components, covering authentication, authorization, logging, encryption, and network access control settings. Pair Config Rules with AWS Systems Manager Automation documents to automatically remediate parameter drift, ensuring security parameters remain configured to prevent misuse. Deploy AWS Network Firewall to enforce stateful inspection and protocol level security parameters at the VPC perimeter, ensuring only approved protocols and traffic patterns are permitted into and out of the cardholder data environment. Use AWS Firewall Manager to centrally define and apply Network Firewall policies, security group rules, and WAF configurations across all accounts in the organization, ensuring consistent enforcement of network security parameters and automatically remediating resources that drift from approved baselines (Requirement 2.2.6).

Enforce and audit encrypted administrative access: Configure AWS Systems Manager Session Manager with session encryption and logging enabled, forwarding session logs to Amazon S3 and Amazon CloudWatch Logs for audit purposes. Deploy AWS Config to detect security groups

that permit inbound SSH (port 22) or RDP (port 3389) access, and use AWS Systems Manager Automation to automatically revoke non-compliant rules. Enforce AWS IAM policies requiring MFA for all administrative actions (Requirement 2.2.7).

Implement continuous configuration change detection and response: Integrate AWS Config with Amazon EventBridge and AWS CloudTrail to detect and alert on configuration changes in real time. Deploy EventBridge rules that trigger AWS Lambda functions or AWS Systems Manager Automation documents to evaluate changes against approved baselines and automatically remediate unauthorized modifications, ensuring configuration standards are verified as in place before or immediately after system components connect to a production environment (Requirement 2.2.1).

Encryption and Key Management

Encryption and key management form the final defense control in protecting cardholder data from unauthorized disclosure, ensuring that even if other security controls fail, the data remains unreadable to unauthorized identities. PCI DSS Requirement 3 mandates protecting stored account data through strong cryptography, while Requirement 4 requires protecting cardholder data with strong cryptography during transmission over open, public networks. The effectiveness of encryption, however, depends entirely on proper key management—cryptographic keys must be generated, stored, distributed, rotated, and destroyed using processes that prevent unauthorized access to key material. In cloud environments, managed encryption services, and automated key lifecycle management provide opportunities to implement cryptographic controls and key management practices with greater security, operational efficiency, and compliance auditability than traditional on-premises key management approaches.

Encryption and Key Management within the AWS SRA

The AWS SRA provides foundational guidance for encryption and key management that directly supports key elements of PCI DSS:

Using AWS KMS with customer-managed keys for encryption at rest allows customers to render stored account data unreadable, addressing Requirements 3.5.1 and 3.5.1.1.

For Managed Services: When customers use AWS managed services such as Amazon RDS or Amazon S3 with customer managed keys, encryption is applied above the disk or partition level at the application layer running on top of the abstracted infrastructure. This distinction is significant for Requirement 3.5.1.2, With customer managed keys, access to the data and access to the encryption key are separated: a user or role must have discrete authorization for the AWS service and then separate permissions to use the encryption key in KMS. This separation of access addresses the intent of Requirement 3.5.1.2 for managed services. Additionally, the customer retains full control over the key lifecycle, they can disable the key, revoke access through key policy or grant changes, or schedule key deletion at any time, immediately rendering all associated data unreadable.

For Storage Services: For Storage Mediums such as Amazon EBS, Amazon EFS, and Amazon FSx, which are the virtual equivalent of physical disks presented to the customer, KMS encryption alone may not satisfy Requirement 3.5.1.2. Once these volumes are mounted or attached, all data is

accessible in decrypted form to anyone with operating system level access, which is functionally equivalent to disk level encryption where access equals decryption. If customers store cardholder data on these storage solutions, additional data level encryption at the file, folder, or application level is needed to ensure that access to the storage medium does not automatically provide access to cleartext PAN (Requirements 3.5.1 and 3.5.1.2).

The AWS SRA illustrates the recommended distribution model for key management, where the AWS KMS key resides within the same AWS account as the resource to be encrypted. AWS KMS manages the complete key lifecycle within its backed HSMs, where symmetric key material cannot be exported in plaintext. For customer managed keys, customers can configure automatic rotation with a customizable period, or invoke on-demand rotation at any time. AWS is responsible for the physical security and logical protection of key material within its HSM infrastructure (supporting Requirements 3.6.1.2 and 3.6.1.4), while customers retain responsibility for defining key policies, managing access grants, and controlling which principals and services can use each key. KMS supports key generation within its HSMs or import of customer-provided key material (BYOK), secure distribution of data keys within the service, and cryptographic erasure of key material when deletion is initiated by customers. The default symmetric key spec (SYMMETRIC_DEFAULT) is AES-256-GCM. However, KMS also supports asymmetric keys (RSA 2048/3072/4096, ECC) and HMAC keys. For symmetric encryption keys used for data at rest, AES-256 is the only option. (Requirements 3.6 and 3.7)

Both AWS KMS and AWS CloudHSM support generation and tracking of cryptographic keys, enabling customers to maintain inventories of keys in use including algorithms, key strength, key custodians, and cryptoperiods, supporting compliance with Requirement 3.6.1.1.

AWS CloudTrail logs all KMS and CloudHSM API calls, providing a complete audit trail of key management activities — including key creation, rotation, usage, policy changes, and deletion — supporting the auditability requirements across Requirements 3.6 and 3.7.

Design Consideration: Use AWS KMS for native AWS integrations and a fully managed, cost-effective key management system. Use AWS CloudHSM when you need dedicated, single-tenant hardware, the ability to export key material, or compliance mandates that prohibit the cloud provider from possessing logical access to your keys. For more detailed design decision refer to AWS cryptography services decision guide.

AWS Certificate Manager provisions and manages SSL/TLS certificates for services such as Amazon CloudFront, Amazon API Gateway, and Elastic Load Balancers, and enables customers to maintain an inventory of certificates in use with relevant metadata including issuing certificate authority and

expiration dates, supporting Requirement 4.2.1.1 for maintaining an inventory of trusted keys and certificates. (Requirement 4.2.1.1)

AWS Secrets Manager supports secure storage and automated rotation of SSL/TLS certificates used to protect cardholder data in transit. When certificates are stored in Secrets Manager, rotation is managed through a custom AWS Lambda rotation function that handles certificate renewal and deployment to target resources, as the default managed rotation feature does not support certificate rotation. This capability complements AWS Certificate Manager for certificates that cannot be managed directly through ACM, such as certificates used by Amazon EMR or other services requiring custom certificate deployment, supporting Requirement 4.2.1.1 for maintaining an inventory of trusted keys and certificates..

VPC endpoints powered by AWS PrivateLink enable private connectivity between VPCs and supported AWS services, ensuring traffic remains on the AWS network backbone and does not traverse open, public networks. Because Requirement 4.2 specifically applies to PAN transmitted over open, public networks, traffic routed through PrivateLink endpoints to AWS services can be considered out of scope for Requirement 4.2 encryption requirements — though this scoping determination should be documented in the network segmentation documentation and validated with the QSA during the assessment. (Requirement 4)

Externally exposed AWS services such as Amazon CloudFront, Amazon API Gateway, and Elastic Load Balancers support transport encryption of TLS 1.2 or greater with configurable security policies to enforce strong cryptography during transmission over open, public networks. (Requirement 4.2.1)

Encryption and Key Management beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

AWS Database Encryption SDK to implement attribute level encryption: Deploy client side encryption using the AWS Database Encryption SDK to implement attribute level encryption for DynamoDB tables storing cardholder data, ensuring PAN is encrypted at the application layer before being written to the database, providing defense in depth independent of service level encryption (Requirement 3.5.1). You can also use Amazon Relational Database Service (Amazon RDS) for SQL Server support for column-level data encryption. Column-level encryption provides encryption at a more granular level of data that can be applied on all or selected columns. With

column-level encryption, you can define different encryption keys for each different column. Refer to 'Column-level encryption on Amazon RDS for SQL Server for more details on column level encryption.

Enable automatic key rotation for KMS customer-managed keys: Configure annual automatic rotation for KMS keys used to encrypt stored account data to satisfy cryptoperiod requirements. Use AWS Config to detect KMS keys that do not have automatic rotation enabled and trigger AWS Systems Manager Automation to remediate non-compliant keys. Customers are responsible for defining cryptoperiods appropriate to their environment and key usage. (Requirements 3.7.4)

Configure encryption by default for all storage services: Enable Amazon S3 default bucket encryption with SSE-KMS using customer-managed keys, enable Amazon RDS encryption during instance creation, and enable Amazon EBS encryption by default in account settings. Deploy AWS Config to detect unencrypted storage resources and trigger automated remediation to ensure all stored account data is rendered unreadable. (Requirement 3.5)

Deploy Amazon Macie for SAD and PAN discovery and data retention monitoring: Enable Amazon Macie across S3 buckets in the cardholder data environment to continuously discover and classify stored PAN. Use Macie findings to identify unauthorized PAN storage locations and support data retention minimization by flagging data that exceeds defined retention periods for automated or manual purging. (Requirement 3.2.1 and 12.10.7)

Implement application-level PAN masking and truncation: While AWS services provide infrastructure-level encryption, customers are responsible for implementing application-level controls to render PAN unreadable when displayed. Use AWS Lambda functions to implement masking (displaying only first six and last four digits) or truncation when PAN must be displayed, ensuring full PAN is never displayed except for users with a legitimate business need (Requirement 3.4.1).

Enforce TLS 1.2 minimum for all data in transit: Configure Application Load Balancer security policies, Amazon CloudFront distributions, and Amazon API Gateway stages to require TLS 1.2 . It is recommended to use TLS 1.3 where applicable as it provides support for Post Quantum cryptography. The Elastic Load Balancing (ELB) new post-quantum TLS (PQ-TLS) based security policy#ELBSecurityPolicy-TLS13-1-2-Res-PQ-2025-09 or ELBSecurityPolicy-TLS13-1-2-Res-FIPS-PQ-2025-09 .Deploy AWS Config to detect load balancers, CloudFront distributions, or API Gateway stages configured with security policies that permit TLS versions below 1.3, and trigger automated remediation. Configure AWS SDKs to enforce TLS 1.3 or higher when accessing public AWS service endpoints. (Requirement 4.2.1)

Implement certificate lifecycle management and inventory: Use AWS Certificate Manager to automate certificate renewal and deploy AWS Config to maintain a current inventory of all trusted keys and certificates used to protect PAN during transmission, including issuing certificate authority and expiration dates, enabling rapid response to discovered vulnerabilities in cryptographic protocols. (Requirement 4.2.1.1)

> **Note**
>
> AWS Payment Cryptography is not discussed in this section as its capabilities are primarily relevant to payment service providers, processors, issuers, and acquirers that perform payment-specific cryptographic operations such as PIN translation, CVV generation and verification, and P2PE decryption rather than merchants whose PCI DSS scope is limited to protecting stored account data and data in transit. Service providers performing these payment processing functions may find AWS Payment Cryptography helpful in meeting PCI PIN Security and PCI P2PE requirements. Refer to the following AWS documentation for guidance on leveraging AWS Payment Cryptography for PCI compliance: PCI PIN and P2PE compliance packages for AWS Payment Cryptography are now available Deploy Amazon S3 Lifecycle Policies and Amazon DynamoDB TTL to automatically delete temporary data stores that may contain sensitive authentication data after processing is complete. (Requirements 3.2.1, 3.3.1, 3.3.2, and 3.3.3)

Secure Software Development Lifecycle (SDLC) and CI/CD

The Software Development Lifecycle (SDLC) is a foundational control area aimed at ensuring that security is embedded into every phase of software development that processes, stores, or transmits cardholder data. PCI DSS requires organizations to define, implement, and maintain a structured development process that integrates security requirements from design through deployment and maintenance. This includes secure coding standards, threat modeling, code review practices, vulnerability management, and security testing such as static and dynamic analysis. The goal is to prevent common software vulnerabilities (e.g., injection flaws, insecure authentication, and misconfigurations) before applications are released into production environments.

In a cloud environment, the Software Development Lifecycle (SDLC) under PCI Security Standards Council means applying the same security-by-design principles but extending them across both application code and cloud infrastructure configuration. Instead of only securing traditional software artifacts, SDLC in cloud also includes securing Infrastructure as Code (IaC), CI/CD pipelines, container images, APIs, and cloud service configurations that interact with cardholder data environments (CDE).

SDLC and CI/CD within the AWS SRA

The AWS SRA provides foundational guidance for SDLC and CI/CD that directly supports key elements of PCI DSS:

Amazon Inspector in Security Tooling account is included as a vulnerability management service that continuously scans your AWS workloads for vulnerabilities and directly supports the PCI DSS requirement to identify and address security vulnerabilities in bespoke and custom software (Requirement 6.3.1). Inspector automatically rescans workloads when new packages are installed, patches are applied, or new CVEs are published, ensuring that known vulnerabilities are identified promptly. Inspector generates risk scores correlated with network accessibility and exploitability data, enabling prioritized remediation aligned with the risk-ranking requirement (Requirement 6.3.1). Amazon Inspector Code Security extends this capability into the development phase by performing SAST and SCA on source code repositories integrated with GitHub, providing two-way integration that suggests fixes as pull request comments. This supports the requirement to identify and address vulnerabilities during the software development process (Requirement 6.2.4).

AWS Identity and Access Management Access Analyzer in security tooling account is references as a security tool to efficiently set fine-grained permissions, verify intended permissions, and refine permissions by removing unused access, while also enabling automated IAM policy checks earlier in the development and deployment (CI/CD) process to adhere to corporate security standards, supporting the requirement that bespoke and custom software is developed securely with appropriate access controls (Requirement 6.2.4). The AWS SRA deploys IAM Access Analyzer at both the organization level (Security Tooling account) and account level (all member accounts), providing comprehensive policy validation coverage. Organizations can integrate IAM Access Analyzer custom policy checks with AWS CloudFormation to automate policy reviews within CI/CD pipelines.

SDLC and CI/CD beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

Deploy Amazon Inspector Code Security for SAST/SCA in CI/CD pipelines: Configure Amazon Inspector CI/CD integration plugins for Jenkins, TeamCity, or equivalent build tools to assess container images and application code for software vulnerabilities during the build phase (Requirement 6.2.4). Inspector's CI/CD plugin can block builds or image pushes when critical vulnerabilities are detected, enforcing security gates before code reaches production. This does not require activating the full Amazon Inspector service and operates on AWS, on-premises, or hybrid environments. Configure scan triggers to execute on code changes, at scheduled intervals, and on demand. Integrate Inspector Code Security with GitHub repositories to enable SAST, SCA, and IaC scanning with automated pull request comments for identified vulnerabilities (Requirement 6.2.4). This addresses the requirement that bespoke and custom software is developed securely by identifying vulnerabilities before deployment.

Deploy AWS DevOps Agent for PCI-DSS Aligned SDLC and CI/CD Security Automation: An AWS DevOps Agent can play a key role in supporting a PCI DSS-aligned Secure Software Development Lifecycle (SSDLC) and CI/CD security reference architecture by acting as an automated enforcement and advisory layer across development pipelines. Within the context of PCI Security Standards Council requirements, the agent can continuously evaluate infrastructure-as-code templates, application code, and deployment configurations against predefined security baselines such as secure IAM policies, encryption standards, logging requirements, and network segmentation rules. It can integrate directly into CI/CD workflows to block or flag non-compliant builds before deployment, ensuring that security controls are enforced "shift-left" during development rather

than validated after release. Additionally, the agent can correlate pipeline events with runtime security signals from tools like AWS Security Hub and AWS-native security services, enabling post-deployment verification that critical controls remain operational after each change. This approach supports continuous compliance, reduces manual audit overhead, and helps maintain a resilient, PCI DSS-compliant software delivery lifecycle in cloud-native environments.

Deploy AWS WAF with managed rule groups for OWASP Top 10 protection: Aligned with AWS SRA recommendation, to meet PCI DSS 6.4.1 requirement, configure AWS WAF web ACLs with AWS Managed Rules for common vulnerabilities including SQL injection, cross-site scripting (XSS), local file inclusion, and remote code execution to protect public-facing web applications that are part of the CDE. Enable the AWS WAF Bot Control managed rule group to detect and manage automated threats. Configure AWS Firewall Manager in the Security Tooling account to enforce consistent WAF policies across all Application accounts containing public-facing payment applications. Enable comprehensive WAF logging to Amazon S3 in the Log Archive account and configure Amazon CloudWatch metrics and alarms for anomalous traffic patterns. For organizations using the customized approach, deploy AWS WAF with custom rules that implement automated technical solutions to continuously detect and prevent web-based attacks specific to the application's attack surface (Requirement 6.4.2).

Implement script integrity management for payment page security: Configure Amazon CloudFront response headers policies to implement Content-Security-Policy (CSP) headers that restrict script execution sources on payment pages (Requirement 6.4.3). Deploy AWS WAF custom rules to monitor and block unauthorized script injection attempts on pages that render or process payment information. Use Amazon CloudWatch and Amazon EventBridge to alert on CSP violation reports, enabling detection of unauthorized script modifications. Note: AWS services support the technical enforcement mechanisms, but the customer retains responsibility for maintaining the script inventory, documenting justifications, and implementing the authorization process for script changes.

Deploy Amazon Inspector for vulnerability scanning of bespoke software: Enable Amazon Inspector network reachability assessments to identify unintended network paths to EC2 instances hosting bespoke applications within the CDE (Requirement 6.3.2). Configure Inspector to scan Lambda functions and layers for vulnerabilities in serverless application components. Enable CIS Benchmark assessments for operating system-level security validation on instances running custom applications. Integrate Inspector findings with AWS Systems Manager Automation for automated remediation of identified vulnerabilities, creating a continuous feedback loop between vulnerability detection and resolution.

Change Management

Change management is a foundational security and governance process that ensures all changes to systems, applications, infrastructure, and configurations are formally reviewed, tested, approved, implemented, and documented in a controlled manner. The objective of change management within PCI DSS is to prevent unauthorized or poorly managed changes from introducing security vulnerabilities, disrupting services, or compromising the confidentiality and integrity of the cardholder data environment (CDE). Effective change management provides traceability and accountability by requiring organizations to assess security impacts, obtain approvals, perform testing, maintain rollback procedures, and document implementation activities to ensure compliance and operational stability.

Change Management within the AWS SRA

The AWS SRA provides foundational guidance for change management that directly supports key elements of PCI DSS:

AWS Config in security tooling account can be used to assess, audit, and evaluate the configurations of supported AWS resources in AWS accounts. AWS config provides the foundational change detection and configuration compliance mechanism prescribed by the AWS SRA. Using AWS Config in proactive mode, organizations can evaluate resource configurations before deployment, preventing non-compliant changes from reaching production environments (Requirement 6.5.1). In detective mode, Config continuously evaluates deployed configurations against defined rules, identifying unauthorized or unintended changes. Config integrates with AWS Systems Manager Automation runbooks for automated remediation of non-compliant configurations, creating a closed-loop change validation system (Requirement 6.5.2). The AWS SRA deploys Config across all accounts with the Security Tooling account serving as delegated administrator, providing organization-wide visibility into configuration changes.

AWS CloudTrail in security tooling account is a service that supports governance, compliance, and auditing of activity for AWS account, providing monitoring through an organization trail that logs all events for all AWS accounts in that AWS organization. AWS CloudTrail records all API calls across the organization trail, providing the audit trail necessary to demonstrate that all changes to system components follow established change control procedures (Requirement 6.5.1). CloudTrail logs capture who made changes, what was changed, when changes occurred, and from where, supporting the documentation requirements for change records. Logs are delivered to the centralized Log Archive account with integrity validation enabled.

AWS Organizations is referenced as AWS SRA building block which aids in centrally managing and governing the environment as additional AWS resources are deployed, while also providing several policies that enable centrally applying additional security controls to all member accounts in the organization. Service Control Policies (SCPs) deployed through AWS Organizations provide the structural foundation for separating production and non-production environments, as required by PCI DSS Requirement 6.5.2. The AWS SRA prescribes distinct organizational units for different environment types, with SCPs enforcing preventive guardrails that restrict which APIs can be called and prevent disabling of security services. SCPs are applied from the Org Management account and inherited by all member accounts, ensuring that environment separation controls cannot be overridden at the account level.

AWS Systems Manager in infrastructure account, is referenced as a network and application protection service which provides a collection of capabilities that enable visibility and control of AWS resources while supporting a number of preventive, detective, and responsive security features. AWS Systems Manager Session manager is a feature of AWS Systems Manager which provides secure, auditable remote access to EC2 instances without opening inbound SSH ports, managing SSH keys, or deploying bastion hosts (Requirement 6.5.1 Changes to all system components in the production environment are made according to established procedures that include: 1/ reason for, and description of, the change, 2/ documentation of security impact, 3/ documented change approval by authorized parties, and 4/ testing to verify that the change does not adversely impact system security.) All sessions are logged and auditable, supporting the requirement 6.4 that production data and accounts are not used in pre-production environments and that access to production systems follows controlled processes. Session Manager operates through VPC endpoints, maintaining private connectivity.

Change Management beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

Configure AWS Config for change control validation: Deploy custom AWS Config and conformance packs that validate all changes to in-scope system components comply with established change control requirements (Requirement 6.5.1). Create Config rules that detect unauthorized modifications to security-relevant configurations including security group rules, IAM policies, encryption settings, and network ACLs. Pair detective Config rules with AWS Systems Manager Automation runbooks to automatically revert unauthorized changes or quarantine non-compliant resources. Route Config compliance change events through Amazon EventBridge

to notify security teams and create audit records in Amazon CloudWatch Logs and Amazon S3 (Requirement 6.5.2). Deploy proactive Config rules using AWS CloudFormation Guard within deployment pipelines to evaluate resource configurations before provisioning, preventing non-compliant changes from being deployed to production.

Configure IAM Access Analyzer custom policy checks in deployment pipelines: Integrate IAM Access Analyzer custom policy checks with AWS CloudFormation to automate policy reviews within CI/CD pipelines (Requirement 6.5.1). Configure the CheckNoNewAccess API to compare proposed IAM policy changes against established reference policies, blocking deployments that introduce unintended privilege escalation. Configure the CheckAccessNotGranted API to validate that deployments do not grant access to defined critical actions or resources within the CDE. This ensures that access control changes undergo automated security validation before reaching production, supporting the requirement that change control processes confirm security controls are not adversely impacted.

Deploy AWS CloudFormation Guard for pre-deployment compliance validation: Implement AWS CloudFormation Guard rules within CI/CD pipelines to evaluate infrastructure-as-code templates against PCI DSS security requirements before resource provisioning (Requirement 6.5.1). Create Guard rule sets that validate encryption configurations, network segmentation controls, logging enablement, and access restrictions. Integrate Guard with cdk-nag for AWS CDK applications to enforce security best practices during development. Run Guard evaluations locally during authoring and within deployment pipelines to shift compliance validation left in the development lifecycle. Store Guard evaluation results in Amazon S3 for audit trail purposes.

Enable separation of development/test and production environments: Configure AWS Organizations organizational unit (OU) structure with separate OUs for development, test, and production workloads, enforcing environment isolation through Service Control Policies (SCPs) (Requirement 6.5.3). Deploy AWS Config to detect and alert on any cross-environment access patterns or data flows between non-production and production accounts. Use AWS Secrets Manager to manage environment-specific credentials, supporting environment promotion (dev to pre-prod to prod) without code changes while maintaining strict separation of authentication material (Requirement 6.5.4). Configure SCPs to prevent production account access from development IAM principals and restrict data copying between environments.

I**mplement change impact analysis and rollback capabilities:** Configure AWS Config advanced queries and configuration timeline to provide before-and-after comparison of resource configurations for all changes to in-scope system components (Requirement 6.5.1). Deploy AWS CloudFormation change sets to preview proposed infrastructure changes before execution,

enabling review of security impact prior to deployment. Implement AWS Systems Manager Automation runbooks with approval workflows requiring documented authorization before changes execute in production environments. Configure Amazon EventBridge rules to capture all CloudFormation stack events and Systems Manager Automation executions, routing them to Amazon CloudWatch Logs for change documentation and audit trail maintenance.

Enforce production data protection in non-production environments: Deploy AWS Config custom rules to detect when production data sources (Amazon RDS snapshots, S3 objects, DynamoDB table exports) are shared with or copied to non-production accounts (Requirement 6.5.4). Configure Service Control Policies to restrict cross-account data sharing between production and non-production OUs. Implement AWS Lambda functions triggered by Amazon EventBridge to automatically revoke unauthorized cross-environment resource sharing. Where test data is required, use AWS Glue or AWS Lambda to implement data masking or tokenization pipelines that sanitize production data before use in non-production environments. Configure AWS CloudTrail to log all cross-account resource access attempts for audit purposes.

Configure automated change documentation and approval workflows: Deploy AWS Systems Manager Automation runbooks with multi-level approval gates that require documented authorization, impact analysis, and back-out procedures before changes execute in production (Requirement 6.5.1). You can also use AWS Systems Manager Change Manager or simple manual approval stages within CodePipeline to implement approval workflow. Configure Amazon EventBridge to capture all change events (CloudFormation deployments, Systems Manager executions, Config rule evaluations) and route them to Amazon CloudWatch Logs with structured metadata including change requester, approver, timestamp, and affected resources. Store change records in Amazon S3 within the Log Archive account with lifecycle policies ensuring retention for assessment evidence. Integrate SSM automation runbooks with AWS Security Hub to validate that security controls remain operational after each change completes.

Leverage Infrastructure as Code (IaC) and source code versioning: IaC source code versioning capabilities helps implement a robust change management that ensure infrastructure is deployed through code and all code changes are authorized and tracked through version control of source code repository. Additionally, when you manage your resources through CloudFormation you can use drift detection to identify stack resources to which configuration changes have been made outside of CloudFormation management. You can then take corrective action so that your stack resources are again in sync with their definitions in the stack template

Access Control and Identity Management

Access control and identity management form the cornerstone of protecting cardholder data environments from unauthorized access and insider threats, especially in the cloud where it can act as the primary security perimeter, replacing traditional network firewalls by defining who can access specific data, applications, and resources. PCI DSS Requirement 7 mandates restricting access to system components and cardholder data by business need to know, while Requirement 8 requires organizations to identify users and authenticate access to system components. In cloud environments, identity and access management services provide opportunities to implement least-privilege access controls, strong authentication mechanisms, and automated access reviews with greater consistency, auditability, and scalability.

Access Control and Identity Management within the AWS SRA

AWS SRA provides foundational guidance for access control and identity management that directly supports key elements of PCI DSS:

Use AWS Identity and Access Management (IAM) to implement least-privilege access controls by creating fine-grained permissions that restrict access to AWS resources based on job functions. IAM supports role-based access control (RBAC) and attribute-based access control (ABAC) patterns, allowing you to grant access based on user attributes such as department, job role, and team name. IAM policies include a default "deny all" as part of the policy evaluation logic, addressing Requirement 7.3.3 for deny-by-default access controls. You are responsible for defining IAM policies that follow the principle of least privilege (Requirement 7.2.2 and 7.2.5) and configuring IAM password policies to enforce minimum password length of 12 characters, password complexity, expiration of 90 days or less, and prevention of password reuse (Requirement 8.3 and 8.6.3). You can generate credential reports for users in their accounts to review who has been granted access to the environment and identify inactive accounts (Requirement 8.2.6). AWS manages the security of IAM as a service, while you are responsible for managing their IAM users, roles, policies, and credentials.

Enable AWS IAM Identity Center to centralize workforce identity management with single sign-on access across multiple AWS accounts within AWS Organizations. Identity Center integrates with external identity providers through SAML 2.0 or can use its own identity store, supporting Requirements 8.2 and 8.3 for centralized user authentication and management. Identity Center uses permission sets to define collections of policies that determine user access levels, supporting

consistent application of least-privilege access controls across accounts (Requirement 7.2.2). Identity Center supports multi-factor authentication (MFA) enforcement, addressing Requirements 8.4 and 8.5. AWS manages the security of the MFA features as part of the Shared Responsibility Model and meets Requirement 8.5.1. You are responsible for configuring permission sets, enabling MFA requirements, and managing user lifecycle processes.

Implement automated access reviews by configuring AWS IAM Access Analyzer to generate least-privilege policy recommendations based on actual access activity. AWS IAM Access Analyzer continuously analyzes resource-based policies to identify resources that are shared with external entities or have overly permissive access. Access Analyzer helps you achieve least privilege by generating policy recommendations based on actual access activity logged in AWS CloudTrail, supporting Requirement 7.2.4 for access reviews and Requirement 7.2.2 and 7.2.5 for least-privilege access. Access Analyzer can identify unused access permissions, enabling you to refine policies and remove unnecessary permissions over time. AWS manages the Access Analyzer service, while you are responsible for reviewing findings and refining policies. Integrate with AWS Security Hub to aggregate access review findings across accounts. Generate access review reports documenting all users, their assigned roles, access levels, and last access dates, storing reports in Amazon S3 for audit purposes. This automation supports Requirement 7.2.4 for reviewing user access at least once every six months.

Leverage AWS Organizations with service control policies (SCPs) to establish organization-wide permission guardrails that apply to all IAM users and roles within member accounts. SCPs act as permission boundaries that define the maximum available permissions, supporting centralized enforcement of access control requirements across the multi-account environment (Requirement 7.3.3). SCPs can prevent accounts from disabling security services, restrict access to specific AWS Regions, or enforce MFA requirements for sensitive operations. AWS manages the Organizations service infrastructure, while you are responsible for defining and maintaining SCPs that align with their access control requirements.

Enable AWS Secrets Manager to securely store and manage database credentials, API keys, and other secrets with automatic rotation capabilities. Secrets Manager encrypts secrets at rest using AWS KMS and in transit using TLS, addressing Requirement 8.3.2 for secure credential storage and transmission. Secrets Manager integrates with AWS CloudTrail to log all API calls, providing an audit trail of credential access and rotation activities (Requirement 7.2.6). Secrets Manager supports automatic credential rotation for Amazon RDS, Amazon Redshift, and Amazon DocumentDB, reducing the risk of credential compromise. You can use Secrets Manager to ensure that application accounts for database applications cannot be used by individual users or other

non-application processes, supporting Requirement 7.2.6 for database access controls. AWS manages the encryption and infrastructure security of Secrets Manager, while you are responsible for defining rotation schedules, access policies, and integration with their applications.

Integrate AWS CloudTrail with Amazon CloudWatch Logs to provide audit trails for all access management activities across AWS accounts. CloudTrail records API calls including IAM user and role activities, authentication events, and access to AWS resources, supporting Requirements 7.2.4 and 8.2 for accountability and access review. CloudTrail logs can be centralized in the Log Archive account as prescribed by the AWS SRA, enabling organization-wide visibility into access patterns. You can use CloudWatch Logs Insights or Amazon Athena to query CloudTrail logs for access investigations and compliance reporting. AWS manages the CloudTrail service infrastructure, while you are responsible for enabling CloudTrail in all accounts, configuring log retention, and analyzing logs for security events.

Add Config Rules in AWS Config to continuously monitor IAM configurations and detect drifts/changes from approved baselines and setup. Config Rules can validate that IAM password policies meet PCI DSS requirements (Requirement 8.3), that MFA is enabled for privileged users (Requirements 8.4 and 8.5), and that IAM policies follow least-privilege principles (Requirement 7.2.5). AWS Config can aggregate compliance data across accounts in AWS Organizations, providing centralized visibility into access control configurations. AWS manages the Config service, while you are responsible for defining Config Rules and remediating non-compliant configurations.

Deploy automated least-privilege validation by configuring AWS Config conformance packs with custom rules that validate IAM policies against least-privilege principles. Deploy Config Rules that detect overly permissive policies (such as policies with wildcard actions or resources), identify IAM users with direct policy attachments instead of group memberships, and flag policies that grant administrative access. Integrate findings with AWS Security Hub to correlate least-privilege violations with other security findings. Configure Amazon EventBridge rules to trigger AWS Lambda functions that generate remediation recommendations and notify security teams of policy violations. This automation supports Requirement 7.2.2 and 7.2.5 for implementing least-privilege access controls.

Implement database access control automation by deploying AWS Secrets Manager to store all database credentials with automatic rotation enabled. Configure Amazon RDS and Amazon Redshift to use IAM database authentication, enabling users and applications to authenticate using temporary credentials instead of database passwords. Use VPC security groups and network ACLs to restrict database access to only authorized application servers and administrative workstations. Deploy AWS Config to validate that databases are not publicly accessible, and that security groups

follow least-privilege principles. Configure AWS CloudTrail to log all database authentication attempts and credential retrievals from Secrets Manager. This automation supports Requirement 7.2.6 for restricting database access to application accounts that cannot be used by individual users.

Configure session management controls by deploying AWS Systems Manager Session Manager to provide secure administrative access to Amazon EC2 instances without requiring SSH keys or open inbound ports. Configure Session Manager to enforce idle session timeouts, log all session activity to Amazon CloudWatch Logs and Amazon S3, and restrict session access based on IAM policies. For IAM Identity Center users, configure permission sets with maximum session durations to restrict the length of time users can be signed in to AWS accounts, supporting Requirement 8.2.7 for third-party access restrictions and Requirement 8.2.8 for idle session timeouts. You must also enforce 15-minute idle session timeouts at user workstations through their external identity provider or AWS Managed Microsoft Active Directory group policies.

Access Control and Identity Management beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

Reduce reliance on long-lived credentials by using IAM roles with AWS Security Token Service (STS) to grant temporary, limited-privilege credentials. STS issues temporary security credentials that automatically expire, reducing the risk of credential compromise and supporting least-privilege access patterns (Requirements 7.2.2, 7.2.5, and 8.2). Programmatic access, including API calls to AWS services, should be performed with IAM roles using temporary credentials issued by STS rather than IAM user access keys. This approach eliminates the need for embedded credentials in application code and supports automated credential rotation. AWS manages the security of STS, while you are responsible for defining IAM role trust policies and permission policies.

AWS recommends not to use IAM users, however in case if you have to user IAM users for any reason you should implement password policy enforcement by configuring IAM password policies across all AWS accounts to enforce minimum password length of 12 characters, require uppercase and lowercase letters, numbers, and symbols, set password expiration to 90 days or less, and prevent reuse of the last four or more passwords (Requirement 8.3). For organizations using IAM Identity Center with an external identity provider, configure password policies in the identity provider to meet PCI DSS requirements. Deploy AWS Config to continuously validate that IAM password policies meet requirements and alert on non-compliant configurations. For IAM users

determined to be in-scope for PCI DSS assessment, implement account lockout mechanisms through identity federation with an external identity provider or AWS Managed Microsoft Active Directory, as IAM does not natively support account lockouts (Requirement 8.3.4).

Deploy MFA enforcement automation by configuring IAM policies to enforce MFA requirements for AWS Management Console, AWS CLI, and API access (Requirements 8.4 and 8.5). Deploy AWS Config to detect IAM users without MFA enabled and automatically publish findings to AWS Security Hub. Configure Amazon EventBridge rules to trigger AWS Lambda functions that notify security teams when privileged users access the console without MFA. For IAM Identity Center, enable MFA enforcement in permission sets to require MFA for all user access. Deploy AWS Lambda functions to analyze AWS CloudTrail logs and detect authentication attempts without MFA, generating alerts through Amazon SNS for security team investigation. This automation supports Requirements 8.4 and 8.5 for multi-factor authentication.

Implement shared credential management by deploying AWS Secrets Manager to securely store any credentials that must be shared on an exception basis with documented business justification. Configure Secrets Manager to log all secret access activities to AWS CloudTrail, providing an audit trail of who accessed shared credentials and when. Implement approval workflows using AWS Lambda and Amazon SNS to require management approval before granting access to shared credentials. Store business justifications and approval records in Amazon S3 for audit purposes. This automation supports Requirement 8.2.2 for managing group, shared, or generic accounts with documented business justification and approval.

Logging, Monitoring, and Incident Response

Logging and monitoring capabilities are essential for detecting, investigating, and responding to security incidents in cardholder data environments. PCI DSS Requirement 10 mandates comprehensive audit logging of all access to system components and cardholder data, enabling organizations to reconstruct events, identify anomalies, and demonstrate compliance through verifiable audit trails. Without adequate logging, it becomes impossible to determine the scope of a compromise, identify the root cause of security incidents, or prove that security controls are operating effectively. In cloud environments, native logging services and centralized log management capabilities provide opportunities to implement comprehensive audit trails with greater automation, retention, and analysis capabilities than traditional on-premises approaches.

Logging, Monitoring, and Incident Response within the AWS SRA

The AWS SRA provides foundational guidance for logging, monitoring, and incident response that directly supports key elements of PCI DSS:

AWS SRA recommends enabling organization trails using AWS CloudTrail with organization of all API activities across all accounts in the AWS organization. CloudTrail automatically logs all management events including user authentication, resource creation and deletion, configuration changes, and policy modifications, addressing Requirements 10.2.1 (logging of individual user access), 10.2.1.1 (all individual user access to cardholder data, when data events are enabled), 10.2.1.2 (actions taken by individuals with administrative access), 10.2.1.3 (access to audit logs), 10.2.1.4 (invalid logical access attempts), 10.2.1.5 (changes to identification and authentication credentials), 10.2.1.6 (initialization of audit logs), and 10.2.1.7 (creation and deletion of system-level objects). CloudTrail logs include the identity of the API caller, the time of the API call, the source IP address, the request parameters, and the response elements returned by the AWS service. AWS manages the underlying logging infrastructure and ensures log integrity through digital signatures, while customers are responsible for enabling CloudTrail in all accounts, configuring organization trails to capture activity across all accounts, and ensuring logs are delivered to a centralized S3 bucket in a dedicated log archive account. Enable CloudTrail log file integrity validation to ensure that log files have not been modified after delivery to S3. Configure S3 Object Lock in compliance mode on the log archive bucket to prevent deletion or modification of log files during the retention period, supporting Requirement 10.5.2 for protecting audit logs from

destruction and unauthorized modifications. Create an AWS Config rule that validates S3 bucket configurations to ensure versioning, encryption, access logging, and Object Lock are enabled on all buckets storing audit logs.

Amazon CloudWatch Logs integrated with CloudTrail, VPC Flow Logs, and application logs provide centralized log aggregation and real-time monitoring capabilities. CloudWatch Logs enables customers to collect logs from EC2 instances, containers, Lambda functions, and other compute resources using the CloudWatch Logs agent or AWS SDK, addressing Requirements 10.2.2 (automated audit trails for system components) and 10.4.1 (review of logs). Customers can create metric filters to extract specific patterns from log data and trigger CloudWatch alarms when suspicious patterns are detected, supporting Requirement 10.4.1.1 for automated mechanisms to perform audit log reviews. CloudWatch Logs Insights provides an interactive query language for analyzing log data in near real-time, supporting forensic investigations and compliance reporting.

AWS Config continuously records configuration changes to AWS resources and evaluates resource configurations against desired states defined in Config Rules. Config automatically logs changes to security groups, IAM policies, S3 bucket policies, encryption settings, and other security-relevant configurations, addressing Requirement 10.2.1.5 for logging changes to identification and authentication credentials and system-level objects. Config integrates with CloudTrail to provide both the "what changed" (Config) and "who changed it" (CloudTrail) for comprehensive change auditing. AWS manages the Config service infrastructure, while customers are responsible for enabling Config in all accounts containing CDE resources and defining Config Rules that validate security configurations.

AWS Systems Manager Session Manager provides secure, auditable access to EC2 instances and on-premises servers managed by Systems Manager. Session Manager eliminates the need for bastion hosts, SSH keys, or open inbound ports by establishing connections through the Systems Manager service. All session activity is logged to CloudWatch Logs and can optionally be logged to S3, providing complete audit trails of interactive access to system components (Requirements 10.2.1.1 and 10.2.1.2). Session Manager integrates with AWS Identity and Access Management (IAM) for authentication and authorization, ensuring that only authorized users can establish sessions and that all session activity is attributed to specific IAM principals.

Amazon GuardDuty analyzes CloudTrail logs, VPC Flow Logs, and DNS logs to detect suspicious access patterns, compromised credentials, and unauthorized behavior. GuardDuty uses machine learning models, anomaly detection, and threat intelligence feeds to identify findings such as unusual API calls, access from known malicious IP addresses, credential exfiltration attempts, and privilege escalation activities, supporting Requirement 10.4.1.1 for automated audit log review

mechanisms. GuardDuty findings are published on Amazon EventBridge and AWS Security Hub, enabling automated alerting and response workflows.

AWS Security Hub detects security issues by automatically correlating and enriching security signals from multiple sources, such as posture management, vulnerability management (Amazon Inspector), sensitive data (Amazon Macie), and threat detection (Amazon GuardDuty). Security Hub findings are formatted in the Open Cybersecurity Schema Framework (OCSF), enabling consistent analysis and response workflows regardless of the finding source.

AWS Security Hub CSPM provides you with a comprehensive view of your security state in AWS and helps you assess your AWS environment against security industry standards and best practices. The service includes the PCI DSS security standard, which automatically evaluates AWS resource configurations against PCI DSS requirements and generates compliance findings, supporting Requirements 10.4.1 and 10.4.2 for log review and audit trail analysis.

Amazon S3 with versioning, encryption, and access logging enabled provides durable, tamper-evident storage for audit logs. The AWS SRA prescribes deployment of a dedicated log archive account where CloudTrail logs, VPC Flow Logs, Config snapshots, and other audit logs are centrally stored. S3 bucket policies and Service Control Policies (SCPs) prevent deletion or modification of logs, supporting Requirements 10.5.2 (protection of audit logs from destruction and unauthorized modifications) and 10.5.3 (prompt backup of audit log files to a secure centralized log server). S3 lifecycle policies enable automated transition of logs to S3 Glacier for cost-effective long-term retention, supporting Requirement 10.5.1 for retaining audit logs for at least 12 months with at least three months immediately available for analysis.

AWS KMS provides encryption key management for protecting audit logs at rest and in transit. CloudTrail, CloudWatch Logs, and S3 integrate with KMS to encrypt logs using customer-managed keys, ensuring that only authorized principals can decrypt and access log data. KMS key policies and IAM policies control access to encryption keys, supporting Requirement 10.5.2 for protecting audit logs from unauthorized access. KMS automatically rotates key material annually for Customer managed-keys with automatic rotation enabled, supporting cryptographic key management best practices.

Amazon Detective automatically collects log data from CloudTrail, VPC Flow Logs, GuardDuty findings, and Amazon EKS audit logs, organizing the data into a graph model that visualizes relationships between users, IP addresses, AWS resources, and actions over time. Detective enables security analysts to rapidly investigate the scope and timeline of security incidents by providing pre-built visualizations showing resource access patterns, API call sequences, and

anomalous behaviors. Detective supports Requirement 10.4.3 for responding to suspected or confirmed security incidents by enabling rapid forensic analysis of audit logs to determine the scope of compromise, identify affected systems, and reconstruct attacker actions. AWS manages the Detective infrastructure and graph database, while customers are responsible for enabling Detective in accounts containing CDE resources and training security personnel on investigation workflows

AWS Security Incident Response helps you prepare for, and respond to, security incidents in your AWS environment. It triages findings, escalates security events, and manages cases that require your immediate attention. Additionally, it gives you access to the AWS Customer Incident Response Team (CIRT), which investigates impacted resources. AWS Security Incident Response also provides automated response and remediation capabilities through AWS Systems Manager documents (SSM documents), which help security teams respond to, and recover from, security incidents more efficiently. AWS Security Incident Response integrates with Amazon GuardDuty and AWS Security Hub CSPM to receive security findings and orchestrate automated responses.

Logging, Monitoring, and Incident Response beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

Enable CloudTrail data events for cardholder data access logging: Configure CloudTrail to log data events for S3 buckets and Lambda functions that process cardholder data. Data events provide detailed logging of object-level API operations such as GetObject, PutObject, and DeleteObject, enabling comprehensive audit trails of all access to cardholder data stored in S3 (Requirement 10.2.1.1). Deploy AWS Lambda functions triggered by Amazon EventBridge to analyze CloudTrail data events in near real-time and generate alerts when cardholder data is accessed by unauthorized principals or from unexpected locations. Store data event logs in a dedicated S3 bucket in the log archive account with encryption enabled using AWS KMS.

Implement automated time synchronization validation: Deploy AWS Config to continuously validate that all EC2 instances, RDS databases, and other system components are configured to synchronize time with authoritative time sources. Create custom Config Rules that query the Amazon Time Sync Service configuration on EC2 instances and validate that NTP or Chrony is configured to use the 169.254.169.123 link-local address. Configure AWS Systems Manager Automation documents that automatically remediate time synchronization misconfigurations,

supporting Requirements 10.6.1, 10.6.2, and 10.6.3 for acquiring, synchronizing, and protecting time data from authoritative sources.

Deploy centralized log correlation and analysis: Configure Amazon CloudWatch Logs Insights or deploy Amazon OpenSearch Service to provide centralized log correlation and analysis capabilities across CloudTrail logs, VPC Flow Logs, application logs, and operating system logs. Create saved queries and dashboards that correlate user authentication events from CloudTrail with network access events from VPC Flow Logs and application access events from CloudWatch Logs, enabling detection of suspicious access patterns that span multiple log sources (Requirement 10.4.1.1). Deploy AWS Lambda functions to execute automated queries daily and generate summary reports of security-relevant events, supporting Requirement 10.4.1 for daily log reviews.

Configure comprehensive application and system logging: Deploy the Amazon CloudWatch Logs agent on all EC2 instances and configure it to collect operating system logs (syslog, auth.log, secure), web server logs (Apache, Nginx access and error logs), application logs, and database audit logs. For containerized workloads, configure Amazon ECS and Amazon EKS to forward container logs to CloudWatch Logs using the awslogs log driver. For serverless workloads, ensure all Lambda functions write structured logs to CloudWatch Logs including user identity, timestamp, event type, success/failure indication, and origination of event, supporting Requirements 10.2.2 (automated audit trails) and 10.3 (audit log record content requirements including 10.3.1 through 10.3.4). Configure log retention periods in CloudWatch Logs to retain logs for at least 12 months, with automated export to S3 for long-term archival.

Configure audit log retention and availability: Implement S3 lifecycle policies that retain audit logs in S3 Standard storage class for three months to ensure immediate availability for analysis (Requirement 10.5.1), then transition logs to S3 Standard-IA or S3 Glacier for cost-effective retention for the remaining nine months of the 12-month retention period. For environments requiring longer retention periods, configure lifecycle policies to retain logs for the required duration. Configure S3 bucket policies and IAM policies to ensure that only authorized security personnel can access archived logs, supporting Requirements 10.5.1 and 10.5.4 for log retention and restricting access to audit log history files.

Implement detection of failures in critical security control systems: Deploy Amazon CloudWatch alarms that monitor the operational status of critical logging and monitoring services including CloudTrail, Config, GuardDuty, and CloudWatch Logs. Configure alarms to trigger when CloudTrail logging is stopped, Config recorder is disabled, GuardDuty is suspended, or CloudWatch Logs agent stops sending logs from critical system components. Integrate alarms with Amazon SNS to immediately alert security personnel of logging failures and configure AWS Systems Manager

Automation documents to automatically attempt remediation of common failure scenarios, supporting Requirement 10.7.2 for detecting failures of critical security control systems.

Implement automated forensic evidence collection: Deploy AWS Step Functions state machines that orchestrate automated forensic evidence collection when security incidents are detected. Configure Step Functions workflows to invoke AWS Lambda functions that collect relevant CloudTrail logs, VPC Flow Logs, EC2 instance snapshots, memory dumps using AWS Systems Manager Run Command, and network packet captures using VPC Traffic Mirroring. Store forensic evidence in a dedicated S3 bucket with write-once-read-many (WORM) protection enabled using S3 Object Lock, ensuring evidence integrity for investigations and potential legal proceedings. Configure Step Functions to automatically invoke Amazon Detective to generate investigation visualizations showing the scope and timeline of the incident, supporting Requirement 10.4.3 for determining the scope of compromise and identifying affected system components and data.

Implement automated containment and remediation workflows: Configure AWS Systems Manager Automation documents that define automated containment and remediation actions for common security incidents detected through log monitoring. Create automation documents that respond to GuardDuty findings by automatically isolating compromised EC2 instances (modifying security groups to deny all traffic), rotating compromised IAM credentials (deleting access keys and forcing password resets), blocking malicious IP addresses (updating AWS WAF rules and network ACLs), and terminating unauthorized resources (deleting EC2 instances or Lambda functions created by attackers).

Vulnerability and Patch Management

Vulnerability and patch management form the cornerstone of proactive security in cardholder data environments, enabling organizations to identify, prioritize, and remediate security findings before they can be exploited. PCI DSS Requirements 5 (Malware Protection), 6 (Secure System and Software Development) and 11 (Security Testing and Monitoring) establish comprehensive requirements for continuous vulnerability assessment, timely patch deployment, and protection against malware that exploits unpatched vulnerabilities. Without effective vulnerability and patch management, organizations accumulate technical debt in the form of known security weaknesses, creating an expanding vulnerable surface that increases the likelihood and potential impact of security incidents. In cloud environments, native vulnerability assessment services and automated patch management capabilities enable organizations to implement continuous vulnerability management with greater speed, consistency, and coverage than traditional periodic assessment approaches.

> **Note**
>
> PCI DSS 4.0.1 Requirement 5, dealing with vulnerability management for network and systems components are out-of-scope for this guide as AWS networking services are AWS managed and fall under AWS responsibility. For compute services like EC2 users are responsible for vulnerability management.

Vulnerability and Patch Management within the AWS SRA

The AWS SRA provides foundational guidance for vulnerability and patch management that directly supports key elements of PCI DSS:

Using Amazon Inspector with continuous scanning enabled provides automated vulnerability assessment that exceeds the minimum quarterly scan frequency required by Requirements 11.3.1 and 11.3.2. Inspector discovers EC2 instances, container images in ECR repositories, and Lambda functions, then continuously scans them for software vulnerabilities using the CVE database maintained by the National Vulnerability Database (NVD). Inspector generates findings with severity ratings based on the Common Vulnerability Scoring System (CVSS) version 3.1, including the CVE identifier, affected software package, installed version, fixed version, CVSS score, and remediation guidance. Inspector findings can be published to AWS Security Hub and

Amazon EventBridge, enabling centralized tracking and automated remediation workflows. Inspector supports both network reachability analysis (identifying unintended network exposure) and package vulnerability scanning (identifying vulnerable software packages), addressing Requirements 11.3.1 for internal vulnerability scans and Requirement 11.3.1.2 since these are considered authenticated scans.

AWS Security Hub provides centralized vulnerability management by aggregating findings from Amazon Inspector, AWS Config, Amazon GuardDuty, and third-party security products into a unified view. Security Hub workflow status enables tracking of vulnerability remediation from initial detection through verification of remediation, supporting Requirement 11.3.1.1 for rescanning to verify vulnerabilities are corrected. Security Hub insights provide aggregated views of vulnerability trends, showing vulnerability counts by severity, age, resource type, and account, enabling security teams to identify systemic issues and measure vulnerability management program effectiveness. AWS manages the Security Hub infrastructure and PCI DSS security standard logic, while customers are responsible for enabling Security Hub in all accounts, configuring automated response workflows, tracking vulnerabilities through to resolution, and ensuring remediation timelines align with Requirement 6.3.3.

AWS Config with Config Rules provides continuous monitoring of patch compliance status and security configurations. Config Rules can validate that EC2 instances are compliant with patch baselines defined in Systems Manager, that security groups do not permit unrestricted access on sensitive ports, that S3 buckets have encryption enabled, and that other security configurations align with PCI DSS requirements. The managed Config Rule ec2-managedinstance-patch-compliance-status-check evaluates whether EC2 instances managed by Systems Manager are compliant with patch baselines, generating non-compliant findings when instances have missing patches or overdue patches. Config integrates with Systems Manager to provide detailed patch compliance data including which patches are missing, patch severity, and how long patches have been overdue, supporting Requirements 6.3.3, 6.3.2, and 6.3.1 for prioritizing, installing, and documenting security patches. Config conformance packs enable deployment of collections of Config Rules that validate multiple security configurations simultaneously, providing comprehensive compliance monitoring across accounts. AWS manages the Config service infrastructure and managed Config Rules, while customers are responsible for enabling Config in all accounts containing CDE resources, deploying Config Rules that validate patch compliance and security configurations, and configuring automated remediation for common compliance violations.

AWS Systems Manager State Manager maintains desired state configurations for managed instances; security configurations, and monitoring agents remain properly configured over time. State Manager associations define the desired state (such as "all instances must have the CloudWatch Logs agent installed and running") and the schedule for enforcing that state. State Manager can automatically remediate configuration drift by re-applying configurations when instances deviate from the desired state, supporting Requirement 11.5.2 for maintaining secure configurations. State Manager integrates with Systems Manager documents (SSM documents) to define the actions taken to enforce desired state, including installing software, applying patches, modifying configurations, and validating compliance. State Manager execution history is logged to CloudTrail and CloudWatch Logs, providing audit trails of all configuration enforcement activities. AWS manages the State Manager service infrastructure, while customers are responsible for defining State Manager associations that enforce patch compliance and security configurations, monitoring association execution status, and investigating association failures.

AWS Systems Manager Automation provides runbook-based automation for vulnerability remediation and patch deployment. Automation documents define multi-step workflows that can patch instances, update container images, modify security configurations, and verify remediation success. Automation integrates with Amazon EventBridge to enable event-driven remediation, automatically executing runbooks when vulnerabilities are detected by Inspector or when patch compliance violations are detected by Config. Automation documents can include approval steps for high-risk remediation actions, conditional logic to handle different scenarios, and error handling to ensure graceful failure recovery. Automation execution history is logged to CloudTrail and stored in Systems Manager, providing audit trails of all automated remediation activities, supporting Requirement 6.5.1 for documenting vulnerability remediation and change control. AWS provides pre-built automation documents for common remediation scenarios, while customers are responsible for creating custom automation documents for application-specific remediation, configuring event-driven triggers, and monitoring automation execution success rates.

AWS Systems Manager Patch Manager integrated with patch baselines and maintenance windows enables automated, customer defined risk-based patch deployment across EC2 instances and on-premises servers. Patch Manager supports predefined patch baselines for major operating systems including Amazon Linux, Red Hat Enterprise Linux, Ubuntu, Windows Server, SUSE Linux, and macOS, with patch baselines defining which patches are automatically approved for installation based on severity, classification, and release date. Customers can create custom patch baselines that implement the customer defined risk-based patch prioritization required by Requirement 6.3.1, automatically approving critical patches for installation within 30 days and high-risk patches according to customer risk ranking. Patch Manager integrates with maintenance

windows to control when patches are installed, minimizing business disruption while ensuring timely patch deployment. Patch Manager provides detailed compliance reporting showing which instances are compliant with patch baselines, which patches are missing, and how long patches have been overdue, supporting Requirement 6.3.3 for documenting security patch installations. Patch Manager can be configured to automatically reboot instances after patch installation when required, or to defer reboots to scheduled maintenance windows. AWS manages the Patch Manager service infrastructure and patch metadata repositories, while customers are responsible for defining patch baselines aligned with organizational risk rankings, configuring maintenance windows, monitoring patch compliance status, and investigating patch installation failures.

Vulnerability and Patch Management beyond the AWS SRA

Some incremental enhancements may be needed to extend the AWS SRA for alignment with additional PCI DSS requirement elements:

Implement risk-based patch prioritization and automation: Create custom AWS Systems Manager patch baselines that implement organizational risk ranking as required by Requirements 6.3.1 and 6.3.2. Define patch baselines that expedite critical patches (CVSS score 9.0-10.0) for installation within 30 days, high-risk patches (CVSS score 7.0-8.9) according to organizational risk assessment, and medium/low-risk patches on a longer timeline. Configure patch baselines with different approval rules for different system types, such as more aggressive patching for internet-facing systems and more conservative patching for sensitive internal systems. Configure Systems Manager maintenance windows with different schedules for different system criticality levels, such as weekly maintenance windows for development systems and monthly maintenance windows for production systems. Deploy Amazon EventBridge rules that trigger AWS SNS notifications when critical patches are available, enabling security teams to expedite patch deployment for high-risk vulnerabilities, supporting Requirements 6.3.1, 6.3.2, and 6.3.3 for timely, risk-based patch deployment.

Configure automated patch compliance monitoring and reporting: Deploy AWS Config that continuously monitor patch compliance status for all EC2 instances and on-premises servers managed by Systems Manager. Create custom Config Rules using AWS Lambda that evaluate patch compliance against organizational risk rankings, identifying instances with critical patches overdue by more than 30 days, high-risk patches overdue according to risk-based timelines, or instances that have not been scanned for patches in the past 30 days. Configure Amazon EventBridge rules to trigger AWS Lambda functions that generate patch compliance reports on a weekly basis, documenting patch compliance rates by account, application, and system type, overdue patches

with severity and age, instances requiring remediation, and patch deployment trends over time. Integrate with Amazon QuickSight to create interactive patch compliance dashboards showing compliance trends, mean time to patch (MTTP) metrics, patch deployment success rates, and comparison of patch compliance across accounts and applications.

Implement automated vulnerability remediation workflows: Deploy AWS Systems Manager Automation documents that define automated remediation actions for common vulnerability types detected by Amazon Inspector. Create automation documents that respond to Inspector findings by automatically applying security patches using Patch Manager (for operating system and application vulnerabilities), updating container images in ECR to patched versions (for container vulnerabilities), updating Lambda function runtime versions (for serverless vulnerabilities), or modifying security group rules (for unintended network exposure findings). Configure AWS Security Hub custom actions that enable security analysts to trigger automated remediation with a single click from the Security Hub console, with automation documents executing the appropriate remediation based on finding type. Deploy Amazon EventBridge rules that automatically execute remediation workflows for critical vulnerabilities (CVSS score 9.0 or higher) while requiring manual approval for lower-severity findings, balancing automation speed with change control requirements. Configure AWS Step Functions workflows that orchestrate multi-step remediation processes, including creating snapshots before remediation, executing remediation actions, validating remediation success, triggering Inspector rescans to verify vulnerability resolution, and updating Security Hub workflow status, supporting Requirements 11.3.1.3 and 6.3.3 for timely vulnerability remediation and verification.

Implement container image vulnerability management: Configure Amazon ECR with enhanced scanning enabled on all repositories containing images used in the CDE. Deploy Amazon ECR lifecycle policies that automatically delete or quarantine container images with critical vulnerabilities (CVSS score 9.0 or higher) that have been unresolved for more than 30 days, preventing deployment of vulnerable images to production environments. Create AWS Lambda functions that monitor Amazon ECR scan results via Amazon EventBridge, identify base images with vulnerabilities, and automatically trigger AWS CodePipeline pipelines to rebuild container images using patched base images. Configure CodePipeline with security gates that block deployment of container images with critical or high-severity vulnerabilities, requiring manual approval or vulnerability remediation before deployment proceeds. Deploy AWS Lambda functions that generate container image vulnerability reports showing vulnerability counts by repository, image age, base image vulnerabilities versus application dependency vulnerabilities, and remediation timelines, supporting Requirements 6.3.1, 6.3.3, and 11.3.1.1 for identifying and remediating vulnerabilities in containerized applications. Integrate with Amazon ECS and Amazon

EKS admission controllers to prevent deployment of container images that have not been scanned or that contain critical vulnerabilities, enforcing vulnerability management policies at deployment time.

Configure vulnerability exception and risk acceptance workflow: Deploy AWS Lambda functions and Amazon DynamoDB to create a vulnerability exception management system that tracks vulnerabilities that cannot be remediated within standard timelines due to technical constraints, business requirements, or vendor limitations. Configure the exception management system to require documented business justification, compensating controls, risk acceptance approval from authorized personnel, and expiration dates for all vulnerability exceptions. Integrate the exception management system with AWS Security Hub to automatically suppress findings for approved exceptions while maintaining audit trails of exception approvals and expirations. Deploy Amazon EventBridge rules that trigger notifications via Amazon SNS when vulnerability exceptions are approaching expiration, requiring re-evaluation of risk acceptance decisions. Create AWS Lambda functions that generate quarterly vulnerability exception reports showing all active exceptions, compensating controls, risk owners, and expiration dates, supporting Requirements 6.3.3 and 11.3.1.1 for documenting vulnerability management decisions and risk acceptance.

Implement patch testing and validation workflows: Deploy dedicated AWS accounts for patch testing using AWS Organizations, enabling isolated testing of patches before deployment to production environments. Configure AWS Systems Manager to deploy patches to test environments first, with automated testing using AWS Lambda functions that validate application functionality, performance, and security after patch installation. Create AWS Step Functions workflows that orchestrate the complete patch lifecycle, including deploying patches to test environments, executing automated tests, collecting test results, requiring manual approval for production deployment, deploying patches to production environments in phases (such as canary deployments to 10% of instances, then 50%, then 100%), and monitoring application health during production deployment. Deploy Amazon CloudWatch alarms that detect application failures or performance degradation after patch deployment, automatically triggering rollback procedures using AWS Systems Manager to restore instances to pre-patch snapshots. Configure AWS Lambda functions that generate patch deployment reports documenting patches tested, test results, production deployment timelines, and any issues encountered, supporting Requirements 6.3.3 and 6.5.1 for documenting patch installations and change control processes.

Conclusion

This guidance demonstrates how the AWS Security Reference Architecture (AWS SRA) provides a strong, extensible foundation for PCI DSS v4.0.1 compliance in a multi-account AWS environment. By mapping AWS SRA patterns to PCI DSS requirements across nine control domains — Organization Architecture and Scoping, Network Security, System Hardening, Encryption and Key Management, Secure SDLC, Change Management, Access Control, Logging and Monitoring, and Vulnerability Management — the guide shows that PCI DSS compliance on AWS is an extension of sound cloud security architecture.

Key Takeaways

Principle	Guidance
1. Scoping drives everything	A well-designed OU and account structure — segmenting CDE, Connected-To, and Out-of-Scope accounts — is one of the most impactful lever for reducing assessment surface and clarifying control ownership.
AWS SRA is necessary foundation with extension to meet PCI DSS	Each domain distinguishes "within the AWS SRA" (baseline already met) from "beyond the AWS SRA" (additional PCI-specific configura tion required). Use this to target gaps without over-engineering.
Automate for continuous compliance	AWS Config, Security Hub, CloudTrail, and EventBridge enable a shift from point-in-time audits to continuous monitoring and automated remediation — essential for PCI DSS v4.0's emphasis on ongoing security.
Identity is the cloud perimeter	IAM, Organizations SCPs, and centralized federation enforce least privilege and strong authentication at the platform level (Requirem ents 7, 8).

Layered segmentation replaces flat perimeters	VPC architecture, Network Firewall, security groups, PrivateLink, and Transit Gateway — applied across accounts and OUs — satisfy Requirement 1 without reliance on single-ap pliance designs.
Tagging is infrastructure	A structured PCI tagging strategy underpins inventory management, automated complianc e checks, and scope validation across the standard.

Next Steps

1. Compare your existing AWS architecture against the OU structure and segmentation boundaries in this guide to identify scoping gaps.
2. Prioritize building a strong AWS multi-account security architecture following the AWS SRA to ensure you are building a strong security posture to meet the intent of PCI DSS requirements.
3. Automate controls and change management as much as possible to ensure consistency and continuous compliance.
4. Engage your Qualified Security Assessor (QSA) during the design phase to validate scoping decisions before implementation.
5. Download AWS's PCI DSS AOC and Shared Responsibility Matrix from AWS Artifact to establish inherited controls.

PCI DSS on AWS - Related Resources

Resource	Description	Link
PCI DSS v4.0 on AWS Compliance Guide	Comprehensive guide covering requirement-by-req uirement guidance, scoping, segmentation, and the Shared	Download PDF

	Responsibility Model for PCI DSS v4.0 on AWS	
Architecting for PCI DSS Scoping and Segmentation on AWS	Detailed network architecture patterns, multi-account segmentation strategies, firewall rule examples, and reference architectures for defining PCI DSS scope boundaries on AWS	Download PDF
Architecting on Amazon ECS for PCI DSS Compliance	Best practices for configuring Amazon ECS (Fargate and EC2 launch types) for PCI DSS compliance, covering network segmentation, host hardening, data protection, and monitoring	Download PDF
Architecting Amazon EKS for PCI DSS Compliance	Guidance on configuring Amazon EKS (Fargate and EC2 launch types) for PCI DSS compliance, including Kubernetes network policies, pod security, secrets management, and container scanning	Download PDF

Contributors

The following individuals contributed to this guide

Authoring:

- Avik Mukherjee, AWS Senior Security SA
- Akanksha Chaturvedi, AWS Senior Assurance Consultant
- Nimesh Ravasa, AWS Senior Assurance Consultant
- Omner Barajas, AWS Senior Security SA
- Viktor Mu, AWS Senior Assurance Consultant

Reviewing:

- Eric Rose, AWS Principal Security SA
- Shashi Karanam, AWS Senior Assurance Consultant
- Steve Christensen, AWS Industry Specialist-PCI

Document History

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Initial Release	First release of the guidance	21 July, 2026