Architecting for PCI DSS Scoping and Segmentation on AWS

Identify and Minimize Your PCI DSS Scope Using Appropriate Segmentation Controls

First published April 2019

Last updated May 2023

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.

AWS Security Assurance Services, LLC (AWS SAS) is a fully owned subsidiary of Amazon Web Services (AWS). AWS SAS is an independent PCI QSA company (QSAC) that provides AWS customers and partners with specific and prescriptive information on PCI DSS compliance. As a PCI QSAC, AWS SAS can interact with the PCI Security Standards Council (SSC) or other PCI QSAC under the confidentiality and contractual framework of PCI.

Contents

Abstract

This paper provides guidance on how to properly define the scope of your Payment Card Industry Data Security Standard (PCI DSS) workloads running on the AWS Cloud, as well as how to define segmentation boundaries between your in-scope resources and your out-of-scope resources by using cloud-native AWS services. This paper is intended for engineers and solution builders, but it also serves as a guide for qualified security assessors (QSAs) and internal security assessors (ISAs) to better understand the segmentation controls available on AWS and the associated scoping considerations.

Introduction

Software-defined networking on AWS transforms the scoping process for applications as compared to on-premises environments. Additional segmentation controls available on AWS go beyond just network segmentation. You can substantially reduce the number of systems and services in your cardholder data environment (CDE) with the thoughtful selection of security impacting services and required controls when designing your applications. This can help reduce compliance costs and efforts significantly. For details and guidance on how to apply security best practices and recommendations for the design, delivery, and maintenance of secure AWS workloads, see the AWS Well-Architected Security Pillar.

PCI DSS scoping process on AWS

It is critical to understand the complete flow of account data within applications and the environment, including interactions with procedures and application code. The evaluation of data flow in the environment, as well as all *connected to* and supporting systems components, determines the applicability of the PCI DSS requirements and defines the boundaries and components of a CDE, and the scope of a PCI DSS assessment.

Understand scoping

The requirements of the PCI DSS are mandatory for organizations that store, process, or transmit account data. Account data consists of cardholder data (CHD) and sensitive authentication data (SAD). This data security standard maintained by the PCI Security Standards Council includes a prescriptive set of IT security requirements—known as *controls*—to protect sensitive credit card information. CHD comprises the primary account number (PAN), and can also include the cardholder's name, card expiration date, and service code, if included with the PAN. SAD includes full magnetic-stripe or track data, card verification codes, and PINs or PIN blocks. The PAN is the defining factor for cardholder data and is the basis for determining the associated CDE. For a service provider, complying with the PCI DSS might also

be a customer requirement. Organizations are responsible for correctly identifying and defining their CDE, which is referred to as the scope of their PCI DSS assessment. The CDE comprises people, processes, and technologies that interact with or impact the security of account data and thus are in-scope for the PCI DSS. In this paper, we focus on the technology aspect of the CDE scope and not on the people or processes. Additional information on the applicability of the PCI DSS and account data can be found on page 8 of the PCI DSS v4.0.

The PCI DSS uses a term called *system components* to identify resources that are considered in-scope and part of the CDE. *System components* are a general term for technology resources that store, process, or transmit account data, have unrestricted connectivity to the aforementioned PCI resources, or could otherwise impact the security of the environment. On AWS, this includes AWS resources and services that handle AWS account data, or support the account and resources that are PCI in-scope. This can include services that are actively involved in storing or processing cardholder data, such as Amazon Elastic Compute Cloud (Amazon EC2), Amazon Relational Database Service (Amazon RDS), or AWS Lambda, as well as supporting services, such as AWS Config, AWS Identity and Access Management (IAM), and AWS Organizations.

Segmentation

Segmentation is colloquially referred to as *PCI DSS Requirement 0,* and it is not a formal PCI DSS requirement. Segmentation is used to limit the PCI DSS scope to the systems involved in the payment flow before addressing other PCI DSS requirements. Traditionally, organizations use network segmentation as a key control to limit their PCI DSS in-scope environment and protect it from the rest of their IT infrastructure. This approach does not imply that organizations must not secure their out-of-scope infrastructure; rather, they have more flexibility in their choice of security controls, and different validation requirements for their out-of-scope infrastructure.

The PCI SSC Guidance for PCI DSS Scoping and Network Segmentation categorizes IT infrastructure and resources into the following segments with respect to PCI DSS scope:

- **CDE systems**: These system components store, process, or transmit account data (CHD and SAD). This can also include system components on the same network segment that have unrestricted network access. These components are in-scope of PCI DSS and can bring other resources in-scope.
- ***Connected to*** **and** ***security impacting*** **systems**: These system components have a direct or indirect restricted connection to CDE systems, or provide some sort of management or security services to CDE systems. These components might help fulfill one or more PCI DSS requirements or help establish segmentation boundaries. These are in-scope systems, but they do not extend PCI DSS scope to other resources.

- **Out-of-scope systems**: These system components do not meet the preceding criteria and do not impact the security or configuration of CDE systems. For a system to be considered out-of-scope, controls must be in place to provide assurance that the out-of-scope system cannot be used to compromise an in-scope system component. These systems are not considered in-scope.

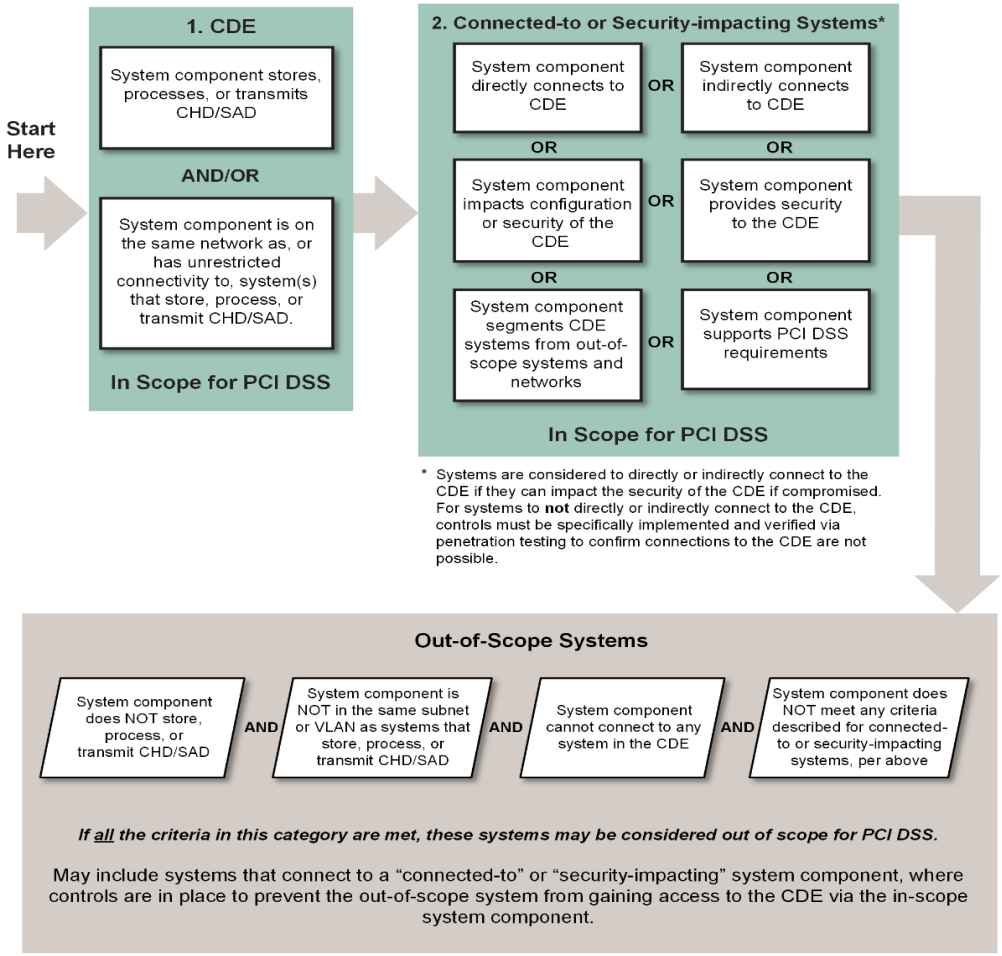

Figure 1: Overview of PCI DSS scoping

Strategies for minimizing PCI DSS scope cover these three categories: CDE, *connected to* or *security-impacting*, and out-of-scope. This whitepaper also addresses scoping and segmentation considerations for a hybrid environment, where the in-scope workloads might be spread across your on-premises data center and the AWS Cloud.

Similar to an on-premises scoping process, scoping an AWS PCI DSS environment begins with the account data flow. The significant difference with the AWS Cloud is that many network segments in the CHD and SAD flow could include AWS purpose-built services, such as Amazon

API Gateway or AWS WAF. These services have well-defined connection configurations and are assessed as part of the AWS PCI DSS Level 1 Service Provider assessment. The AWS endpoints for these services are RESTful web service interfaces that are protected by firewall functionality as part of the AWS PCI DSS scope and serve as segmentation boundaries for services not receiving account data. You need to carefully review the segmentation capabilities provided by AWS services and networking features against your documented data flows to determine an accurate assessment scope. You will learn about these segmentation capabilities in this whitepaper.

Infrastructure services

For infrastructure services where you control both the data and network layer, you need to take additional scoping steps. First, you need to identify instances that store, process, or transmit account data. Then you must identify the *connected to* and *security impacting* AWS resources. For example, an Amazon EC2 instance running a web service that handles account data would be in-scope. However, this EC2 instance can bring many other *connected to* resources in-scope because you are responsible for defining and managing the network connections. An AWS security group acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic. If security groups attached to EC2 instances are not adequately configured and restricted, other instances that can establish a network connection to and from these instances in the same subnet or virtual private cloud (VPC) are potentially *connected to* systems and are considered in-scope for PCI DSS. These *connected to* systems might include systems such as a monitoring server polling non-CHD data for reporting. EC2 instances and other AWS resources that are considered *connected to* could also be considered *security impacting*. *Security impacting* systems can include EC2 instances that host security tooling, such as antivirus protections for the EC2 instances, or other AWS services such as AWS Directory Service for Microsoft Active Directory that provide authentication and authorization. This same scoping guidance also applies to other AWS services where you manage the underlying EC2 instance, such as Amazon Elastic Kubernetes Service (Amazon EKS) and Amazon Elastic Container Service (Amazon ECS) with the EC2 launch type.

Managed services

Managed services require a similar consideration as infrastructure services regarding scoping and network connectivity. However, with AWS managed services, AWS manages more of the underlying resource infrastructure. This abstracts the responsibility of maintaining operating systems from you and reduces your PCI DSS scope at the resource level. However, the non-abstracted portion of the service is your responsibility and includes network security. For example, an organization setting up Amazon RDS needs to verify network segmentation by applying appropriate security groups to the elastic network interface in the VPC and by restricting user access to the service with IAM. Managed service instances are isolated by design to help ensure that data is not shared between different instances of the service unless otherwise explicitly allowed through a configuration change that you manage.

Abstracted services

By design, abstracted services are secured to verify that data is not shared between different instances of the service unless otherwise explicitly allowed. For abstracted services that store, process, or transmit account data, the component in-scope is the particular instantiation of the AWS service that handles account data. Also in-scope are specific resources that both connect to or are connected to from the abstracted service instance. For example, an organization might have many Amazon DynamoDB tables provisioned but only use a subset of those tables to store or process account data. In this case, the tables that are used to store account data are part of the organization's PCI DSS scope. The same is true for Lambda instances that handle account data. The parties calling the function are in-scope, as well as the resources that the Lambda function connects to as part of the data flow. *Connected to* in-scope resources are both the systems and services that pass account data to the abstracted service resource in question, and also those that in some form connect to the CDE even if the data flow does not contain account data. An example is an Amazon Simple Storage Service (Amazon S3) bucket that receives operational logs from a payment application server hosted on an EC2 instance. The S3 bucket is considered *connected to* even if no account data exists in those logs. In this example, you can use Amazon Macie, a fully-managed data security and data privacy service that uses machine learning and pattern matching to discover and protect your sensitive data on AWS. To verify that your PCI DSS scope does not extend past the S3 log bucket, you can configure Macie to monitor objects for account data in those buckets and provide an alert if sensitive data is identified.

Some abstracted services, such as Elastic Load Balancing (ELB), do maintain a persistent network connection to the resources in the target groups, and you must evaluate these services as part of your scoping process.

Unlike traditional virtual servers in a subnet, which have default network connections available to each other, abstracted service instances are segmented by default. The connections between these services are on-demand data connections rather than persistent network connections. For scoping, the focus is placed on the type of data traversing the connection, and the configuration and permissions of the abstracted service to allow data in or out. For example, a Lambda function will only communicate based on the code that you provision. That same function can only be communicated to, or *connected to*, using explicitly allowed IAM permissions and configurations.

If an abstracted service does not handle account data, does not impact the security of the CDE, and does not communicate with a resource that handles account data, you can demonstrate that you have content-aware controls to sufficiently show that account data cannot travel beyond your CDE through these abstracted services. This will allow you to exclude it from your PCI DSS scope. Abstracted services that communicate to an account data storage location, such as collecting system logs or metrics from a CHD-containing database, would still be considered *connected to*.

You must adopt proper architectural designs, including data filtering and monitoring, to help ensure that these services do not store, process, or transmit account data, and therefore can be safely considered out-of-scope for the PCI DSS.

Scoping guidance for hybrid environments

As your organization migrates workloads to the AWS Cloud, you might run a hybrid infrastructure. You might have workloads that are running both in your on-premises data centers and on AWS. Your segmentation controls must consider this hybrid infrastructure and communication between on-premises and AWS Cloud environments.

Consider the following scenarios:

1. **Scenario 1:** PCI DSS resources hosted on AWS have network connections to resources in an on-premises data center.
2. **Scenario 2:** PCI DSS resources hosted in an on-premises data center have network connections to resources on AWS.
3. **Scenario 3:** PCI DSS resources hosted both in an on-premises data center and on AWS.

For scenarios 1 and 2, the scope of the non-CDE resources (regardless of where they are hosted) is determined by the type of connectivity that these non-CDE resources have with in-scope systems.

- If the non-CDE resources have direct connectivity to CDE resources, then those on-premises or AWS resources are in-scope for PCI DSS and become part of the CDE.
- If the non-CDE resources have a connection to systems that are considered *connected to*, then those on-premises or AWS resources can be considered out of scope for PCI DSS as long as they do not provide security or management services. Further, if those out-of-scope systems are compromised, there should be no way for those systems to be used to further compromise the CDE. For example, AWS Systems Manager can manage both AWS Cloud and on-premises servers and in-scope EC2 instances. In this scenario, the on-premises servers are potentially out of scope provided they do not interact with account data or have other connections to the CDE.

To limit scope, consider the following guidelines:

- Restrict network connections from on-premises networks to only non-CDE resources.
- If connectivity to CDE resources is required, implement proper security controls and designs to prevent transitive network connections, which might bring unwanted network and system components in-scope.

- Implement multiple segmentation controls to help ensure that scope boundaries are not altered erroneously. You can implement these controls through a combination of access control rules defined and implemented using on-premises stateful firewall technologies, security groups, and network access control lists (network ACLs), and backed up by IAM permissions to achieve defense in depth.

For scenario 3, you should group CDE resources into either on-premises or AWS Cloud environments for simplicity of scope determination. If this grouping is not workable, carefully verify that CDE, *connected to*, and *security impacting* systems are identified both on-premises and on AWS and that the connections to these systems are correctly identified.

Decision flow – PCI DSS scope identification

A decision flow can help you correctly identify the PCI DSS scope in your organization. This process starts with correctly identifying the flow of account data in your organization's environment.

When you have correctly identified the account data flow, the next step is to identify in-scope resources in your environment.

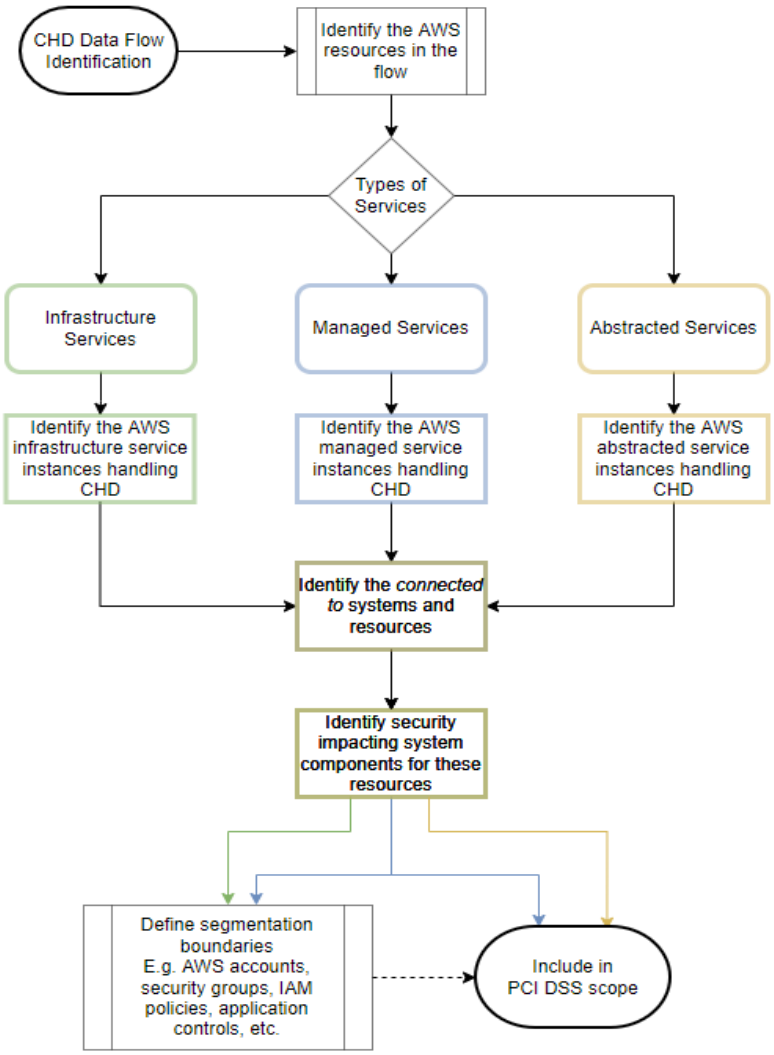

Figure 2: Decision flow to correctly identify the PCI DSS scope and associated segmentation boundaries

The preceding figure shows the decision flow that you can follow to correctly identify the PCI DSS scope. It can also help you correctly define segmentation boundaries at different stages of the flow to segregate other AWS resources from in-scope resources and reduce the PCI DSS scope.

Step 1: Identify the CHD and SAD flow

Before you can define your PCI DSS CDE and design segmentation boundaries, you must have an accurate understanding of the CHD and SAD flow in your organization. To correctly identify

the account data flow, you must identify and define the entire lifecycle of account data in your organization. This includes the entry of account data into your environment; the subsequent transmission, processing, and storage of the account data; and the eventual secure destruction, devaluation, or exit of the account data from your environment.

Step 2: Identify in-scope resources in your environment

Identify the various AWS resources that make up the account data flow. These resources are those that receive, process, store, or transmit account data. It is critical to define what is and is not in-scope as part of your analysis so that your company's ISA, external QSA, or both can clearly understand what services their assessment should be limited to when conducting their audit.

Step 3: Categorize the systems and services

This step categorizes systems and resources into infrastructure, managed, and abstracted services. The scope identification and segmentation of these resources are based on different connections. Infrastructure and managed services primarily communicate with each other over a network (OSI layer 3-4) connection, and communication to abstracted services is often through the service's API (OSI layer 7). After you identify the different AWS resources, you can identify the PCI DSS scope.

Step 4: Identify connected to resources

After you have defined the systems and services involved with the account data flow, identify resources that have connections to or from them. These connections might be for purposes like operational monitoring, logging, or configuration management. Both persistent and on-demand connections must be reviewed, such as from an infrastructure service like an EC2 instance or to an abstracted service like an S3 bucket. These *connected to* resources are part of the CDE.

Step 5: Identify security impacting resources

After you have defined *connected to* system components, you should identify resources and AWS services that provide security services or could otherwise impact the security or configuration of the CDE. This includes a wide range of potential system categories, such as system components that are used to satisfy a particular PCI DSS requirement like logging, or services like DNS that are used to support an environment. Many types of system components could fall into this category. For more information, see page 12 of the PCI SSC Information Supplement: Guidance for PCI DSS Scoping and Network Segmentation.

Step 6: Design segmentation boundaries

After you have identified the in-scope AWS resources, you should design segmentation boundaries to help ensure that other AWS resources are properly segmented to exclude them from the PCI DSS scope. For AWS abstracted services, this segmentation is primarily controlled by the application and associated application code controlling the flow of the account data. For infrastructure service resources like those of Amazon EC2, and managed services resources like those of Amazon RDS, you must also design network-level segmentation along with application-level segmentation.

AWS Cloud considerations for solutions

The AWS Cloud has benefits such as scalability, disposable resources, traceability, security control automation, and continuous validation and testing. AWS also offers various business advantages like the ability to shift from a fixed cost model to a variable cost model, and benefits such as the ability to pay only for the individual services that you need and only for as long as you use them, and no long-term business contracts. These unique characteristics make cloud adoption advantageous for most organizations.

For more information on AWS economic benefits, see the Cloud Economics Center. You can realize these benefits when you use these cloud-specific characteristics to design your payment application infrastructure. To learn about additional benefits of AWS, see the AWS Well-Architected Framework and Overview of Amazon Web Services.

Shared Responsibility Model

Security and compliance are shared responsibilities between AWS and their customers. This shared model can help relieve your operational burden: AWS operates, manages, and controls the components from the hypervisor host operating system and virtualization layer down to the physical security of the facilities in which the service operates. Primarily, AWS is responsible for the security *of* the cloud, and you are responsible for protecting the confidentiality, integrity, and availability of your data *in* the cloud. Your responsibility includes meeting your specific business requirements for information protection.

Make sure that you understand the shared responsibility model before you embark on your compliance journey. The shared responsibility varies depending on the level of abstraction provided by a particular AWS service. As service abstraction increases, your responsibility decreases. Evaluate every service used in your environment and understand the impact of using the service on your overall PCI DSS scope. This approach can help you understand what you must do to meet your compliance obligations.

Virtualization of traditional network controls

The software-defined networks (SDN) that AWS provides mimic traditional networking constructs, but require a new way of viewing and managing your networks. For example, traditional on-premises network controls like virtual local area networks (VLANs) are implemented differently on AWS because layer 2 networking is transparent to you.

On AWS, Amazon Virtual Private Cloud (Amazon VPC) represents a logically isolated section of the AWS Cloud, where you can launch resources in a virtual network. It is common for resources that provide similar functionality or similar scope to span across multiple subnets across different Availability Zones (AZs) to provide redundancy. Therefore, Amazon VPCs and subnets should be used as grouping constructs and not segmentation controls.

Elasticity

AWS resources allocated to an application, such as compute or storage, can be scaled horizontally based on demand. AWS Auto Scaling monitors your applications and automatically adjusts compute capacity to maintain steady, predictable performance at the lowest possible cost. Such AWS resources can be short-lived, potentially measured in minutes or hours rather than months or years like their physical on-premises counterparts. Other AWS managed services or features, such as Lambda and ELB, scale vertically to accommodate the resource requirement. The segmentation controls that you design must be able to accommodate the elastic and transient nature of the cloud environment. Design these controls so that they remain enforced as the infrastructure changes; otherwise, it could lead to an incorrect scope definition. It is important to note that when resources scale horizontally, such as EC2 instances that are part of an AWS Auto Scaling group, the population of PCI DSS in-scope resources scales with it, either up or down.

Abstracted services and API-based infrastructure

Many AWS offerings are provided as managed or abstracted services. This means that AWS manages much of the underlying infrastructure for you, and you do not have control over it. Many abstracted AWS services are only communicated with through the API endpoint of the service over HTTPS. AWS service APIs include inherent network segmentation controls, in addition to other controls such as authentication, authorization, and data integrity. This verifies that only data from authorized entities is exchanged between the calling system and the service. When configured to do so, these services communicate among themselves and other AWS services over access-controlled APIs. This configuration is provided as part of the service and meets the layer 3-4 network security control provided by firewalls. You should use AWS abstracted services to reduce the *connected to* scope of your environment. For example, select a log consolidation service like Amazon CloudWatch that makes encrypted API calls to forward log data, rather than an agent that maintains an open TCP/IP connection.

When you configure abstracted services like Lambda or ELB for account data flows, you must also configure secure connectivity such as HTTPS using TLS version 1.2 or later. If an AWS abstracted service instance stores, processes, or transmits account data, the resources that it is configured to connect to are also PCI DSS in-scope. You must design application layer-based segmentation and traffic filtering controls as part of your responsibility under the shared responsibility model.

Automation

With automation, you can implement most infrastructure and application changes without manual intervention. This provides agility, decentralizes the change management process, and expedites the deployment process. You must also automate the segmentation controls to the greatest extent possible so that the segmentation controls are applied in tandem with changes to the infrastructure and applications. This approach preserves the defined scope boundary. Automation can also help detect when segmentation controls are changed so that you can implement proper remediation steps, such as re-establishing the controls or alerting someone in near real time to analyze the cause and effect of changes.

Design segmentation for the cloud

This section explains the various segmentation boundaries that you can design based on the principle of using cloud features to protect cloud services and achieve defense in depth. You can achieve these boundaries at various layers on AWS and then combine them with each other to reduce the PCI DSS in-scope systems to the minimum required for your secure and functional CDE.

AWS account layer

An individual AWS account provides the highest level of segmentation that you can achieve on AWS. By design, the resources provisioned within an account are logically isolated from the resources provisioned in other accounts, even within your own organization in AWS Organizations. When you design your AWS environment, it is a best practice to use isolated accounts for PCI DSS workloads. You can reduce your PCI DSS scope by segmenting in-scope and out-of-scope resources into their own accounts.

An AWS account acts as an identity and access management isolation boundary. This helps prevent accidental scope creep because logical account-level isolation can only be changed by establishing explicit communication channels between resources in separate accounts. This approach also helps reduce the impact by not allowing changes to architecture and controls in out-of-scope accounts from adversely affecting the security of in-scope resources in other accounts. There are some additional operational benefits to using multiple AWS accounts. This includes distributing the AWS service quotas (limits) and request rate limits (throttling), because

these are allocated for each account. For more details, see Benefits of using multiple AWS accounts.

The following diagram shows a proposed multi-account architecture in relation to PCI DSS scope and segmentation:

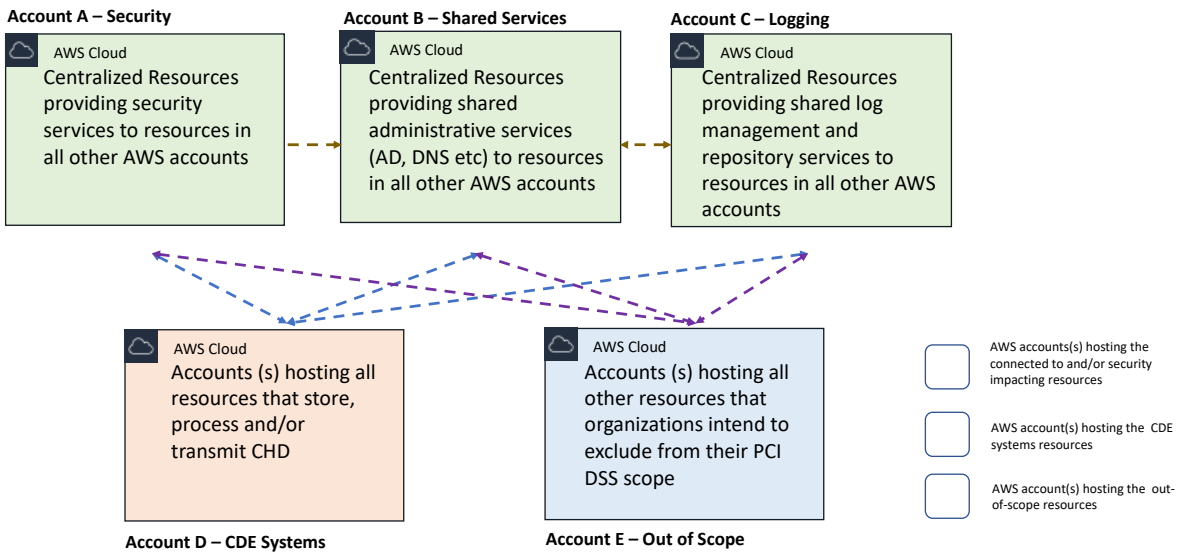

Figure 3: Multi-account architecture to restrict PCI DSS scope

Components of a multi-account architecture

Figure 3 shows a multi-account architecture designed according to AWS best practices, as published in the Organizing Your AWS Environment Using Multiple Accounts whitepaper. Resources that provide similar functionality are grouped within accounts. The accounts are further grouped under organizational units (OU) within a root. This grouping enables the sharing of resources that provide security and management functionality among in-scope and out-of-scope AWS resources, while verifying that the PCI DSS scope is restricted. This arrangement also supports separation of duties and helps enforce least privilege for the teams that need to operate in these accounts. OUs provide a way for you to organize your accounts so that it is simpler to apply common overarching policies to accounts that have similar needs. An example of an organizational policy is an AWS Organizations service control policy (SCP). SCPs can help ensure that your accounts stay within your access control guidelines by restricting the permissions that IAM policies can grant to entities in an account, such as IAM users and roles. For example, you can use SCPs to prohibit provisioning of non-PCI DSS compliant AWS services within an OU that is composed of PCI DSS in-scope accounts.

In our current scenario, the accounts are grouped together under different OUs based on the PCI DSS scope of the resources, CDE systems, *connected to* and *security impacting* systems, and out-of-scope systems.

Recommended OUs and accounts

This section provides details on the recommended OUs and accounts, based on the AWS multi-account strategy guidance, while adjusting the grouping names to help align with PCI DSS scoping. You should name your OUs according to your organization's naming conventions.

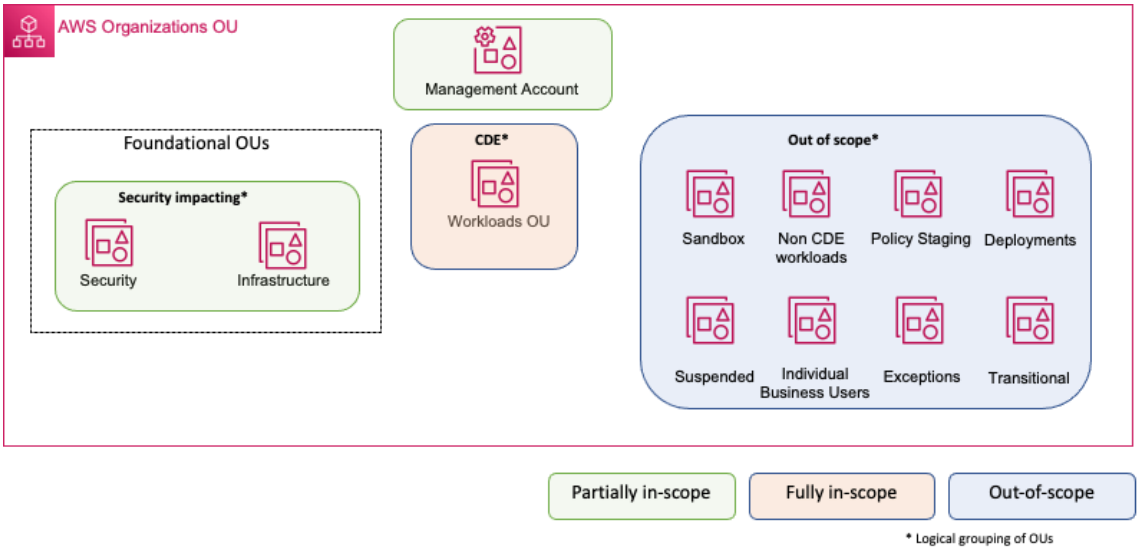

Figure 4: Recommended AWS Organizations OU logical grouping with respect to PCI DSS scope

Although the recommended OUs are aligned toward common use cases, you can define your own OU structure that best meets your needs. This guidance should align with the needs of most customers; however, it is not one-size-fits-all. Depending on your requirements, you might not need to establish all of the recommended OUs.

The recommended OU categories with respect to PCI DSS scoping include the following:

- PCI – This is the OU category that groups accounts hosting the CDE, workloads, and resources that directly store, process, or transmits account data, along with resources that are categorized as *connected to* systems. This OU is usually a child OU under the top level workloads OU.
- Security impacting – This is the category of OUs that groups accounts that provide common security and shared services capabilities to help secure and support your overall AWS environment, including your CDE.
- Out-of-scope – This is the category for other OUs that group accounts that host resources and workloads deemed to be out-of-scope for PCI DSS.

Management account

AWS Organizations is an account management service that you can use to merge multiple AWS accounts into an *organization* that you create and centrally manage. By using the AWS Organizations account management and merged billing capabilities, you can better meet the budgetary, security, and compliance needs of your business.

You should consider the management account in-scope for PCI DSS because it impacts the security of your organization and your CDE. We recommend that you use the management account only for tasks that can only be performed by that account. Store your AWS resources in other accounts in the organization and keep them out of the management account.

One important reason to keep your resources in other accounts is because AWS Organizations SCPs do not work to restrict users or roles in the management account. For more information, see Best practices for the management account.

Workloads OU

The Workloads OU is intended to house most of your business-specific workloads including both production and non-production environments. This OU consists of multiple child OUs grouped by business unit or team and software development lifecycle (SDLC) environment (production or non-production). You should dedicate one such child OU to group the accounts hosting your CDE; this child OU could include connected-to workloads and resources. This dedicated OU would be the focal point for your PCI DSS scope, and you should define organization policies that are implemented by using SCPs. This layered structure helps to ensure that the security controls required to meet PCI DSS requirements, including segmentation controls, for accounts hosted under this OU are highly controlled and restricted. In this paper, we have named the OU as the *PCI OU* and the account hosting your PCI workload as the *PCI application account* for simplicity. However, you can name these according to your organizational naming conventions.

You can segregate your CDE and *connected to* resources into separate PCI application accounts or combine them in the same account. You should base this design on your workload requirements. If you are mixing these types of resources, you can use additional network- and application-level controls to segregate these resources from each other. For this guidance, we are only using one account. You should only use PCI application accounts to provision resources that are in-scope for PCI DSS, unless required by your workload design.

These accounts include hosting the following resource types:

- Compute and network resources that receive or transmit account data, from and to external entities
- Compute resources that process account data
- Resources that store account data at rest

- Resources that establish a network- or application-level connection to the preceding resources

Your PCI application accounts are the most sensitive part of your PCI DSS infrastructure. System components in these accounts can bring other resources in scope. This means that any system components that these accounts connect to have the potential to be considered in-scope, regardless of the functionality or the data involved in the communication. Communication to and from these accounts should be highly restricted (both logical and network access) to only necessary resources in other accounts, and based on system management or business requirement purposes. You should also implement change management processes to help ensure that changes in these accounts do not negatively impact the segmentation boundaries and the overall PCI DSS scope of the organization. AWS Config can track changes to a particular AWS resource, which could then cause an Amazon CloudWatch event to generate an alert, followed by a Lambda function to automatically remediate security control deviations or orchestrate a variety of other steps to implement change control processes.

You can group your other AWS accounts that host non-PCI workloads into one or more separate child OUs under the workload account, without bringing them in-scope for PCI, as long as you make sure that resources within the accounts associated with the OU meet both of the following criteria:

- Impact the security of the CDE systems accounts
- Have connections (logical access or network) with resources in CDE systems accounts

As noted previously, AWS accounts by design provide logical isolation, and therefore, these accounts isolate your resources related to non-PCI workloads from resources related to PCI workloads (unless you explicitly introduce a cross-account network or application connection). In this paper, we refer to these OUs as *non-PCI OUs*; and the account(s), as *non-PCI application accounts*.

You can have additional child OUs to host test accounts that your application team owns and uses to develop, test, and stage workloads. PCI DSS v3.2.1 Requirement 6.4.5.3 and PCI DSS v4.0 Requirement 6.5.2 require that all changes to system components in production environments be tested to verify that they do not adversely impact system security. However, you can safely exclude these non-PCI production resources from your PCI DSS scope if you make sure that resources within the accounts do not have connections to CDE resources or impact the security of CDE resources.

The following is an example structure of a workload OU with associated accounts and their PCI DSS scope:

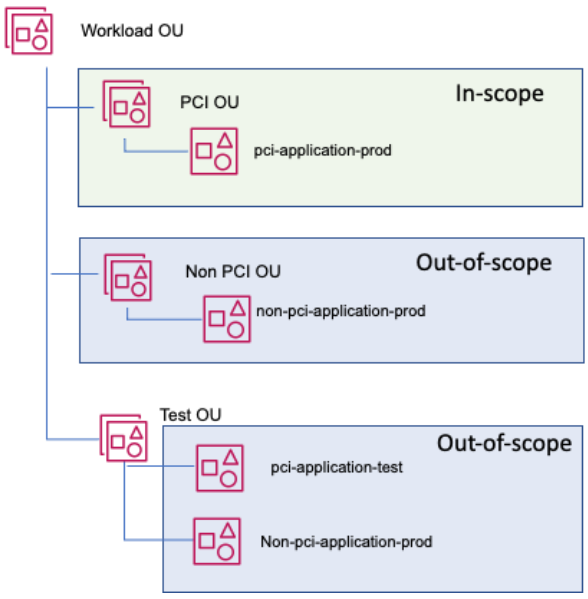

Figure 5: Recommended AWS Organizations member account structure for the workload OU

Foundational OU

This is a logical collection of one or more OUs and sub-OUs to group accounts that provide security and management services to CDE resources. Your centralized cloud environment or cloud engineering teams, made up of cross-functional representatives from your security, infrastructure, and operations teams, typically own the accounts, workloads, and data that reside in the foundational OUs. The AWS multi-account recommendation outlines two distinct OUs: security and infrastructure.

Security OU

You should group accounts that provide security services to the CDE accounts together in the security OU. Your security organization should own and manage this OU, along with child OUs and associated accounts. For your PCI environment, use the security OU to host resources that provide security services to CDE systems. It is recommended that you have sub-OUs to host non-production accounts for testing.

The following are the recommended accounts in the security OU:

- **Log archive** – This account acts as the audit trail consolidation point for log data collected from the accounts in the organization. This includes logs from the CDE system accounts. You should consider the log archive account to be in-scope because this account helps fulfill several controls outlined under Requirement 10 of both PCI DSS v3.2.1 and PCI DSS v4.0.

- **Security tooling** – This account groups broadly applicable, security-oriented workloads. These might include anti-virus or anti-malware services, vulnerability scanning services, and intrusion detection systems (IDS) and intrusion prevention systems (IPS). You should consider the security tooling account to be in-scope because this account can help fulfill a number of controls outlined under Requirements 2, 3, 5, 6, 7, 8, and 11 of both PCI DSS v3.2.1 and PCI DSS v4.0.

As outlined in the workload OU, you can consider the test sub-OU and associated accounts to be out of PCI DSS scope as long as you make sure that resources in the accounts do not have connections to CDE resources and cannot impact the security of PCI in-scope resources. The following figure shows an example structure for the security OU, with associated accounts and their PCI DSS scope:

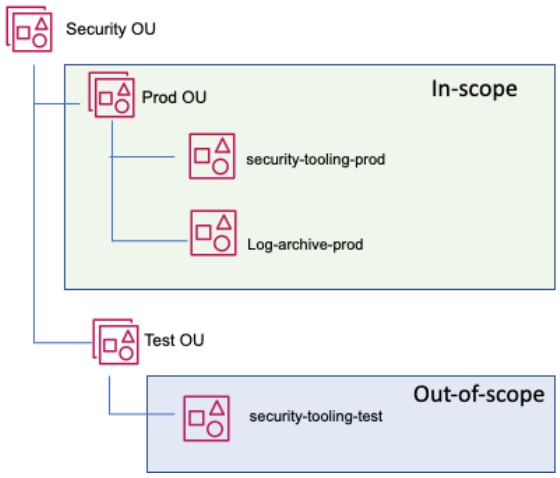

Figure 6: Recommended AWS Organizations member account structure for the security OU

Infrastructure OU

The infrastructure OU is another foundational OU and should contain services to support shared infrastructure. Your infrastructure teams should own and manage this OU and child OUs and associated accounts. For your PCI environment, use this OU to host resources that provide management functions to CDE systems. Common use cases for this OU include shared services such as hybrid DNS infrastructure and directory services. Similar to the security OU, the recommended best practice is to separate the production and test resources through production and test child OUs.

The following is the recommended account for the infrastructure OU:

- **Shared services** – This account supports the services that multiple applications and teams use to deliver their outcomes. For example, directory services like AWS Directory Service for Microsoft Active Directory, messaging services, and metadata services are in this category. You should consider this account to be in-scope because it can help fulfill controls laid out in Requirements 2, 3, 5, 6, 7, 8, and 11 of both PCI DSS v3.2.1 and PCI DSS v4.0.

> The AWS multi-account guidance whitepaper recommends a centralized networking account under the infrastructure OU. This is recommended to be the central hub for your network on AWS hosting network resources that route traffic between accounts and egress or ingress traffic to the internet. To reduce PCI DSS scope, do not centralize network traffic to and from your CDE systems, but rather host it locally as part of your CDE accounts. If you centralize network resources that handle account data traffic, you could extend the PCI DSS scope into and through the centralized network account.

As outlined for the workload OU, you can consider the test sub-OU and associated accounts to be out of scope for PCI DSS as long as you make sure that resources in the accounts do not have connections to CDE resources and do not impact the security of PCI in-scope resources.

The following figure shows an example structure for the infrastructure OU, with associated accounts and their PCI DSS scope.

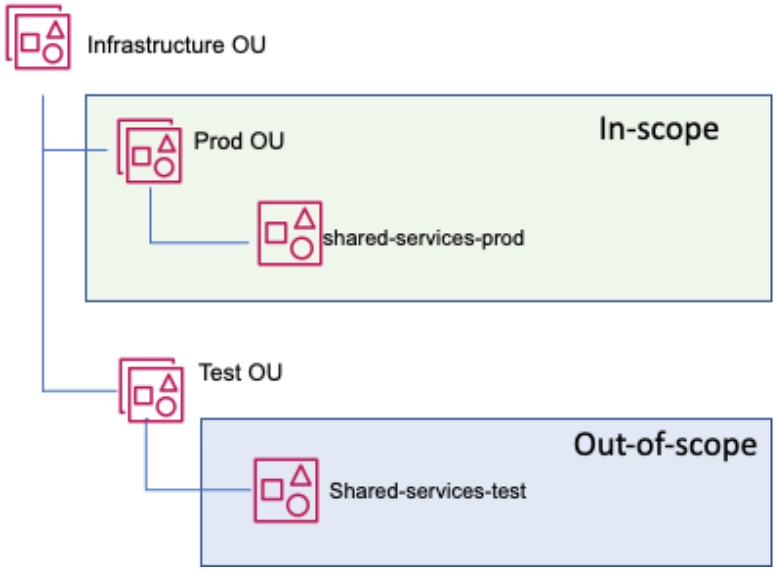

Figure 7: Recommended AWS Organizations member account structure for the infrastructure OU

Out-of-scope OUs

Use separate AWS accounts to provision IT infrastructure that does not require connectivity to or does not provide services for CDE systems. This separation helps ensure that resources in these accounts are adequately isolated from PCI in-scope systems by design, through inherent AWS account isolation. You can group these accounts under separate individual OUs according to business functions and needs. For recommendations on other types of OUs, see Organizing Your AWS Environment Using Multiple Accounts.

AWS Control Tower

AWS Control Tower is designed to help you set up and govern a secure, multi-account AWS environment called a *landing zone*. AWS Control Tower creates your landing zone by using AWS Organizations, which brings implementation best practices with ongoing account management and governance. With AWS Control Tower, you can provision new accounts with a few clicks while making sure that the accounts conform to your organizational policies. You can even add an existing account to a new AWS Control Tower environment. AWS Control Tower orchestration extends the capabilities of AWS Organizations. To help protect your organizations and accounts from *drift*, or divergence from best practices, AWS Control Tower applies preventive and detective controls, called guardrails. For example, you can use guardrails to help ensure that security logs and necessary cross-account access permissions are created and not altered.

If you use AWS Control Tower, you should consider it to be in-scope for PCI DSS because it orchestrates and governs the multi-account environment where your CDE resides.

> AWS Control Tower names the account that is under the security OU the *Audit Account* by default. You can rename this account during the AWS Control Tower setup.

For more information, see Best practices for AWS Control Tower administrators.

Network layer (OSI Layer 3-4)

A security group is a feature of Amazon VPC that provides stateful network layer traffic filtering that works on the principle of an implicit deny. Only traffic defined in a security group can pass. A security group is similar to a host-based firewall and is associated with the network interface of an Amazon EC2 instance. You can use security groups to restrict network communication based on the port, and source and destination addresses required to segment CDE systems from other *connected to* systems. Security groups are the core segmentation boundaries in an Amazon VPC, and you should design your network layer PCI DSS scoping strategy around these security groups. With AWS software-defined networking capabilities, you can bring these stateful firewalls “closer” to your EC2 instances, and attach them directly to your network interfaces. Moving resources “closer” on the network means reducing the length of the network

path between two resources. This reduces the number of network interconnection possibilities that might impact the *connected to* scope, and potentially reduces the network latency. This means that neighboring EC2 instances might not be considered *connected to* and in-scope for PCI DSS if you have configured the security groups to restrict traffic. In traditional on-premises environments, a single host that contains account data would bring its entire subnet into scope, because networking boundaries were present further up the network stack. You can also use security groups to restrict traffic flow between instances to help meet PCI DSS v3.2.1 Requirements 1.2 and 1.3 and PCI DSS v4.0 Requirements 1.3.1, 1.3.2, and 1.4.1.

You can attach security groups to Auto Scaling groups so that they apply to, and are removed from, instances in the group when the groups scale out and in. Between peered Amazon VPCs, you can chain together security groups so that one security group can reference the other security group as the source or destination, instead of using hard-coded IP addresses or ranges. This design helps automate the security group architecture and provides scalability by controlling peering traffic through security group membership instead of Classless Inter-Domain Routing (CIDR) ranges.

The default outbound configuration for a newly created security group does not meet PCI DSS v3.2.1 Requirement 1.2.1 or PCI DSS v4.0 Requirement 1.3.2 because it permits all outbound traffic. You must modify the default configuration for the security group to limit your outbound traffic to allow only necessary traffic.

You can enable connections between your Amazon EC2 instances in out-of-scope AWS accounts with a *connected to* AWS account without impacting your overall PCI DSS scope. Because Amazon VPC peering connections are non-transitive, the peering connection between CDE accounts and *connected to* system accounts does not extend connectivity and scope into those out-of-scope accounts as long as there is no explicit peering connection between CDE accounts and out-of-scope accounts.

The following diagram shows an example of how to use security groups to achieve segmentation. As shown, security groups primarily define the network segmentation regardless of other network constructs, such as VPCs and network segments.

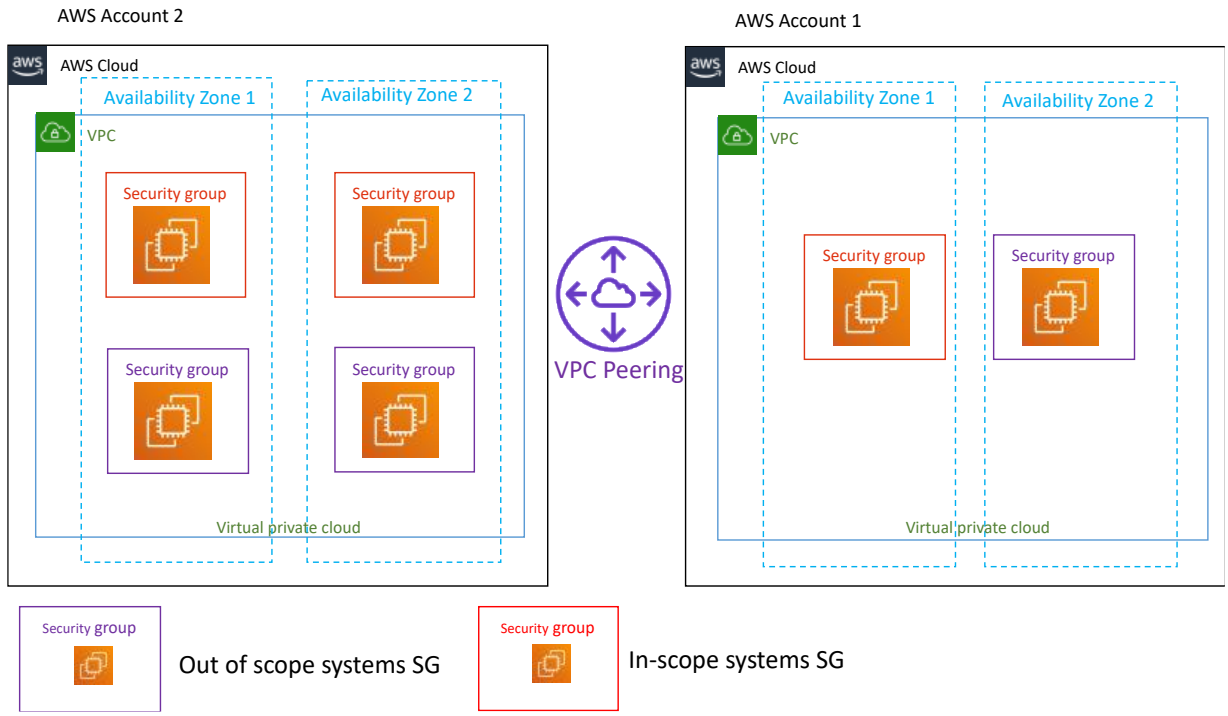

Figure 8: AWS VPC reference architecture

You should use one AWS account to provision your PCI DSS in-scope resources. However, you should control security group rules centrally for your CDE to help ensure that a change does not negatively affect the PCI DSS scope. You can achieve this by using AWS Firewall Manager, which provides you with the ability to centrally manage security groups across multiple accounts. With AWS Firewall Manager, you can group resources by account, resource type, and tag to help you group your in-scope security groups and enforce policies across them. AWS Firewall Manager helps you apply policies hierarchically, so you can delegate the creation of application-specific rules while retaining the ability to enforce certain rules centrally. AWS Firewall Manager constantly monitors centrally-applied rules for accidental removal or mishandling to verify that the rules are applied consistently. With AWS Firewall Manager, you can create policies that set guardrails defining which security groups are allowed or disallowed across your VPCs. You can get notifications of accounts and resources that are non-compliant or set up AWS Firewall Manager to act directly through auto-remediation. Incorporating the ability of AWS Firewall Manager to identify unused, redundant, or overly permissive security groups into your semi-annual firewall rule review process can help you meet PCI DSS v3.2.1 Requirement 1.1.7 and PCI DSS v4.0 Requirement 1.2.7.

If your organizational security controls require more advanced firewall capabilities or the centralization of ingress and egress traffic inspection within a VPC, consider using AWS Network Firewall. AWS Network Firewall is a stateful, managed network firewall and intrusion

detection and prevention service for your VPC. With AWS Network Firewall, you can filter traffic at the perimeter of your VPC. This includes filtering traffic to and from an internet gateway, NAT gateway, or over VPN or AWS Direct Connect. AWS Network Firewall uses the open-source intrusion prevention system (IPS) engine Suricata for stateful inspection. Usage of AWS Network Firewall is not mandatory to meet PCI DSS requirements for network segmentation. However, it provides additional security benefits of monitoring and protecting your VPC traffic in the following ways:

- Only allows traffic from known AWS service domains or IP address endpoints, such as Amazon S3
- Uses custom lists of known bad domains to limit the domain names that your applications and resources can access
- Performs deep packet inspection on traffic entering or leaving your VPC
- Uses stateful protocol detection to filter protocols like HTTPS, independent of the port used

Application layer (OSI Layer 7)

At this layer, the application that handles account data must manage the data flow and define segmentation boundaries. AWS provides many API-based abstracted services, such as Lambda, Amazon S3, and DynamoDB, that your organization can use to enable business functionality without the need to manage servers and Amazon EC2 instances. You can only communicate to most abstracted services through the service's encrypted API over HTTPS. For communication to be possible, you must also explicitly allow access to a service by using IAM policies. When you use AWS abstracted services, IAM policies become application layer logical segmentation boundaries based on the implicit deny with IAM policies and permissions. It is the customer's responsibility to design application-layer segmentation to reduce the PCI DSS scope when using abstracted AWS services.

Segmentation for abstracted AWS services

Abstracted services implement network isolation by design. Unlike traditional virtual servers in a subnet that have default network connections available to each other, abstracted service instances are segmented by default and only establish connectivity based on permitted events. For scoping, the focus is placed on the type of data traversing the connection, and the configuration and permissions of the abstracted service to allow data in or out. For example, a Lambda function will only ever communicate based on the code that you provision, and connect to only those resources that you explicitly configure. That same function can only be communicated to, or *connected to*, using explicitly allowed IAM permissions and configurations.

Segmentation with Amazon API Gateway

For customer-defined web API-based services, such as serverless applications, you can use Amazon API Gateway to broker connections between the CDE resources and services, and other web-based services. API Gateway can act as a “front door” to applications for accessing data, business logic, or functionality from backend services. This could include workloads running on Amazon EC2, code running on AWS Lambda, other supported AWS services, or web or mobile applications. You can use API Gateway as the CDE boundary interface to broker communication from CDE systems and services hosted on AWS. Here, the connections from CDE systems and services are terminated by your custom API instance, and a new connection is established from your API instance to the non-CDE destination system or services. Then, you can exclude from your PCI DSS scope resources outside of your AWS CDE that communicate with CDE systems through API Gateway, as long as the resource does not handle account data or provide security services to CDE systems. Only the API Gateway instance is in-scope as a connected system. Because this is a PCI DSS validated service, you do not need to maintain and validate the service itself and its ability to act as a segmentation boundary for PCI DSS compliance. You are still responsible for other aspects of the implementation as part of the Shared Responsibility Model. This includes, but is not limited to, requirements for access management with IAM and logging with AWS CloudTrail. Although API Gateway provides a secure, access-controlled web service API mechanism, applications validate incoming data, including monitoring for unexpected account data from a CDE.

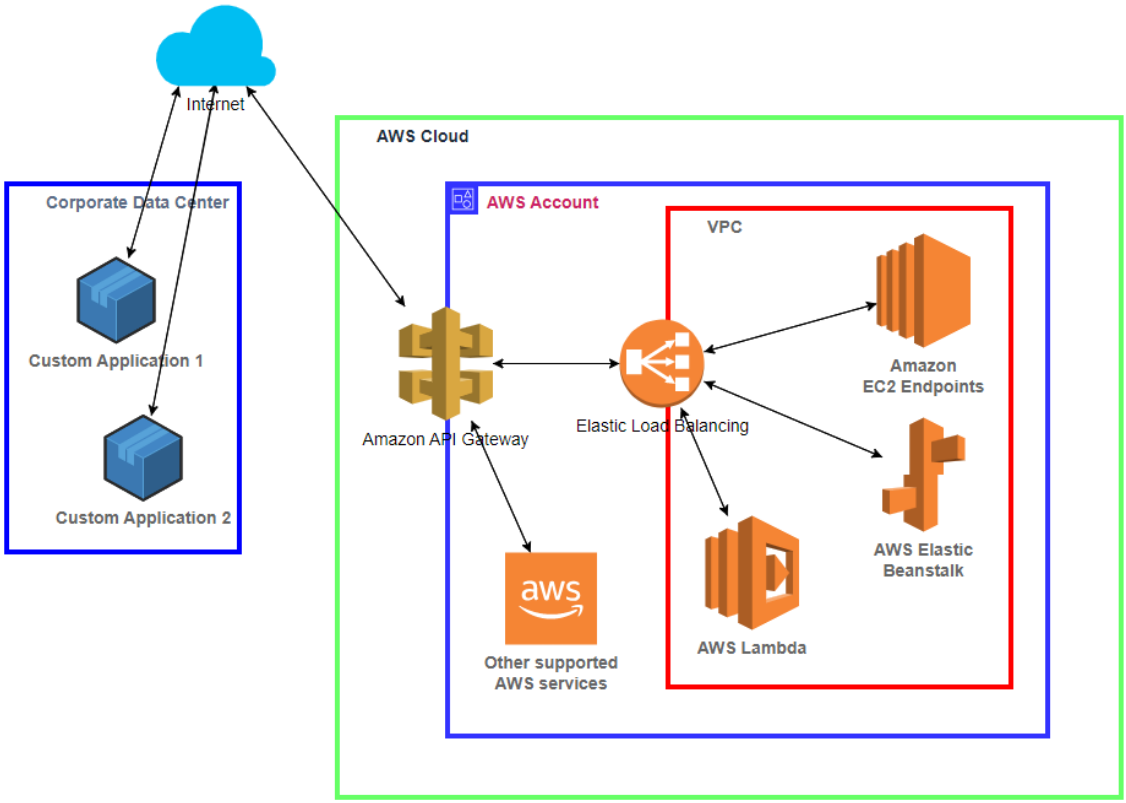

Figure 9: Segmentation using API Gateway for abstracted services

Scoping and segmentation for containerized workloads

Organizations use containers to quickly, reliably, and consistently deploy applications independent of the underlying infrastructure.

Containers allow you to create a self-contained standardized unit of software development that includes an application's code and related configurations and dependencies. You can use containers as scalable building blocks for a larger business application, to help deliver consistency in environments, support robust version control, and increase developer and operational efficiency. Amazon ECS and Amazon EKS are highly scalable, highly performant container orchestration services that allow you to run and scale containerized applications on AWS. AWS Fargate is a serverless compute engine compatible with both Amazon ECS and Amazon EKS and provides a greater level of abstraction, scope reduction, and segmentation than EC2 instances. If you are running or architecting containerized PCI DSS in-scope

applications, you must verify that they are properly scoped and segmented. For specific guidance, see the following AWS whitepapers.

- Architecting on Amazon ECS for PCI DSS Compliance
- Architecting Amazon EKS for PCI DSS Compliance

Container scoping

Amazon ECS tasks and Kubernetes pods on Amazon EKS are logical groups of running containers for the respective services. Both Amazon ECS and Amazon EKS support two launch types:

- EC2 launch type: This type allows you to run your containerized applications on a cluster of EC2 instances that you manage.
- Fargate launch type: This type allows you to run your containerized applications without the need to provision and manage the backend infrastructure.

As described previously, you should use the PCI application account under the PCI OU to deploy your containerized workloads that handle account data. Then use a separate account under out-of-scope OUs to host containerized workloads that are isolated from in-scope containers. If you have containerized workloads that need to connect to in-scope workloads, but whose communication does not involve account data, you should consider hosting them in accounts under the PCI OU or in shared services accounts that contain other *connected to* or *security impacting* resources.

You can use AWS Cloud Map to dynamically discover resources and maintain an inventory of them, and assist with defining the scope of your containerized environment. AWS App Mesh can also help define your scope by maintaining the traffic flow and assisting with your data flow diagrams. You can use App Mesh to dynamically update traffic routing between your CDE services.

When technical or business constraints prevent you from using account-level segregation of containerized workloads, you should create separate ECS and EKS clusters and VPCs to group in-scope and out-of-scope containers. Customers should be aware that having in-scope and out-of-scope resources in the same AWS account imposes additional administrative and logical constraints in order to successfully keep the desired resources in that PCI account out of scope. Processes and resources that support these out-of-scope resources themselves are potentially in-scope because they have the ability to impact the security of the CDE. This includes, but is not limited to, access provisioning procedures for IAM permissions in the account, deployment processes that provision the out-of-scope resources that exist adjacent to in-scope resources, change management procedures for changes to out-of-scope resources in the in-scope account, and change detection for those supporting resources. The AWS account that contains in-scope clusters is part of the CDE, so you will need to demonstrate that even if the supporting processes that maintain the out-of-scope resources are compromised they *could not* impact the security of the CDE. This might be a difficult conversation with your QSA. They must agree that

your segmentation and separation efforts sufficiently rule your desired resources as out-of-scope after viewing evidence that supports your assertion.

Customers should place clusters using the EC2 launch type in separate VPCs to enhance the ability to segment at the network layer. These clusters are your lowest construct for PCI DSS scoping. Use further network controls, as described in the next section, to restrict inter-cluster communication to segment out out-of-scope clusters within the same account.

Segment ECS clusters

An Amazon ECS cluster is a logical grouping of tasks or services. Your tasks and services are run on infrastructure that is registered to a cluster. You can assign security groups to the Amazon ECS cluster and design security group rules to restrict network communication to only in-scope systems, isolate the cluster from out-of-scope system components, or both. Security groups act as a firewall for associated container instances, controlling both inbound and outbound traffic at the container instance level. An Amazon ECS cluster is the lowest construct that you can use to define your PCI DSS scope boundary.

For the AWS Fargate task launch type, tasks are the lowest scoping construct, so you don't have to worry about cluster assignment and scope. Each Fargate task has its own isolation boundary and does not share the underlying kernel, CPU resources, memory resources, or network interface with another task. You should group tasks running in-scope containers and use the awsvpc network mode in combination with security group rules to restrict communication between in-scope and *connected to* tasks.

Segment Amazon EKS clusters

An EKS cluster with the EC2 launch type has the following primary components:

- EKS control plane
- EKS nodes that are registered with the control plane

The control plane runs in an account managed by AWS, and the Kubernetes API is exposed through the EKS endpoint associated with your cluster. Each EKS cluster control plane is single-tenant and unique, and it runs on its own set of Amazon EC2 instances. EKS nodes run in your account and connect to your cluster's control plane through the API server endpoint and a certificate file that is created for your cluster. Your EKS cluster is created in a VPC. The Amazon VPC Container Network Interface (CNI) plugin provides pod networking.

When you create a cluster, Amazon EKS creates a security group named *eks-cluster-sg-my-cluster-uniqueID*. The default rules allow all traffic to flow freely between your cluster and nodes, and allows all outbound traffic to any destination. If you need to limit the open ports between the cluster and nodes, you can remove the default rules and add some minimum rules. Use this cluster security group to segment network traffic from other EKS clusters. An EKS cluster is the lowest construct to define your PCI DSS scope boundary.

For the AWS Fargate task launch type, Kubernetes pods are the lowest scoping construct, so you do not need to perform cluster assignment and design segmentation between pod clusters. Each pod that runs on Fargate has its own isolation boundary. They don't share the underlying kernel, CPU resources, memory resources, or network interface with another pod. You should group pods running in-scope containers and use pod networking in combination with security group rules to restrict communication between in-scope and *connected to* pods

Scoping and segmentation validation

PCI DSS v3.2.1 Requirement 11.3.4 and PCI DSS v4.0 Requirement 11.4.5 state that you must perform penetration and segmentation testing at least annually for merchants and every six months for service providers. This step corresponds to the “Assess and authorize controls” phase of the system lifecycle approach for security and privacy, as outlined in NIST SP 800-37. You should review the AWS Customer Service Policy for Penetration Testing before conducting this kind of security testing.

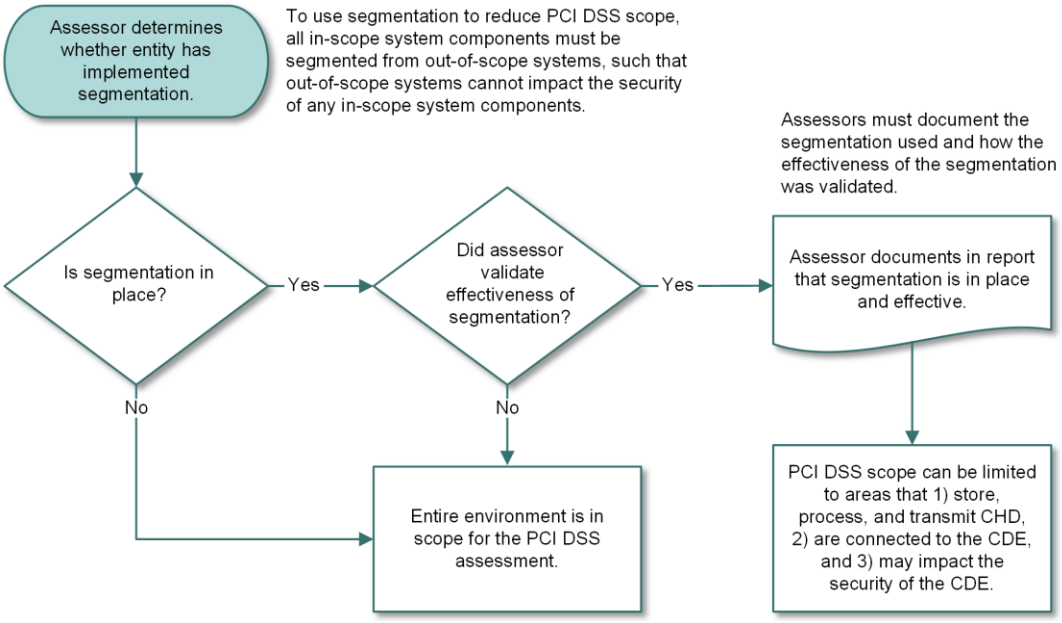

Figure 10: Segmentation and impact to PCI DSS scope

According to the PCI DSS published Information Supplement: Penetration Testing Guidance, the scope for penetration testing includes the entire CDE perimeter and testing critical system components from both inside and outside the network. This applies both to the external perimeter (public-facing attack surfaces) and the internal perimeter of the CDE (LAN attack surfaces). You should also include remote access vectors, such as VPN connections to the CDE. You can use the Amazon VPC feature Reachability Analyzer prior to or as part of your segmentation testing to verify and validate network configurations and connectivity intents.

Whether you use infrastructure, managed, or abstracted services, you must perform some type of testing and validation of segmentation boundaries. Traditional segmentation methods, which are composed of logical connectivity checks between two points inside and outside your CDE, are needed for environments with infrastructure and managed services. For infrastructure services such as Amazon EC2 instances, and managed services such as Amazon RDS, security groups form the core segmentation boundaries.

For managed services, security groups provide similar segmentation boundaries because of the network interfaces involved. The testing aims to validate isolation of in-scope resources from out-of-scope resources. For container clusters with EC2 launch types, you should perform segmentation testing to validate segmentation between in-scope and out-of-scope clusters, and isolation of underlying EC2 instances between the in-scope and out-of-scope clusters. Amazon ECS and Amazon EKS clusters with the Fargate launch type are abstracted services because AWS manages the underlying data plane and is responsible for the security and isolation of underlying compute resources.

For in-scope abstracted services, the only interface with the service is through an AWS service endpoint like *dynamodb.us-east-2.amazonaws.com*. The scanning and penetration testing of these endpoints is covered by the PCI DSS assessment for AWS as a service provider. However, you need to validate that the endpoints are restricted to your specific resources through IAM policies. When you use abstracted services, the account data flow and segmentation boundaries are controlled by the application and the application code. Hence, you should focus on application testing to validate the segmentation boundaries as designed within the application. This focus helps ensure that the account data flow and the PCI DSS scope are maintained by the application as depicted in the account data flow diagram.

For a hybrid environment, the source of segmentation testing can be an out-of-scope network segment of the physical on-premises network; and the target, the in-scope EC2 instances.

Proactive security controls

In addition to the periodic penetration testing mandated by PCI DSS, you must also have proactive security controls in place to prevent unauthorized modification of the segmentation controls.

This section provides information on monitoring the status of implemented segmentation controls. This monitoring helps ensure that the defined PCI DSS scope is not violated intentionally or erroneously, and is required by PCI DSS v3.2.1 Requirement 10.8 (*for service providers only*) and PCI DSS v4.0 Requirement 10.7. You should design the preventive and detective controls so that in case of a violation, respective stakeholders are notified as early as possible and remediation steps can be taken immediately. As your security posture matures, you should automate most responses so that deviations can be remediated without human intervention on a near real-time basis. The following are some ways that you can monitor your segmentation boundaries:

- Periodically validate security group rules against the planned CDE scope
- Manage security group and network configurations with a change control process
- Monitor security group rule changes in the CDE systems account
- Monitor Amazon VPC peering connections to the CDE systems account
- Monitor configuration changes to in-scope APIs or AWS services that allow data in or out of the CDE
- Implement data loss prevention controls on *connected-to* systems to prevent account data leakage and to validate scope boundary

You can automate responses by using AWS Config, which provides you with resource configuration history and configuration change notifications to help with governance and security. You can create custom AWS Config rules to monitor changes to security groups, Amazon VPC peering connections, Amazon API Gateway APIs, and other resources on AWS that enforce segmentation boundaries. Attach AWS Config rules to appropriate Lambda responders to evaluate deviations and initiate automatic remediation when a change violates the defined PCI DSS segmentation boundaries.

You can use AWS CloudTrail to monitor configuration changes for your AWS resources. You can configure CloudWatch events to initiate Lambda responders to take remediation action on your behalf. These are a sample of the various AWS services that you can use to design and automate security controls for your CDE on AWS.

You can also integrate most of these services and other third-party tools and applications with AWS Security Hub. You can activate the PCI DSS standard in Security Hub to help you monitor the status of relevant security controls to help you meet particular PCI DSS requirements. This includes monitoring segmentation controls. When you detect a deviation through Security Hub, you can define further downstream action that might include integrating with your workflow ticketing systems such as Jira, or defining Security Hub custom actions to send findings and results to Amazon EventBridge.

Feedback loop

Your business is constantly changing, and the design and functionality of your PCI DSS cardholder data environment and associated AWS service choices can change. Apart from business requirement changes, you should also develop your AWS infrastructure to make it more secure, efficient, and manageable. This constant change makes it imperative to establish a feedback loop in the security and segmentation controls design process. You should establish channels to gather feedback from the previous phases and from industry peers and use this feedback to make the current process better and more secure. The feedback loop and the associated changes might require reevaluation of current segmentation controls implemented to define the PCI DSS scope. This reevaluation can also stem from a change in your

organization's CHD flow. Regardless of the reasons, there must be at a minimum a yearly process of validating your established PCI DSS scope and recategorizing systems in the scope, if required.

Conclusion

This paper addressed various architectural patterns that you can adopt to design proper segmentation boundaries to help restrict the PCI DSS scope to system components necessary for secure functioning of the CDE resources hosted on AWS. The services and features used to design segmentation boundaries include AWS accounts, security groups, and abstracted AWS services. All are cloud native and thus resilient and elastic in nature. You can implement the infrastructure as code and deploy it through automation by using your organization's existing continuous integration and continuous delivery (CI/CD) pipeline to help ensure compliance.

The segmentation philosophy defined by PCI DSS, of isolation and restricted communication, does not vary for resources on AWS; the variation is in the way those controls are achieved and is unique to the AWS Cloud. In addition, as part of the shared responsibility model, using PCI DSS validated AWS services does not imply that use of those services leads to the achievement of PCI DSS compliance for your environment. You must use and architect those services in a PCI DSS complaint manner. Your organization is responsible for the scope, design, and determination, but the design becomes more agile on AWS.

Contributors

Contributors to this document include:

- Ted Tanner, Principal Assurance Consultant, PCI DSS QSA, AWS Security Assurance Services LLC
- Avik Mukherjee, Sr. Security Consultant, AWS Global Services Security
- Joseph Okonkwo, Sr. Security Architect, AWS Professional Services

Further reading

For more information, see the following resources:

- Payment Card Industry Data Security Standard (PCI DSS) 3.2.1 on AWS
- Architecting on Amazon ECS for PCI DSS Compliance
- Architecting Amazon EKS for PCI DSS Compliance

- PCI DSS and AWS Foundational Security Best Practices Controls using AWS Security Hub on the AWS Cloud
- Audit companion for the AWS PCI DSS Quick Start
- AWS Whitepapers and Guides
- SP 800-37 Rev. 2. Risk Management Framework for Information Systems and Organizations: A System Life Cycle Approach for Security and Privacy
- PCI Security Standards Council Penetration Testing Guidance
- Organizing Your AWS Environment Using Multiple Accounts
- PCI SSC Cloud Computing Guidelines
- PCI DSS Virtualization Guidelines

Document revisions

Date	Description
April 2019	First publication
May 2023	Second publication