PCI Security Standards Council®

Date: September 2024

Author: Scoping and Segmentation for Modern Network Architectures Special Interest Group

INFORMATION SUPPLEMENT

PCI DSS Scoping and Segmentation Guidance for Modern Network Architectures

Contents

1 Introduction

As organizations embrace modern network architecture advancements and leverage its benefits, ensuring the security of payment account data becomes paramount. While adopting modern network architecture provides considerable advantages, it also introduces challenges in maintaining secure environments by increasing the complexity of PCI DSS scope determination and segmentation practices. One key challenge is maintaining effective network segmentation controls.

Network segmentation is a fundamental security measure that isolates systems and networks to minimize the impact of an incident and make it easier to restrict unauthorized access. Segmentation allows the application of security controls based on the security needs of systems, which can ultimately reduce PCI DSS scope to only necessary systems and processes within and connected to an entity's cardholder data environment (CDE). However, the dynamic nature of cloud environments and evolving network topologies present unique challenges in achieving and maintaining robust network segmentation.

Traditional network configurations typically follow a perimeter-based security model, where a clearly defined network perimeter acts as the primary line of defense. However, the evolution of technology—including cloud adoption, virtualization, and containerization—has allowed organizations to scale resources rapidly, adopt microservice architectures, and leverage serverless computing, all leading to changes in network topology. Networks are no longer confined to physical boundaries, and instead can span multiple cloud service provider (CSP) regions, availability zones, virtual private clouds (VPCs), and even hybrid environments.

Security strategies have had to evolve to support this rapid technological advancement. Traditional network security controls (NSCs) may no longer seamlessly translate between environments, which is causing a fundamental shift toward security strategies centered around identity and access management (IAM). One shift has led the way to the development of zero trust architecture models, which operate on the assumption that no user or system can be inherently trusted, regardless of its location within the network. Security controls are implemented based on system component identity, device posture, and other contextual factors. Zero trust aligns well with cloud computing principles, which focus on granular access control and verification of every transaction or communication, irrespective of network boundaries.

This document provides insights into network segmentation controls, cloud environments, and modern network architectures. It addresses common challenges and provides guidance on how organizations can adapt traditional network security principles to the dynamic and distributed nature of these modern network architecture environments to support their PCI DSS compliance efforts.

Note: *Each entity must make its own PCI DSS scoping decisions, design effective segmentation (if used), and meet its own PCI DSS compliance and related validation requirements. Following this guidance guarantees neither effective segmentation nor compliance with PCI DSS.*

1.1 Intended Use and Target Audience

This guidance is intended for any entity or individual seeking to understand PCI DSS scoping and segmentation practices and their applicability to modern network architectures.

***Note:** This information supplement provides supplemental guidance that does not add, extend, replace, or supersede requirements in any PCI SSC standard or existing PCI SSC information supplements.*

The best practices in this document can apply to organizations of all sizes and complexities within the payments ecosystem that use modern network architectures, regardless of the specific technology or tools employed. While this document does not endorse any specific technologies, products, or services, it recognizes that these technologies exist and may influence the security of payment card data. As such, the following sections may include terms associated with specific technologies but the concepts described remain broadly applicable.

***Note:** At the time of publication, the current version of PCI DSS is v4.0.1; however, the general principles and practices offered here may also apply to other versions.*

The level of detail provided in this document is based on the assumption that the core target audience has foundational knowledge of:

- Scoping and segmentation principles and practices, as described in the PCI SSC *Guidance for PCI DSS Scoping and Network Segmentation* Information Supplement, located in the Document Library of the PCI SSC website.
- Zero trust, cloud computing, and network security control architectures.

As such, the target audience for this document includes, but is not limited to:

- Merchants, third-party service providers (TPSPs), issuers, and others that implement controls to meet PCI DSS requirements for their enterprises or clients.
- Assessors (such as Qualified Security Assessors or Internal Security Assessors) that confirm that PCI DSS requirements are being met for modern network architectures.
- PCI Forensic Investigators (PFIs) that determine PCI DSS scope as part of an investigation.
- Compliance-accepting entities (such as acquirers, payment brands, or other entities) responsible for determining and evaluating the PCI DSS validation effort for their clients.

The PCI Security Standards Council (PCI SSC) is not responsible for enforcing compliance or determining whether a particular implementation is compliant. Entities should work with the organization(s) that manages their compliance program, such as an acquirer (merchant bank), payment brand, or other entity to understand the entity's compliance validation and reporting responsibilities.

1.2 Terminology

The following terms and acronyms are used throughout this document:

- CDE – Cardholder Data Environment
- CHD – Cardholder Data
- SAD – Sensitive Authentication Data
- Account Data – Cardholder and/or Sensitive Authentication Data
- NSC – Network Security Controls

- CSP – Cloud Service Provider
- ZT – Zero-Trust

Definitions for payment-specific terms noted within this document can be found in the *Payment Card Industry Data Security Standard (PCI DSS) v4.x, Appendix G* Glossary.

2 Understanding Scoping and Segmentation Practices for Modern Network Architectures

As networking services and device connection structures evolve to create modern architectural environments, the ways organizations apply PCI DSS scoping, segmentation practices, and controls in these environments must also evolve. Before applying PCI DSS controls, it is important to first understand PCI DSS scoping fundamentals, described at length in *PCI DSS* section 4, *Scope of PCI DSS Requirements*. The foundational scoping statement noted below is the basis of all best practice and guidance information found in the following pages of this document.

Scope of PCI DSS Requirements [1]

PCI DSS requirements apply to:

- The cardholder data environment (CDE), which is comprised of:
  - System components[2], people, and processes that store, process, or transmit cardholder data (CHD) and/or sensitive authentication data (SAD); and
  - System components that may not store, process, or transmit CHD/SAD but have **unrestricted connectivity** to system components that store, process, or transmit CHD/SAD.

  AND

- System components, people, and processes that could impact the security of cardholder data and/or sensitive authentication data.

To establish a CDE, defined in the first bullet above, there should first be a clear understanding of business needs and processes related to the storage, processing, and transmission of account data (CHD and/or SAD). Restricting account data to as few locations as possible by eliminating unnecessary data and consolidating necessary data may require an organization to reengineer some of its long-standing business practices.

To be considered out of scope for PCI DSS, a system component must be properly segmented (isolated) from the CDE, such that the out-of-scope system component could not impact the security of account data, even if that component was compromised. Segmentation must be implemented and verified via penetration testing to confirm connections to the CDE are not possible.

Note: *For information on the fundamental PCI DSS scoping and segmentation principles, as they apply to traditional environments, review Guidance for* PCI DSS Scoping and Network Segmentation *Information Supplement.*

Different modern network architectures come with varying lists of benefits and drawbacks, which means there are many factors an organization should consider before choosing a modern network architecture for its

[1] For details, refer to PCI DSS v4.0.1 section 4, *Scope of PCI DSS Requirements*.

[2] For examples of "system components," refer to PCI DSS v4.0.1 section 4, *Scope of PCI DSS Requirements*.

environment. Any decision should include involvement from as many teams as possible to account for potential impacts, such as networking, information security, sales, infrastructure, and legal, to name a few. Common examples of modern network architectures that can present unique data security challenges include multi-cloud environments, hybrid on-premises and cloud environments, zero-trust networks, and network virtualization technologies.

A Multi-Cloud environment integrates multiple cloud service providers (CSPs) into a single network architecture to provide various benefits such as redundancy, vendor variation, cost optimization, and risk diversification. The main challenges a multi-cloud environment presents for PCI DSS scoping and segmentation are that the complex topology can lead to limited visibility into activity, traffic, data transfer, and resource discoverability, and managing complex and varying product functionality offerings across multiple vendors can be time-consuming and complicated. A correctly implemented segmentation strategy that defines PCI DSS scope boundaries should be comprehensive with clear policies and centralized management and monitoring, using each CSP's tools and automation. Collaboration between security and networking teams is vital.

Zero-Trust Network Architecture is an approach to, and framework for, network security that moves away from traditional perimeter-based security, which includes firewalls, switches, and routers to micro-segmentation. Instead of having broad perimeter access rules, the network is divided into smaller segments with network security controls (NSCs), leveraging least privilege access and continuous authentication. The challenge in moving to a zero-trust environment is that it may require new skill sets, and is usually difficult and time-consuming to adopt, implement, and document.

A hybrid environment of on-premises and cloud-based architecture combines a traditional on-premises data center with cloud-based services. When correctly implemented, hybrid environments offer many benefits, such as the ability to keep sensitive data on premises and can provide legacy system support while providing the scalability and flexibility of the cloud. However, determining scope, creating data flow diagrams, and applying consistent segmentation controls across multiple environments can be challenging.

Network virtualization technologies, such as software-defined networks (SDNs) and service meshes, abstract and manage the underlying physical network infrastructure to support the network flexibility, scalability, and overall efficiency. SDN focuses on centralized control and management of network resources, while service meshes focus on data transmission between microservices. The dynamic and complex nature of network virtualization technology topologies makes creating an effective segmentation policy and consistent enforcement challenging.

Regardless of the modern network architecture(s) an organization deploys, certain factors should remain consistent to ensure effective PCI DSS scoping and segmentation practices. These include the consistent application of documented policies, strong identity and access management practices, and most importantly, a solid understanding of where the data is stored and how it flows. This understanding is fundamental to PCI DSS scoping and segmentation practices and should be a priority in any network architecture decision.

The following sections delve into each of these modern network architecture types and explore the risks and best practices associated with each.

3 Multi-Cloud Environments

When an entity's cloud infrastructure[3] relies on two or more distinct cloud service providers (CSPs), this is known as a multi-cloud environment. In multi-cloud environments, an entity's cloud infrastructure commonly includes, but is not limited to, interactions between two or more distinct CSPs specifically where Infrastructure as a Service (IaaS) or Platform as a Service (PaaS) services are used.

Depending on an entity's growth strategy and how operations are managed, multi-cloud environments can be implemented by design or as a byproduct of their evolution. These environments might develop through mergers, acquisitions, or changes in leadership. Alternatively, these environments may arise to leverage the unique strengths or offerings of different CSPs to meet specific business drivers, provide enhanced support for desktop applications, or accommodate legacy systems.

An often-overlooked aspect of multi-cloud environments is that different CSPs within an environment may use different terms to represent similar service offerings, which can add a layer of complexity to managing the overall environment. A simple example is the ubiquitous Domain Name System (DNS). Each CSP may name its offered DNS resolution service differently, and functionality may differ between cloud platforms. Organizations should review the nuances of the functionality for each CSP they use and document those details as they relate to PCI DSS requirements—for example, Requirement 1 for network security controls.

Multi-cloud environments introduce additional complexity beyond traditional cloud implementations due to variations in boundaries, responsibilities, service offerings, and their interactions. While the increased attack surface alone may not directly add complexity, the combination of increased complexity and the potential for an expanded attack surface can elevate risk. Therefore, entities should carefully consider these risks as part of any multi-cloud deployment strategy.

PCI DSS-specific considerations for managing scope boundaries and segmentation practices in multi-cloud environments can be found below in "Guidance for Specific PCI DSS Scoping and Segmentation Requirements."

In cloud environments, organizations should pay special attention to how their resources are deployed and how they connect to each other. The determination of "connected-to" and/or security-impacting system components must also consider software-defined environments and the use of on-demand software-as-a-service (SaaS) products. CSPs offer serverless event-driven computing services that allow organizations to run code on-demand and without provisioning persistent resources. When the compute service or function is called, the CSP temporarily creates a small computing environment complete with connectivity as defined by the parameters an organization sets when the resource is deployed. If the defined parameters include the ability to connect to a CDE resource, this pre-compiled code resource would be considered "connected to" and in-scope. When the conditions are met to run the defined compute service or function, a network connection is briefly created as defined by the code and a connection is established to the CDE resource. This potential connectivity presents itself differently than traditional "connectivity" found in on-premises environments but would still be considered "connected-to."

[3] Refer to *PCI SSC Guidance for Cloud Computing* Information Supplement for definitions

3.1 Multi-Cloud Environments and Segmentation

Once the scope of each provider, service, application, and role is identified, and its implication to the scope of an organization's PCI DSS assessment is known, it may be possible to limit the scope of a PCI DSS assessment by implementing segmentation.

3.1.1 Ways to segment Cloud Service Providers (CSPs) in a Multi-Cloud Environment

Organizations that implement a multi-cloud design should first consider how to interconnect these environments before implementing segmentation controls. To achieve proper segmentation between PCI DSS in-scope and out-of-scope systems across various cloud implementations, each entity should implement an overall connectivity strategy and then configure segmentation controls to refine the scope of their CDE. Some of the options customers have for interconnectivity include

- Site-to-Site VPNs – This connectivity option can be achieved by using CSP-managed services and customer or provider managed infrastructure within the CSP computer platforms.
- Dedicated private links – Customers and CSP can work together to implement dedicated links between environments to establish a private direct connection between multiple cloud environments.
- Exposing IPs over the public internet – Deployed or managed systems components can be given access to the public internet via a NAT configuration or CSP configuration using access control lists (ACLs) to control the traffic allowed or denied.

For additional information on interconnectivity options, consult with a CSP or visit a CSP's documentation pages for supported options and features available to customers.

Within a multi-cloud implementation, the organization will need to configure settings for each CSP to achieve proper segmentation between in-scope and out-of-scope components, which can be achieved with network-layer segmentation or host- or application-layer segmentation, as described below:

Network-Layer Segmentation

When connectivity is established between CSPs over site-to-site VPNs or dedicated private links, the customer has created a private connection between their CSPs where a private IP space can be used. This does not alone achieve sufficient segmentation between PCI DSS in-scope and out-of-scope systems. Since large subnets are typically configured for interconnectivity, these subnets may contain systems to which PCI DSS controls will not be applied, therefore additional segmentation logic should be implemented to remove these systems from the scope of the assessment. Customers can use cloud managed services, or vendor open-source or custom solutions to implement network security controls (NSC). They may configure firewall rules, network ACLs, network policies, or other network security technology to control network traffic to achieve segmentation between systems at the network or application layer, as applicable. The goal of this segmentation is to ensure that out-of-scope systems do not have access to in-scope systems or account data and cannot impact the security of the CDE.

For system components which are exposed over the public internet, the customer is responsible for implementing proper access controls on the ingress and egress side between their CSPs. When using network layer segmentation, the customer should implement NSCs at the edge of the network containing system components processing or transmitting account data, or systems that could impact the security of account data. Traffic may be controlled by using a proxy to expose internal systems to the public internet and only allowing traffic to the alternate CSP system component that requires access. While this communication may occur between two or more CSPs, the communication would traverse the public internet, which is considered untrusted and therefore subject to PCI DSS Requirement 4, as well as Requirement 1 ensuring least privilege is configured between the publicly exposed PCI DSS in-scope systems.

Host- or Application-Layer Segmentation

In a multi-cloud environment, regardless of the connectivity type, there is an advantage to using host and application layer segmentation because it abstracts specifics about the CSP network solutions to the host or application level. Host-based firewall solutions can provide a level of segmentation to the specific system running an in-scope system component without having to remove an entire network segment from scope. This provides the ability to have a simpler network architecture where segmentation controls are close to the specific host that is storing, processing, or transmitting account data. Segmentation controls can be applied at the application layer, which may contain APIs, microservices, containers, and databases where the underlying infrastructure may vary between deployments in the various CSPs. This level of segmentation is usually applied through software-defined networking, service mesh, or other products that provide a policy engine or control plane allowing the application administrator to configure which applications are allowed to communicate with each other. An additional approval layer can be built on top of this type of technology to assure that changes are reviewed and validated prior to allowing additional communication between systems in production environments.

3.1.2 Ongoing Segmentation Management

Understanding the implications of introducing new workloads or changing existing workloads in a multi-cloud environment is crucial. Even simple modifications to a defined infrastructure-as-code (IaC) implementation can inadvertently remove segmentation controls. To mitigate this risk, it is vital to prioritize change management, security testing of IaC, and change detection in both single- and multi-cloud environments.

To effectively assess the impact of changes on a multi-cloud environment, it is advisable to implement change management processes that automatically identify alterations to segmentation controls, such as Identity and Access Management or networking configurations. This enables thorough review and evaluation of how the change may affect a multi-cloud environment. Organizations should implement preventive and detective measures, such as cloud policies and configuration checks, that reinforce segmentation controls by thwarting destructive changes. The capabilities of these measures may vary across CSPs and services. Consequently, organizations should ensure they have implemented complementary controls across their multi-cloud environments.

Upon completing any changes that may affect segmentation controls, it is essential for organizations to verify and document that the segmentation controls remain intact. They should also confirm that the introduced changes do not cause additional segmentation controls or result in increased scope for the PCI DSS assessment, planning, and execution. The aim is to validate the effectiveness of restricted network controls and ensure that customer systems and shared infrastructure are appropriately segmented. This means that direct and indirect changes do not cause additional segmentation controls or lead to in-scope creep.

3.2 Penetration Testing of Segmentation Controls

Penetration testing to demonstrate effectiveness of segmentation controls (segmentation pen testing) should include tests between CSP management nodes and customer systems, and between customers on shared infrastructure to verify the isolation efficacy. It should consider architectural nuances in multi-cloud environments, such as serverless components, clusters of containers, sidecars or similar proxy devices, as well as subnets, routing tables, and security groups.

Segmentation pen testing should also consider connections into the in-scope environment from build and deployment systems or other development pipelines and should also expressly consider how the CSP console or other management of raw cloud resources affect the security of the environment. With these parameters, the identification of the intended segmentation boundaries would become clearer.

Particular care should be taken to avoid testing components owned by other parties for which the tester does not have authorization. Segmentation pen testing should be performed with a clear understanding of the boundaries between the target entity and other entities, such as the CSP(s) involved, or other third parties connecting to or providing services to the tested entity.

When performing segmentation pen testing in a multi cloud environment, the following considerations should be made for each of the potential situations described below:

Type of Testing	Considerations
Within and across different virtual private clouds (VPCs)	Testing for segmentation involves more than just validating the existence of layer 3 access controls lists (ACLs) using port scanning techniques between in-scope and out-of-scope VPCs. Testing should consider whether and how the VPCs connect to one another and whether those connection points can be exploited. Further, testing should consider how the organization changes its cloud resources, including VPC configurations, and evaluate whether any testing scenarios should simulate attacking the console or similar control plane to change the VPC configuration to defeat the segmentation controls.

Type of Testing	Considerations
Inter-account connectivity using more than one account in the same CSP	Many organizations make use of more than one cloud resource account within a given CSP (for example, a CSP customer may have many projects, each running in its own tenancy). In many cases, these tenancies are subsidiary to an organization-wide master account and may deploy resources into the same or adjacent VPCs. These separate accounts may have valid reasons to connect or interact, and a segmentation test should consider the scope boundaries between these accounts and whether the connectivity between the two could be exploited. Like other segmentation testing, the testing should contemplate an attack against the control plane (for example, master account console) to change the configuration to gain access, or an attack where an attacker stands up a new account with new cloud resources and attempts to interact with legitimate cloud resources.
Across more than one CSP	Organizations that make use of multi-cloud environments will have connectivity established between components in these different CSPs. These may involve WAN connections, remote access, public network connections, or linkages using combined identity and access management (IAM). Penetration testing should consider scenarios where segmentation is implemented to provide isolation between components in different CSPs and where segmentation provides connectivity through specific defined mechanisms, to determine whether the pen testing can defeat or allow new, unintended connections to be established. Further, testing in environments using joint sets of user accounts or other entitlements should consider whether attacks against the entitlement-granting process allow cross-CSP access in unintended or harmful ways.

4 Zero-Trust Architecture

Zero-trust (ZT) architectures are designed to continuously revalidate the authenticity and authorization of persons, devices, and services to specific resources based upon the classifications of each. While assuming a “deny-by-default” approach to security, implementing ZT architecture can apply detailed, restrictive segmentation that goes beyond classic networking security controls.

In classical architectures, an environment boundary is defined by the network segments containing devices in which data is stored or processed, or through which it is transmitted. All devices in an in-scope segment are considered being within that boundary as well. In contrast, ZT architectures can rely on the individual devices themselves, bound by software-defined inspection points that determine what data is allowed to pass between components.

Both traditional segmentation methods and ZT techniques must be shown to secure access to the CDE from out-of-scope environments, in line with PCI DSS Requirement 1.

ZT architectures are those designed to protect hosted resources with granular, real-time policy enforcement points. These policy mechanisms determine who or what can query a resource based on a wide range of factors. Only when these granular policies are met may data be passed between subjects and objects. Boundaries may even isolate individual processes within a resource.

Detailed processes to build and maintain a ZT environment can be found in other publications. NIST Special Publication 800-207 and the CISA Zero-Trust Maturity Model (ZTMM) are valuable references for additional guidance.

4.1 Zero-Trust Architecture and Segmentation

When implementing PCI DSS segmentation for traditional network architectures, environment boundaries are often defined, in part, by the network segments containing system components that are in scope for PCI DSS requirements, and supported by the implementation of devices, such as firewalls or routers at the network segment’s perimeter. If a network segment is in scope for PCI DSS requirements, all devices in that segment are therefore in scope as well. The environment scope under ZT architectures, in contrast, consists of these individual devices themselves, bound by software-defined inspection points that determine what data is allowed to pass between components. In a ZT environment, these boundaries are as granular as the data validation and inspection points—perhaps even isolating individual processes within a device.

ZT architectures, focus on the devices themselves, managing device access via software-defined inspection points that determine what each device is allowed to do and what data and resources each device can access.

ZT can both rely on and enhance the application of PCI DSS requirements controls in an environment. As an organization’s ZT maturity advances, its ZT architectures can employ PCI DSS controls to supplement its policies, processes, and security strategy to help determine–in real time–whether to permit access by users, devices, and services to resources in the CDE and connected-to/security-impacting environments.

ZT does not replace traditional network security controls at the perimeter but understands and accounts for their limitations.

4.1.1 Documenting and Validating Zero-Trust Implementations

The assessed entity should document how ZT is implemented and maintained. The policies and procedures used to implement ZT are akin to the processes used to implement any access control. At a minimum, policies and procedures should define how these controls are established, maintained, and monitored. The following are examples of content:

- Identify Protected Data: Clearly identify and document all areas where sensitive data, including payment account data (cardholder data and/or sensitive authentication data), authentication values, encryption keys, etc.) is stored, processed, or transmitted. This includes databases, servers, applications, and network segments that handle CHD and/or SAD.
- Implement Zero-Trust Principles:
  - Verify and Authenticate: Explain how authentication mechanisms are enforced for all users, devices, and applications accessing sensitive data.
  - Limited Access: Describe how the principle of least privilege is applied, ensuring that users and systems have only the minimum access necessary to perform their roles.
  - Micro-Segmentation: Detail how network segmentation is employed to isolate sensitive data from other parts of the network. This should include network diagrams and policies.
  - Monitoring: Document how these controls are monitored. Examples include real-time monitoring of network traffic, user behavior, and access patterns.
  - Encryption: Describe how data is rendered unreadable in storage and in transit.
  - Logging and Auditing: Define logging configurations and centralization to support how the organization will identify and respond to suspicious behavior.
- Network Diagrams and Data Flows: Create diagrams that illustrate the flow of sensitive data within the environment. Highlight how ZT principles apply to these data flows.
- Documentation of Policies and Procedures: Update documentation to include roles and responsibilities, incident response plans, and security training programs.
- Risk Assessments and Ongoing Reviews: Document how the organization conducts regular audits and assessments to ensure that ZT principles and PCI DSS requirements are met continually. Include evidence of audit results and any remediation actions taken.
- Third-party Service Provider Considerations: If third-party vendors are used for any part of a ZT infrastructure or PCI DSS compliance there should be documentation on assessment and management of their security controls to ensure they align with ZT and PCI DSS requirements.
- Incident Plan: Describe how current incident response plan integrates with ZT architecture to quickly detect and respond to security incidents that may impact sensitive data.

The first three bullets above are evaluated at the outset of an assessment to confirm the scope of the assessment. The assessor should correlate documented actions with the controls in place—defining trust boundaries, the CDE, connected-to/security impacting systems/resources, and out-of-scope systems/resources. Given the conditions set in ZT, the assessor needs to determine how the continuous evaluation of identities—i.e., human user, service accounts—and data inventorying apply restrictive, preventative controls.

On-premises vs. Cloud Environment Zero-Trust Implementations

The principles for any ZT implementation do not change between on-premises networks and cloud networks. It is important to confirm that policy enforcement points are placed at the access points of each application/service and data source so that access to policy enforcement points can be directly tracked, monitored, and validated.

Where a hybrid or multi-cloud environment is in use, the evaluation by the broker between asset classification and resource (human, device, application, other) attempting to access that asset should be defined and substantiated. Where controls fail, a clear path to maintain segmentation but also provide business continuity should be understood and documented. It may be necessary to map the asset classifications and access categories when establishing persistent communications channels between organizations implementing different ZT frameworks.

It's important to note that considerations for implementing these controls may require unique integrations with cloud service providers.

4.2 Penetration Testing of Segmentation Controls

Penetration testing in both Zero Trust (ZT) and classical environments shares the common goal of identifying vulnerabilities and validating the effectiveness of defenses. However, the ZT model's emphasis on internal, per-transaction verification rather than traditional perimeter defense significantly alters how penetration tests are planned and executed. This shift necessitates a more nuanced approach, considering the unique characteristics of a ZT environment.

Key Considerations for Penetration Testing in a Zero Trust Environment:

1. Approach Based on ZT Architecture:

The penetration testing strategy must be tailored to the specific ZT architecture model implemented. This involves confirming the effectiveness of segmentation efforts by focusing on the validation of micro-segmentation controls and trust boundaries. Unlike traditional environments, where broad perimeter defenses are the primary focus, ZT requires granular testing of internal interactions and the enforcement of trust zones.

2. Defense Verification:

In a ZT environment, the emphasis shifts from perimeter testing to the robustness of internal defenses. Penetration testers must focus on verifying that internal security policies are effective in detecting and mitigating threats originating from both external and internal sources. This involves rigorous testing of access controls, authentication mechanisms, and the ability to contain threats within segmented zones.

3. Lateral Movement Testing:

In traditional networks, once inside, entities are often implicitly trusted, allowing for lateral movement. However, ZT assumes no inherent trust. Testers must validate that each action or step within the network requires proper authentication and is subject to strict validation. Testing should ensure that micro-segmentation effectively limits or prevents lateral movement, reducing the risk of an attacker spreading within the network.

4. Target Selection:

The selection of targets for penetration testing in a ZT environment is driven by the granularity of trust boundaries. Testers must identify and focus on individual environment elements that grant access based on defined characteristics, such as data sensitivity or user roles. Devices or systems within finer-grained trust structures require individualized testing, excluding those that, based solely on network addresses, might have been included in traditional tests.

5. Penetration-Testing Payloads:

In a ZT environment, the payloads used to test segmentation controls may need to be specially crafted to challenge the assertions implemented through data tagging schemes. The payloads should be designed to test the effectiveness of policies that govern data access and movement within the segmented network.

6. Policy Testing:

ZT heavily relies on well-defined and enforced policies for network access. Penetration testers must rigorously test these policies to ensure they are comprehensive and free from exploitable loopholes. This involves testing policy enforcement across various segments of the network, ensuring that access controls are both effective and aligned with the organization’s security posture.

7. Behavioral Analytics Testing:

ZT networks often incorporate User/Entity Behavior Analytics (UEBA) as part of their security framework. Penetration testers should assess the effectiveness of these systems in detecting and responding to anomalous behavior. This includes testing the UEBA’s ability to identify deviations from established baselines and its integration with other security mechanisms in the ZT environment.

Additional Considerations:

- Continuous Validation:

In a ZT model, where the network environment is dynamic and policies may frequently change, continuous validation through regular penetration testing is essential. Testers should establish a schedule for ongoing assessments to ensure that segmentation controls remain effective over time.

- Collaboration with Security Operations:

Penetration testers should work closely with the Security Operations Center (SOC) to understand the monitoring and incident response processes in place. Testing should include scenarios that simulate real-world attacks to validate the effectiveness of detection and response mechanisms.

- Documentation and Reporting:

Comprehensive documentation of the testing process, findings, and recommendations is crucial. This includes detailed reports on any vulnerabilities identified, their potential impact, and suggested remediation steps. In a ZT environment, documenting the effectiveness of segmentation controls is particularly important for ongoing security management and audits.

4.3 Risks with Zero-Trust Architecture

When managing ZT in hybrid cloud and multi-cloud environments, aspects of each environment should be uniquely considered. The nuances between environments and supported technologies should be documented to support ongoing use and align with incident response actions to protect the confidentiality of sensitive data.

The complete implementation of ZT architecture in a new environment can be lengthy and include a transition period when the implementation may be partial or incomplete. During this transition period, there may be an increased risk of an attacker gaining unauthorized system component access. Given this increased likelihood, incomplete implementations may require additional controls to create and maintain PCI DSS scope boundaries until the transition is complete

Ultimately, due to the inherent complexity of ZT, consistent policy application and asset categorization are critical. Ideally, organizations should use ZT implementation management tools and avoid manual processes to monitor ZT technical controls.

In addition, given that ZT architecture requires active maintenance for continued effectiveness, an absence of security control upkeep can increase an attack surface in the same way that zombie systems do in virtual environments. Zombie systems, which are virtual machines that are no longer in active use but still consume system resources, add cumbersome layers of review, maintenance, and evidence collection during a PCI DSS assessment.

5 Hybrid Cardholder Data Environments

Hybrid cardholder data environments are defined as those that incorporate both on-premises and cloud-based components, presenting unique challenges and opportunities for PCI DSS scoping and segmentation. To effectively manage PCI DSS scope, it is essential to apply a strategic segmentation approach that aligns with the key principles outlined below:

- **Identify Critical Segmentation Boundaries:** Clearly define and identify the critical segmentation boundaries between on-premises and cloud environments. Understand the flow of account data and ensure it traverses secure channels.
- **Apply Consistent Controls:** Maintain consistency in security controls across both on-premises and cloud components. This includes access controls, encryption standards, and monitoring mechanisms.
- **Design Dynamic Scalability:** Segmentation strategy should accommodate dynamic scalability, allowing for the seamless integration of new cloud resources without compromising security.

Establishing robust controls is paramount to effective segmentation in these environments, the following controls are especially critical:

- **Network-Based Controls:** Leverage firewalls, VLANs, and intrusion detection/prevention systems to enforce network-based segmentation controls. Ensure these controls are applied consistently across on-premises and cloud networks.
- **Identity and Access Management (IAM):** Implement IAM solutions that unify access controls across on-premises and cloud platforms. This includes robust authentication mechanisms and the principle of least privilege.
- **Data Encryption:** Apply strong encryption protocols for data in transit and at rest, maintaining a consistent encryption strategy across all environments to protect account data.

Clearly define roles and responsibilities for managing segmentation in a Hybrid CDE. Assign ownership for each aspect of the segmentation strategy, covering both on-premises and cloud components. This ensures accountability and facilitates effective communication between on-premises and cloud teams. Refer to the example Responsibility Matrix in Appendix B of this document for additional guidance.

5.1 Hybrid Cardholder Data Environments and Segmentation

Segmentation testing is a critical process to validate the effectiveness of access controls and ensure that the cardholder data environment (CDE) remains isolated and secure. In a hybrid solution that integrates various components, the following guidance provides a comprehensive approach to conduct thorough segmentation testing.

Scope Definition	Clearly define the scope of the segmentation testing efforts, encompassing all components within the hybrid environment that play a role in handling account data. This includes on-premises servers, cloud instances, networking devices, and any intermediary components.

Test Scenarios	Craft diverse test scenarios that simulate real-world conditions and potential attack vectors. Consider scenarios where unauthorized access attempts are made from both internal and external sources. This should include testing interactions between on-premises and off-premises components, ensuring that data transfers occur securely, and access controls are consistently enforced. Where WAN connections join these environments, consider the use cases for the organization to traverse from on-premises to cloud components or vice versa, and if the WAN connectivity is used for any sort of access control. Also consider attacks such as spoofing a client from an on-premises environment attempting to access a cloud resource or spoofing a cloud resource's response to an on-premises request to subvert the segmentation controls. Where a remote access gateway joins these environments, as in conventional, on-premises environments, consider how an attacker could circumvent a remote access gateway, such as attacking the gateway system(s) directly, attacking the user provisioning system—e.g., Active Directory or IAM—or impersonating privileged users to defeat segmentation controls.
Dependency Mapping	Understand the dependencies between different components within the hybrid environment. Identify critical interconnections and interactions between software-defined networking, service meshes, and ZT architecture. This mapping is essential to ensure that segmentation controls are not inadvertently bypassed.
Penetration Testing	Conduct realistic penetration-testing exercises that mimic sophisticated attacks. This includes scenarios where attackers attempt to exploit potential misconfigurations or weaknesses in the hybrid environment to gain unauthorized access to the CDE.
Compliance Validation	Ensure that segmentation testing aligns with PCI DSS requirements. Validate that access controls and segmentation practices comply with industry standards and regulations.
Network Diagram and Dataflows	Ensure that network diagrams and account dataflow diagrams depict the different environments—e.g., on-premises, cloud hosted, etc.
Documentation and Reporting	Document the segmentation testing process, including test scenarios, methodologies, findings, and remediation actions. Provide clear and concise reports to stakeholders, highlighting areas of strength and improvement in the segmentation strategy.

5.2 Risks with Hybrid Cardholder Data Environments

It's important to acknowledge and address the inherent risks associated with Hybrid CDEs, including:

- **Data Transfer Risks:** Secure data transfer between on-premises and cloud environments to mitigate the risk of unauthorized access during transit.

- **Integration Challenges:** Anticipate and address challenges in integrating security controls across diverse environments, balancing the need for consistency with the unique characteristics of each platform.
- **Compliance Consistency:** Ensure that security measures align with PCI DSS requirements consistently across both on-premises and cloud deployments.

In summary, a well-defined and consistently applied segmentation strategy is fundamental in Hybrid CDEs, fostering a secure architecture for handling account data across on-premises and cloud platforms.

6 Network Virtualization Technologies

Network virtualization technologies are a set of methods and tools used to abstract and manage network infrastructure to establish a foundation for the flexible, scalable, resilient, and efficient management of network environments. Software-defined networks and service meshes are two such technologies. They are often deployed in large-scale environments and can play a pivotal role in helping organizations achieve segmentation between PCI DSS in-scope and out-of-scope services deployed across diverse compute types, including traditional VMs, containers, and physical servers.

Software-Defined Networking (SDN)

Software-defined networking (SDN) architecture aims to centralize and simplify network management through the separation of the control plane and data plane, which is the fundamental idea behind SDN, allowing network administrators to control and manage the network infrastructure centrally and programmatically.

Traditionally, in conventional network architectures, the control plane and data plane are tightly integrated within networking devices like switches and routers. The control plane decides how data should be forwarded, routing protocols, and network management, while the data plane forwards the data packets to their destinations.

In an SDN architecture, the control plane is abstracted from underlying network devices and moved to a centralized software-based SDN controller. This controller communicates with the network devices through APIs, enabling programmable configuration and management of the network. The SDN controller makes near real-time decisions based on network conditions and defined policies, dynamically configuring the network devices to forward data packets according to specific rules.

The data plane consists of network devices, such as switches and routers, that handle the actual forwarding of data packets. These devices receive instructions from the SDN controller and implement the forwarding behavior accordingly. By separating the control plane from the data plane, SDN provides network administrators with a holistic view of the entire network and enables centralized orchestration, leading to increased agility, automation, and security. SDN architecture also supports network virtualization, allowing the creation of multiple isolated virtual networks, enhancing scalability, and resource utilization in complex network environments.

Security Benefits

Software-defined networking (SDN) offers a range of security benefits that enhance network protection and response capabilities. For example, the support for network virtualization in SDN enables micro-segmentation and network isolation, limiting the lateral movement of threats and reducing the attack surface. Fine-grained access control ensures that only authorized communication occurs between services and devices. SDN's automation capabilities enable rapid threat mitigation by facilitating quick updates to security policies and the isolation of compromised devices.

SDN can also enhance security by managing and enforcing end-to-end data encryption via the control plane, adding an extra layer of protection against unauthorized access and eavesdropping to secure communication.

Service Mesh

As organizations embraced the implementation of microservice-based architectures hosted across various cloud service providers (CSPs) and on-premises infrastructure, they encountered challenges in managing traditional network architectures and segmentation. The result was overly permissive access control lists (ACLs) and an expansion of their PCI DSS scope, causing concerns for security and operational efficiency.

The service mesh approach was designed to tackle these issues. It not only implements application-level segmentation but also offers a variety of network features and security controls. These include mutual transport layer security (mTLS) encryption, authentication, authorization between services, dynamic traffic routing, service discovery, load balancing, health checking, and more. In doing this, it provides organizations with the means to reduce the cumbersome task of manually configuring IP/Port-based ACLs in complex network environments.

The service mesh architecture comprises two key components: the data plane and the control plane. This pair of components integrates with microservice architectures running on containers, virtual machines, and even physical servers.

The data plane consists of proxies sitting alongside each service within the microservice architecture. These proxies act as intermediaries, managing ingress and egress traffic for the services they protect. The actual services remain hidden from external access, while the proxies route traffic between them, ensuring secure communication.

Proxies offer features, such as mTLS functionality, which ensure secure communication by both validating and mutually authenticating digital certificates and encrypting the traffic, which adds an extra layer of protection to payment data during transmission. The proxies can be equipped with traffic monitoring capabilities, which can detect any anomalous behavior or inactivity. This enables timely alerts to personnel or automatic access removal between services, enhancing security.

The control plane serves as the orchestrator, managing the fleet of proxies. It offers administrators the flexibility to create well-defined policies and can configure access control and routing rules for the proxies in the data plane. Whether through a command-line interface, graphical user interface, or APIs, the control plane is used to manage the service mesh.

Using service meshes to support data transmission between microservices underscores the importance of ensuring consistent system configuration across system components in use. Configuration standards ensure the broad application of security features across system components, as required per PCI DSS Requirement 2.2.1.

Security Benefits

Beyond managing scope and segmenting PCI DSS in-scope services at the application level, the service mesh adds a suite of security benefits that are beneficial for meeting PCI DSS requirements and reinforcing the overall security posture of a service-based architecture. mTLS communication provides encryption and mutual authentication, establishing trust between services and ensuring secure communication channels. Without a service mesh, encryption and service to service authentication/authorization is typically handled individually within each application, increasing the likelihood of introducing vulnerabilities and misconfiguration. A centralized control point of policy enforcement reduces the overall audit and management complexity.

Another aspect is the real-time auditing and traceability features, which simplify access reviews between services. By monitoring traffic within the service mesh, administrators can easily identify unused or unnecessary access and take action, whether manually or automatically. The service mesh's proxies can inspect packets as they traverse the network, monitoring for potential intrusion attempts or suspicious traffic patterns. This approach helps safeguard services from potential threats to payment account data.

6.1 Network Virtualization Technologies and Segmentation

6.1.1 Software-Defined Networking

Network segmentation is an important component in software-defined networking (SDN) as it enables granular and dynamic control of network policies. This segmentation enhances network security by minimizing the attack surface, containing potential breaches within specific segments, and enforcing fine-grained access controls. SDN's centralized controller simplifies management and configuration, allowing administrators to efficiently deploy changes and troubleshoot issues from a single point of control.

Network segmentation in SDN optimizes resource allocation, prioritizing critical applications and ensuring their performance and availability, even during network congestion. However, organizations should strike a balance in segmentation planning to avoid over-complexity or under-segmentation.

With SDN, rather than applying traditional perimeter security using firewalls, applications run within their own segments behind a virtualized network environment layered over the physical network. This level of network segmentation enables the creation of micro segments between payment processing system components and non-payment systems. This reduces the PCI DSS scope, and enhances security through sensitive data isolation, simplified compliance, and minimized potential impact of security breaches on the network.

For more information about software-defined networking and PCI DSS, refer to the *PCI DSS Cloud Computing Guidelines* Information Supplement, which can be found in the Document Library of the PCI SSC website.

6.1.2 Service Mesh

In a service mesh, the need to segment systems at a network level using VLANs and specific trusted versus untrusted subnets is no longer necessary. The service mesh serves as the central point for traffic authentication and authorization, achieved through implementing authorization policies at the application layer. Configuring these policies can be achieved via Infrastructure as Code, incorporating a review and approval process to handle changes to the authorization rules between services. Supplementary approval systems can be implemented to automatically revoke access to services during periods of inactivity, and regular explicit approvals can be gathered for services reliant on each other.

For instance, consider the scenario of service identities handling payment data. These can be tagged or labeled as in scope for PCI DSS. The authorization policies would then dictate that only systems with this identifier are permitted to communicate with each other, while strictly denying all other traffic. This design, combining authorization policies and unique identifiers, ensures effective segmentation, helping meet PCI DSS requirements.

The service mesh brings about a shift in network segmentation, providing organizations with tools to improve their security posture and implement a more efficient approach to traffic authentication and authorization at the application level through logical access controls.

6.2 Risks with Network Virtualization Technologies

6.2.1 Software-Defined Networking

Software-defined networking (SDN) presents several risks that organizations should carefully manage. One significant concern is the vulnerability of the centralized control plane, where unauthorized access could lead to network disruptions, and data breaches, potentially causing traffic manipulation or denial-of-service attacks. The complexity of SDN and the potential for misconfigurations may cause unintended consequences, security vulnerabilities, or performance issues. Since SDNs can be configured on top of traditional network protocols, the number of customizations or special rules can become quite large, making the network layer more difficult to manage and can increase the risk of security incidents. Ensuring scalability of the control plane becomes crucial as the network grows. While SDNs can be deployed securely, their complexity and relative industry newness may lead to an increase in security threat exposure if the implementation is not correctly configured and managed.

Organizations should also address any reliance on open-source components, which can introduce risks like unpatched vulnerabilities or limited support. Properly securing east-west and north-south traffic is essential for preventing sensitive data exposure and unauthorized access. To mitigate these risks, proactive implementation of access controls, authentication, encryption, and regular security assessments is necessary to ensure the reliability and security of SDN operations.

6.2.2 Service Mesh

Service-mesh technology, while offering many benefits, also presents certain risks that organizations should recognize. One concern is the potential for a single point of failure. If the service-mesh control plane encounters issues or becomes unavailable, it could disrupt communication between services, impacting overall system functionality. Additionally, misconfigurations within the service mesh can lead to unintended consequences, such as traffic routing errors, resulting in degraded performance, service outages, or overly permissive authorization policies. Given the highly distributed and complex nature of service mesh environments, effectively monitoring and debugging issues can be a challenging task.

The increased east-west traffic within the service mesh can expose sensitive data or create security vulnerabilities if encryption and access controls are not appropriately configured. To mitigate these risks, organizations should dedicate resources to thorough testing, continuous monitoring, and robust security measures to ensure the reliability and security of their service-mesh implementation.

Since the service mesh often operates within an organization-managed infrastructure, the security of the proxies and control plane servers is critical. This makes areas such as running vulnerability scans, keeping system images up to date, hardening of the system components, proper key management for the mTLS architecture, conducting penetration tests, and actively monitoring the platform for potential intrusions, that much more important.

It is essential to note that many popular service-mesh offerings do not have all security features enabled by default. As a result, administrators should proactively enable and configure these features to fully leverage the provided security benefits. With various service-mesh offerings available, assessors should thoroughly familiarize themselves with the specific configurations in advance, allowing them to validate that security features are properly enabled and appropriately set up during the assessment process.

7 Software Deployment

Modern network architectures often make use of the advanced virtualization capabilities available, such as software-defined networking mentioned in Sections 2, 5, and 6. This introduces the ability to define, provision, and support the organization's network infrastructure using code instead of manual processes and settings. This is commonly referred to as infrastructure as code (IaC). When these newer technologies are used, an organization's software development lifecycle (SDLC), the methodology and processes it uses to design and build its high-quality software, plays an important role in the organization's PCI DSS scope determination.

7.1 PCI DSS Scope and DevOps

In modern architectures, software is often developed using the *DevOps* model, with a goal to shorten delivery times for new software updates while maintaining quality and security. DevOps especially emphasizes the use of automation to streamline the deployment process.

- ***Continuous integration*** **(CI)** is a DevOps software-development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run. Newly integrated code can be tested through an automated-build process to support early bug detection and removal.
- ***Continuous delivery*** **(CD)** is a complementary software development practice where code changes are automatically prepared for a release to production. A pillar of modern application development, continuous delivery expands upon continuous integration by deploying code changes to a testing environment and/or a production environment after the build stage. When properly implemented, developers will always have a deployment-ready build artifact that has passed through a standardized test process. Developers who use CD also depend on real-time monitoring to help detect performance, operational, and security issues once the code has been deployed to the live environment.
- ***A CI/CD pipeline*** is the automated process used to streamline the creation, testing, and deployment of applications and resources. This pipeline includes the people, processes, and technology supporting the development and deployment of resources to a CDE.

For organizations using a DevOps model for their software development and/or CI/CD pipelines for the build and deployment of resources into their CDE, their PCI DSS scope extends to these elements. Each stage of an organization's software development lifecycle (SDLC) should be reviewed and assessed as part of its adherence with the software-development and change-management requirements in PCI DSS Requirement 6. This includes the service account(s) used by an organization's pipeline, that have elevated privileges to a CDE to manage deployments and can impact the security accordingly. It also includes the artifacts that are created and built, tested, and eventually deployed to a CDE through one or more CI/CD pipelines. Assessors should be careful to capture and evaluate all in-scope repositories, such as code and image repositories, as well for appropriate security and quality reviews and associated requirements such as change detection and proper logging and monitoring. The CDE's deployment controls should also be evaluated to ensure they cannot be overridden or bypassed. Organizations using CSP-managed and abstracted services should evaluate whether the CSP's implementation provides a measure of segmentation natively with the service and what impact it may have on their CDE.

8 Guidance for Specific PCI DSS Scoping and Segmentation Requirements

The following table outlines concepts and best practices for entities to consider when implementing the scope and segmentation practices in PCI DSS. While not exhaustive, this list is intended to provide foundational insights to evaluate when initiating or refining an organization's scope boundaries and for applying segmentation controls within modern network architecture.

PCI DSS Requirement	Scope and Segmentation Considerations
1.2.3-1.2.4 **Network and data-flow diagram maintenance**	All in-scope locations, network segments (VLANS, Subnets, etc.), NSCs (WAFs, Firewalls, TAPs, IDS/IPS), and other critical infrastructure that plays a part in managing these segments (log aggregation, VLAN Management Tools, NTP services, DNS, etc.), should all be considered in building accurate and effective network diagram(s). Implement automated discovery and monitoring tools, either offered by an entity's cloud provider, or by a third party, to ensure that they can scan your environment regularly and provide diagrams auto-updates. Having a centralized configuration management database (CMDB) is essential to getting an accurate diagram at any moment, which is particularly crucial in ephemeral, dynamic virtual environments.

PCI DSS Requirement	Scope and Segmentation Considerations
1.4.1 – Network security control implementation between trusted and untrusted networks	PCI DSS requires that network connections between trusted and untrusted networks are controlled. To ensure connections that do exist are sufficiently managed, organizations should consider implementing a segmentation buffer, like a demilitarized zone (DMZ), where connections from untrusted networks land before transmission of data, into trusted networks: • Cloud networks may include a hypervisor, which can be configured as a landing zone to mirror DMZ functionality. • Web application firewalls (WAF) could be architected to sit at the front end of an untrusted network boundary, and filter and monitor traffic between that untrusted network and a web application within a trusted network. This would allow a WAF to provide an additional layer of security protection between the untrusted and trusted networks, which would effectively serve a similar function as a traditional DMZ would. • Note that the implemented DMZ and any NSCs that perform DMZ functionality should be evaluated as part of PCI DSS scope, as there would be connectivity to account data. Ensure front-end CDE system components have the same segmentation controls as backend CDE system components, given there is transmission, or unrestricted connectivity to systems that can transmit CHD. It is important to protect and isolate all types of CDE systems when architecting boundaries and implementing segmentation controls between trusted and untrusted networks. Modern cloud network topology is often designed to support underlying shared architecture or an under-cloud segment (likely comprised of a hypervisor, hardware or orchestration tools). Within this underlying shared architecture/under-cloud, there could be a WAF, terminating proxy, or web front-end workload and any of these could also transmit account data. As such, be sure to include the underlying architecture/under-cloud segment of your host CDE system(s) in the evaluation of your PCI DSS scope.

PCI DSS Requirement	Scope and Segmentation Considerations
11.4.1 – Penetration testing methodology documentation	When considering a penetration testing methodology for modern network deployments, entities should test not only the application layer or the segmentation layer covered in Requirement 11.4.5, but, as part of a wholistic practice, should include testing the infrastructure and potential weaknesses in the various layers of orchestration and hypervisor technologies. Many modern technologies around cloud environments often employ open-source tooling, whether at the hypervisor layer, NSC technologies, and/or network infrastructure. For example, recently publicized supply chain attacks raise the need to review source code for open-source and third-party infrastructure tools. Establishing a robust vendor management program to assess both open-source and third-party code should be part of any penetration testing methodology that includes outside source code. Ensure source code assessments have been performed and vulnerabilities addressed.
11.4.5 – Penetration testing on segmentation controls	As with traditional firewall-segmentation technologies, effectively testing segmentation for a modern architecture should be more than a simple port scan from a single host IP outside the segment. Modern segmentation includes a variety of orchestration layers, TLS proxies, and host or network-based firewalls. Even routing is more complex in modern technologies. Testing outside the K8s environment will yield different results than within the containerized workload environment. For instance, routes for a containerized workload in a Kubernetes (K8s) environment may not be routable outside of K8s without NATed IPs. Policies may use named resources and not IPs, so testing procedure should take this into account for all segmentation variables. Many cloud service providers supply these services, but as many companies deploy hybrid or in-network cloud infrastructure, relying solely on a CSP service can be challenging.
12.5.1 – In-scope system component inventory maintenance	Given the ephemeral nature of cloud-hosted micro services and systems in a modern cloud network topology, an automated inventory tool may be helpful to address not only the more static inventory, but the continually changing, dynamic workloads. To maintain an understanding of the description of function/use of each component it may be necessary to maintain a configuration management database (CMDB) that lists components or applications by a configuration identifier and potentially a sub-identifier if it becomes necessary to understand the function of each subcomponent.

PCI DSS Requirement	Scope and Segmentation Considerations
12.5.2, 12.5.2.1 – **PCI DSS scope documentation and confirmation**	Modern cloud technologies provide new challenges to documenting scope both annually and semi-annually for service providers. Defining how the entity achieves segmentation boundaries is key to the annual and semi-annual assessment of scope. If the logic or technology behind segmentation methodologies changes between these documentation exercises, a risk-based process should be carried out to validate the effectiveness of the segmentation. For instance, if a CSP or entity relies on ZT model segmentation it is critical to validate least privilege access as part of any scope exercises. Various cloud deployments and responsible parties come into play with modern cloud environments. For instance, some modern cloud technologies may include deployments where the entity manages both the infrastructure, such as hardware, network equipment, hypervisors, and orchestration, plus the workloads that host account data. Others may deploy Platform as a Service on top of providers “below the line” technology and the entity is only responsible for the policy that segments the CDE from the rest of the entity’s hosted environment. In either scenario, it is important to clearly document how the entity validates their PCI DSS scope based on these factors.
12.8 – TPSP relationship management	Maintain an accurate and detailed inventory of cloud service providers: • Include service contracts, service-level agreements, policies, and shared security and compliance responsibilities. • Review Appendix A: Sample Inventory and Appendix B: Responsibility Matrix below for specific guidance on how to support TPSP relationship management. ***Note:*** *Not all CSPs provide PCI DSS responsibility matrices that extend to the microservice level. Some CSPs may generalize responsibilities based on implementation type. It is an organization’s responsibility to maintain documentation specific to its implementation. These compliance responsibilities may need to be identified at the microservice level, as responsibilities may vary from service to service even within a single CSP.*

9 Segmentation Strategies for Modern Network Architectures

Implementing effective segmentation techniques is a crucial component to reducing the PCI DSS assessment scope and streamline the assessment process. When properly configured, implemented, and verified, organizations can minimize costs and effort associated with implementing and maintaining PCI DSS controls. This section describes commonly used segmentation strategies that, when applied correctly, can effectively de-scope system components.

While the sample deployments covered in this document represent common scenarios, it is important to remember that each organization's specific business and technical requirements and overall security strategy should drive the design and implementation of network segmentation controls across their environment(s). It is also essential to acknowledge that introducing these techniques can add complexity to the PCI DSS assessment process and may require a specialized skillset to evaluate and assess.

Each technique outlined in this section includes critical parameters or implementation conditions that should be met to achieve effective de-scoping. Organizations should seek guidance from a QSA or an ISA with the skillset to validate the approach and implementation of these techniques, as applicable to their specific environment.

9.1 Segmentation Fundamentals

In modern network architectures, various components collaboratively form a framework for deploying, managing, and scaling containerized applications. The following components should be understood to properly implement network segmentation within container orchestration[4] and service-mesh environments.

9.1.1 Containers

Containers serve as lightweight and portable units that encapsulate software, including its code and dependencies. These containers are deployed within a group of one or more containers to allow for networking and resource sharing.

9.1.2 Container Groups

Container groups—e.g., pods—encapsulate one or more containers that share the same network namespace and storage resources. They serve as the atomic unit for scaling applications horizontally, with multiple containers within a group typically addressing closely related processes. Container groups provide a shared context for containers, allowing them to communicate over localhost and share storage volumes.

9.1.3 Services

Services are essential for managing communication and load distribution between microservices in a container orchestration environment. They provide a consistent endpoint through a virtual IP address and port, which lets clients easily adjust to changes in container groups. Since containers in these environments often scale up or down, are rescheduled, or change IP addresses, services allow clients to

[4] Refer to PCI SSC *Guidance for Containers and Container Orchestration Tools* Information Supplement for more information about the use and functionality of containers and container orchestration tools.

find and connect to necessary microservices without having to deal with the underlying complexity of the containers' temporary lifespans.

9.1.4 Namespace

Namespaces are a way to group and isolate resources, such as containers and services, allowing for multiple isolated environments within a single instance of a container orchestration platform to exist independently. This isolation is essential for security and resource management, and to ensure that each set of services or applications has its own space and can be managed independently. In multi-tenant environments, namespaces help to prevent naming conflicts and provide control over who can access which resources, making it easier for different teams or projects share the same cluster securely.

9.2 Segmentation Approaches and Examples

9.2.1 Proxy Pattern

The proxy pattern employs a structural design, using a proxy to control and authorize traffic to and from the CDE, effectively reducing the scope of systems and networks that are not part of the CDE. In this architecture, the proxy serves as a crucial interface component as a network security control for the CDE and would be included in scope for a PCI DSS assessment. Non-CDE systems and networks that interact through the proxy are deemed out of scope, as the proxy intercepts and manages their communications, ensuring that they do not directly or indirectly access, or impact the security of the CDE.

This pattern is versatile and can be integrated into various environments, including public clouds, private clouds, and traditional non-cloud IT environments. Using sidecar proxies, which are deployed alongside each service or application component, enables fine-grained control of the traffic and security policies applied to a single component. Similarly, waypoint proxies can establish predetermined points of traffic routing and inspection within the network, further enhancing the security of the system with minimal changes to existing architectures.

By leveraging these types of proxies, organizations can effectively segment a PCI DSS in-scope environment with minimal restructuring required to embed the proxy components into their current architectures.

Implementation Criteria:

- **Perimeter Placement**: The proxy is strategically placed outside the CDE boundary, serving as a clear marker for where security controls are defined.
- **De-scoping Criteria**: To fulfill the proxy pattern's intent of segmenting non-CDE assets, traffic passing through the proxy should not carry account data. This can be verified through data in-transit classification ensuring a clear separation of in-scope and out-of-scope systems.
- **Connection Termination**: The proxy operates as a termination point for incoming connections, creating a secure, monitored link into the CDE. This is crucial for maintaining the integrity and security of data as it transitions into the protected environment.

- **Vulnerability Management**: When an IDS (Intrusion-detection system) is in place, the proxy's role can be strategically limited to traffic management rather than vulnerability inspection, enhancing resource efficiency.

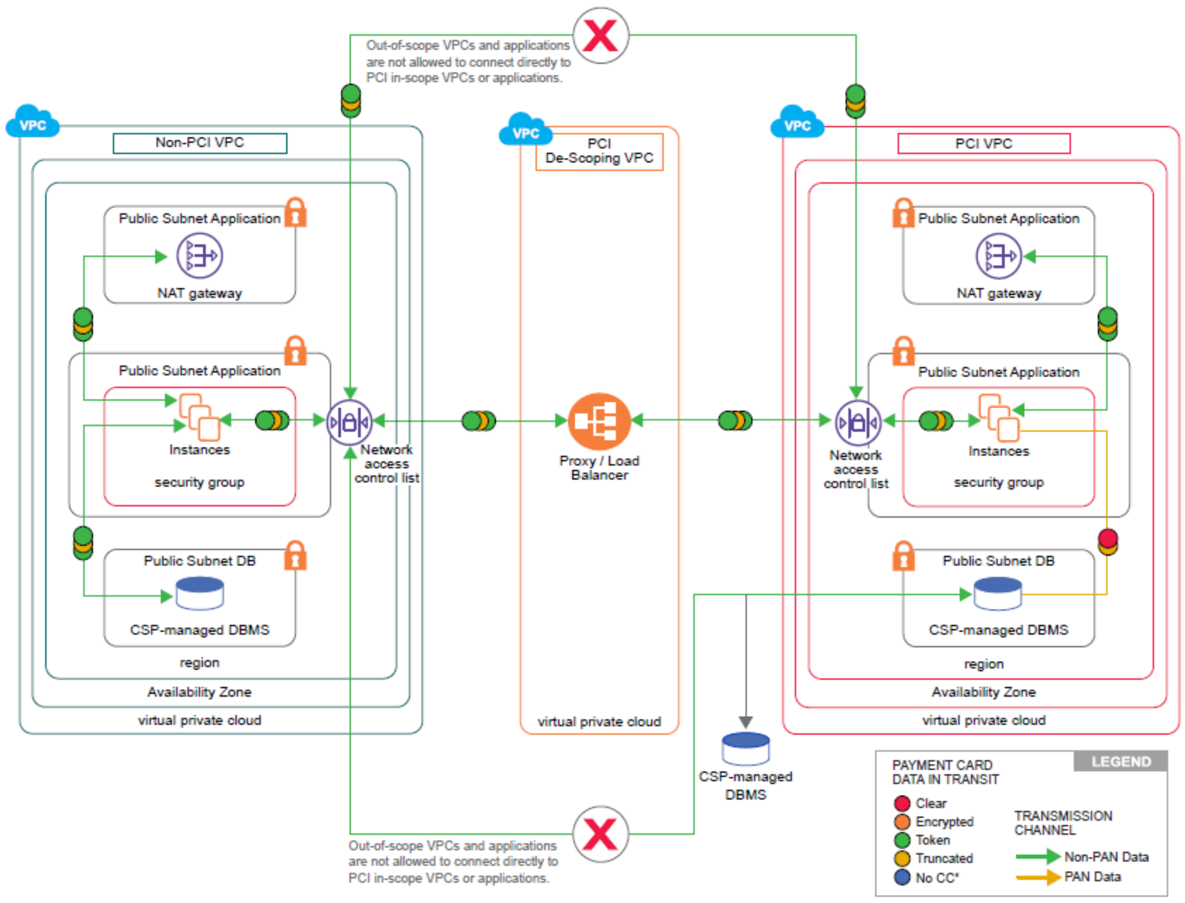
9.2.2 Managed Proxy

The managed proxy approach is where a proxy server is used as a network access control and security policy enforcement point and provides anonymity by managing network traffic between different segments. This server, usually maintained and run by a cloud service provider (CSP), falls within the CSP's PCI DSS assessment scope. This arrangement enables cloud service customers (CSCs) to rely on the CSP's evidence of PCI DSS compliance, as per the specifications delineated in Requirement 12.9.2.

Implementation Criteria:

- **Traffic Routing**: Traffic from de-scoped virtual private clouds (VPCs) or applications must traverse through a specialized de-scoping VPC and proxy prior to reaching PCI-designated VPCs or applications.
- **Communication Prohibition**: Direct communication is strictly forbidden between de-scoped entities and connected-to and security-impacting systems.
- **Proxy Location**: The managed proxy resides within a PCI DSS de-scoping VPC.
- **Traffic Redirection:** Out-of-scope systems needing non-PCI DSS data from connected-to, security-impacting systems use a security group to direct the traffic through the proxy.
- **Traffic Management**: The proxy disrupts the traffic flow to reassign new IP addresses and then forwards it through the security controls to reach the intended connected-to, security-impacting systems.

In the image below, the managed proxy is in orange



9.2.3 API Gateway

APIs and API gateways are integral to microservice architectures, offering advantages as proxies by disrupting direct network traffic to enhance security. To function as an effective de-scoping solution, API gateways should implement native Transport Layer Security (TLS) interception. This involves terminating the original connection and initiating a new one, which is crucial for the gateway to be considered a de-scoping proxy.

Implementation Criteria:

- **TLS Interception:** The API gateway must intercept and terminate TLS connections, reestablishing them under the new parameters to function as a de-scoping proxy.
- **Traditional System Updates:** Traditional systems that rely on uninterrupted end-to-end TLS traffic should be updated or re-engineered to integrate this mechanism and align with applicable PCI DSS requirements.

9.2.4 Intermediate Managed Services

Intermediate CSP managed services, introduce a break in end-to-end connections for risk mitigation, which can allow them to function as enhanced proxies. They add latency but maintain the real-time essence of transmissions, often resulting in asynchronous communication. These services, such as cloud storage buckets, managed file transfer services (MFTS), and Pub/Sub (publish/subscribe) queues, stand as intermediaries in the communication process, which could potentially support segmentation from in-scope systems. These services, typically presented through API interfaces, add a layer of logic to support a variety of use cases. The decoupling of data transfer from network performance through these services is a commonly adopted architectural pattern and does not impact the scoping of clients if the CSP-managed services have been verified to be PCI DSS compliant.

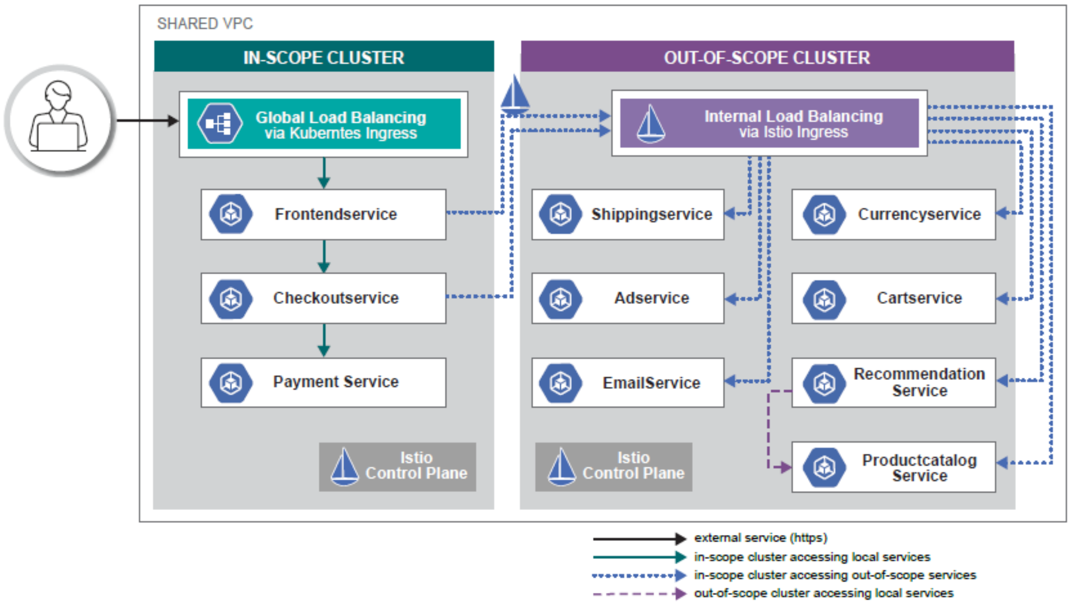
Implementation Criteria:

- **Scope of Storage Buckets**: Storage buckets should be PCI DSS in-scope components if they contain account data.
- **Service Assessment**: Intermediate services should be evaluated for PCI DSS compliance, regardless of defined function and service in their specific use case, based on the assumption that they may impact the security of their customer’s CDE.

9.2.5 Native Service Mesh

A service-mesh architecture facilitates the secure, segmented communication required within microservices-based applications. For instance, it allows the deployment of TLS at the pod level within an application environment. Services within these pods can utilize mTLS to validate and confirm the identities of other services operating on separate pods, ensuring authenticated and encrypted service-to-service communication.

Within this service mesh, communications are channeled through client-side and server-side proxies. For example, Envoy, a high-performance proxy, leverages SPIFFE IDs (Secure Production Identity Framework For Everyone) to establish and maintain secure mTLS connections between services. This enforces not only network-level isolation but also secure service identity verification across the entire service mesh.



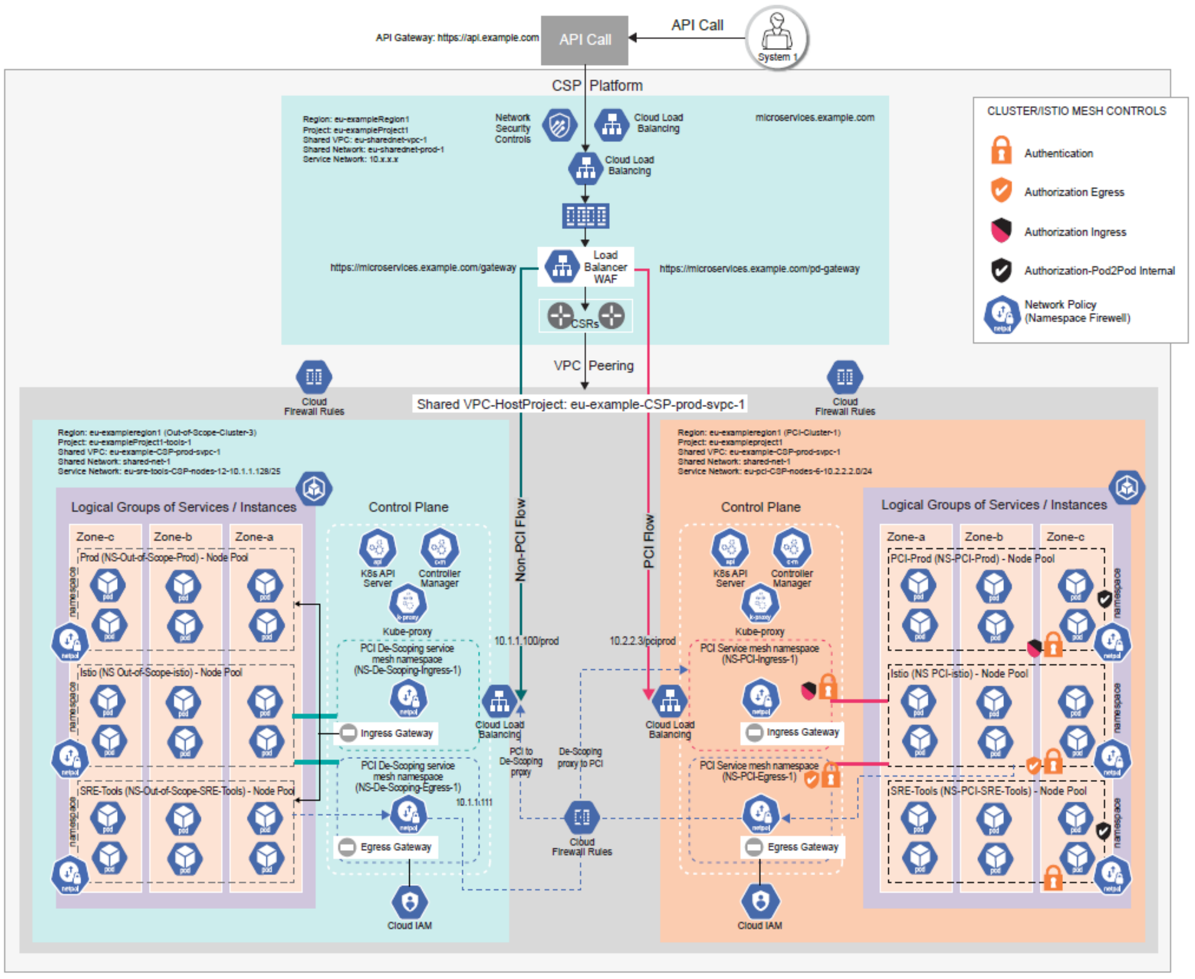
In the context of a shared VPC, a service mesh ingress gateway acts as a secure entry point for traffic entering the mesh. Specifically, it can function as a secured HTTPS load balancer that manages traffic entering the CDE, ensuring secure ingress without compromising the sensitive internal environment. Similarly, a service-mesh egress gateway securely manages the traffic leaving the CDE, effectively controlling communication to connected-to and security-impacting systems or out-of-scope services, depending on the specific network segmentation needs.

The use of service-mesh gateways reflects the principles of the traditional proxy pattern, employing service-mesh-specific ingress and egress controllers to facilitate the secure and controlled traffic flow. In this model:

- Ingress traffic into the CDE is routed through an ingress gateway for the CDE and proxied by the sidecar proxies for in-scope services within the CDE.
- Egress traffic from the CDE is routed through an egress gateway for the CDE and proxied by the sidecar proxies for in-scope services within the CDE.
- Ingress traffic to out-of-scope systems is routed through an ingress gateway and proxied by the sidecar proxies for out-of-scope services outside the CDE.
- Egress traffic from out-of-scope systems is routed through an egress gateway and proxied by the sidecar proxies for out-of-scope services outside the CDE.
- The service mesh should be configured to deny by default and allow only explicitly authorized connections.

Implementation criteria:

- **Policy Configuration**:
  - Kubernetes network policies and namespaces are configured to control pod-to-pod traffic and provide an additional layer of logical segmentation within the in-scope cluster.
  - An authentication policy within the service-mesh control plane of the CDE Systems cluster is configured to enforce strict mutual TLS authentication by default, blocking any unencrypted traffic and ensuring all communications are authenticated and encrypted.
  - An authorization policy within the service-mesh control plane of the CDE Systems is set to a default-deny mode, ensuring that only explicitly permitted requests are allowed through the mesh.
  - The service-mesh ingress gateway within the CDE Systems is configured with an authorization policy to authenticate and authorize all incoming traffic from connected-to and security-impacting systems.
  - The service-mesh egress gateway within the CDE Systems is set up to authenticate and authorize all outbound traffic to maintain the integrity and security of data exiting the CDE Systems
- **Data Classification**: A cardholder data classification policy is configured in the service mesh to enforce classification of CHD in-transit across in-scope and out-of-scope proxies to verify segmentation enforcement.


API Gateway: https://api.example.com
API Call
API Call
System 1
CSP Platform
Region: eu-exampleRegion1
Project: eu-exampleProject1
Shared VPC: eu-sharednet-vpc-1
Shared Network: eu-sharednet-prod-1
Service Network: 10.x.x.x
Network Security Controls
Cloud Load Balancing
microservices.example.com
Cloud Load Balancing
https://microservices.example.com/gateway
Load Balancer WAF
https://microservices.example.com/pd-gateway
CSRs
VPC Peering
Cloud Firewall Rules
Shared VPC-HostProject: eu-example-CSP-prod-svpc-1
Cloud Firewall Rules
CLUSTER/ISTIO MESH CONTROLS
Authentication
Authorization Egress
Authorization Ingress
Authorization-Pod2Pod Internal
Network Policy (Namespace Firewell)
Region: eu-exampleregion1 (Out-of-Scope-Cluster-3)
Project: eu-exampleProject1-tools-1
Shared VPC: eu-example-CSP-prod-svpc-1
Shared Network: shared-net-1
Service Network: eu-sre-tools-CSP-nodes-12-10.1.1.128/25
Logical Groups of Services / Instances
Zone-c
Zone-b
Zone-a
Prod (NS-Out-of-Scope-Prod) - Node Pool
Istio (NS Out-of-Scope-istio) - Node Pool
SRE-Tools (NS-Out-of-Scope-SRE-Tools) - Node Pool
namespace
pod
netpol
Control Plane
K8s API Server
Controller Manager
Kube-proxy
PCI De-Scoping service mesh namespace (NS-De-Scoping-Ingress-1)
Ingress Gateway
PCI De-Scoping service mesh namespace (NS-De-Scoping-Egress-1)
10.1.1.111
Egress Gateway
Cloud IAM
Non-PCI Flow
10.1.1.100/prod
Cloud Load Balancing
PCI to De-Scoping proxy
De-Scoping proxy to PCI
Cloud Firewall Rules
PCI Flow
10.2.2.3/pciprod
Cloud Load Balancing
Region: eu-exampleregion1 (PCI-Cluster-1)
Project: eu-exampleproject1
Shared VPC: eu-example-CSP-prod-svpc-1
Shared Network: shared-net-1
Service Network: eu-pci-CSP-nodes-6-10.2.2.2.0/24
Control Plane
K8s API Server
Controller Manager
Kube-proxy
PCI Service mesh namespace (NS-PCI-Ingress-1)
Ingress Gateway
PCI Service mesh namespace (NS-PCI-Egress-1)
Egress Gateway
Cloud IAM
Logical Groups of Services / Instances
Zone-a
Zone-b
Zone-c
PCI-Prod (NS-PCI-Prod) - Node Pool
Istio (NS PCI-istio) - Node Pool
SRE-Tools (NS-PCI-SRE-Tools) - Node Pool
namespace

9.2.6 Service-Mesh Policy Configuration for Zero Trust

Authentication Policy

Peer authentication policies specify the mutual TLS mode Istio enforces on target workloads. The following modes are supported:

- **PERMISSIVE:** ⚠Workloads accept both mutual TLS and plain-text traffic.

  Use Case: This mode is most useful during migrations when workloads without sidecar cannot use mutual TLS. Once workloads are migrated with sidecar injection, the mode should be switched to STRICT.

- **STRICT:** 🛡Workloads only accept mutual TLS traffic.

  Use Case: Ideal for environments where strict security controls are required. All communications are encrypted and authenticated.

- **DISABLE:** ⚠Mutual TLS is disabled. From a security perspective, this mode should not be used unless the organization provides its own security solution.

  Use Case: This mode should only be used when the organization provides its own security solution, as it leaves traffic unprotected.

Mesh-wide peer authentication policies with an unset mode use the PERMISSIVE mode by default.

For example, the following peer authentication policy requires all workloads in namespace “PCI-namespace” to use mutual TLS:

```
apiVersion: security.istio.io/v1beta1
kind: PeerAuthentication
metadata:
  name: "example-authentication-policy"
  namespace: "PCI-namespace"
spec:
  mtls:
    mode: STRICT
```

Best Practices:

- Always Use STRICT for Critical Workloads: Ensure that any sensitive or critical workloads are configured with STRICT mode to enforce maximum security.
- Avoid DISABLE Mode: Use this mode only when absolutely necessary and ensure alternative security measures are in place.

For more information, see Istio's Documentation.

Authorization Policy

To configure a default-deny policy, Istio authorization policy can be defined to deny all requests with an ALLOW policy that matches nothing, as shown in the following example. If there are no other ALLOW policies, requests will always be denied because of the “deny by default” behavior.

```
apiVersion: security.istio.io/v1
kind: AuthorizationPolicy
metadata:
  name: allow-nothing
spec:
  action: ALLOW
```

It is recommended to start with the allow-nothing policy and incrementally add more ALLOW policies to open more access to the workload. For example, configure authorization policy as shown below to allow GET requests from all workloads in prod-uptime-namespace to the "/info" path and POST requests to the "/data" path from all workloads in namespace prod-uptime-namespace to the PCI-in-scope-workload.

```
apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
  name: PCI-in-scope-workload                      ‘ in-scope workload
  namespace: PCI-namespace                         ‘ in-scope namespace
spec:
  action: ALLOW
  rules:
  - from:
    - source:
        namespaces: ["prod-uptime-namespace"]      ‘ source connected-to namespace
    to:
    - operation:
        methods: ["GET"]                           ‘ GET operation allowed
        paths: ["/info*"]                          ‘ URL path allowed
    - operation:
        methods: ["POST"]                          ‘ POST operation allowed
        paths: ["/data"]                           ‘ URL path allowed
```

Best Practices:

- Start Restrictive: Begin with an allow-nothing policy to enforce strict security from the outset.
- Incremental Access: Gradually add specific ALLOW policies to permit only the necessary actions, reducing the attack surface.

9.2.7 From theory to practice

Kubernetes Network Policies and Service-Mesh Frameworks

Kubernetes Network Policies and service-mesh frameworks—e.g., Istio—serve complementary roles in managing traffic within containerized environments. While Kubernetes Network Policies provide fundamental network-level segmentation, allowing one to specify which pods can communicate based on labels and ports, a service mesh offers advanced service-to-service communication control, including sophisticated traffic routing and security policies like mutual TLS. Employing both enables a layered defense strategy: Kubernetes Network Policies establish a secure baseline for pod interactions, and a service mesh adds a detailed operational and security overlay for fine-grained control over microservices, enhancing the overall management and security of containerized applications.

Complementary Roles of Kubernetes Network Policies and Service Mesh:

- Kubernetes Services: Act as stable endpoints for accessing a set of pods. These services provide a consistent way to interact with various containers, storage, APIs, and data without understanding the underlying network configuration.
- Kubernetes Network Policies: Provide basic traffic segmentation, enabling you to define how pods can interact based on labels and ports. In the following example, a network policy is created that permits traffic from pods with the “frontend” label to pods labeled as “backend,” restricting the flow to a specific port and protocol (TCP/80):

▪

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-frontend-to-backend
  namespace: default
spec:
  podSelector:
    matchLabels:
      app: backend
  ingress:
    - from:
      - podSelector:
          matchLabels:
            app: frontend
      ports:
      - protocol: TCP
        port: 80
```

  - A service-mesh framework introduces a sophisticated layer that augments the capabilities provided by Kubernetes. It operates at a higher level than network policies, focusing on the service-to-service communication rather than just pod-to-pod. While Kubernetes network policies act at the network level, controlling traffic based on the IP addresses and ports, a service-mesh framework applies policies directly to the service level. It can dictate more complex operational behaviors, such as service discovery, load balancing, failure recovery, metrics collection, and observability, as well as sophisticated routing and policy enforcement.

Example of Service-Mesh Configuration: The following service-mesh configuration sets up a gateway for ingress traffic to a **front-end** service and ensures that this service can only communicate with a **back-end** service:

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: frontend-gateway
spec:
  selector:
    service-mesh: ingressgateway # use the service mesh's default gateway implementation
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "frontend.example.com"

apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: frontend-virtualservice
spec:
  hosts:
  - "frontend.example.com"
  gateways:
  - frontend-gateway
  http:
  - match:
    - uri:
        prefix: "/"
    route:
    - destination:
        host: frontend
        port:
          number: 80

apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
  name: frontend-to-backend-only
  namespace: frontend-namespace
spec:
  selector:
    matchLabels:
      app: frontend
  action: ALLOW
  rules:
  - to:
    - operation:
        ports: ["80"]
        hosts: ["backend.namespace.svc.cluster.local"]
```

- In this configuration, the Gateway defines the entry point for traffic coming from outside the cluster to the **front-end** service. The VirtualService then routes the incoming traffic to the actual **front-end** service based on the defined rules, like URI paths. Lastly, the authorization policy restricts the **front-end** service so that it can only send traffic to the **back-end** service on port 80, implementing micro-segmentation at the service-mesh level.
- A service-mesh framework can manage security and compliance more adeptly by providing end-to-end encryption, service-specific access control, and authentication mechanisms that go beyond the capabilities of Kubernetes network policies. It offers a dynamic approach to enforce policies that adapt to the transient and agile nature of microservices, like dynamically rerouting traffic or implementing fault injection for testing.

Kubernetes network policies establish the security parameters for how different parts of an application can communicate at the pod level. In contrast, a service-mesh framework extends this by adding a rich set of traffic control and security features at the service level, offering fine-grained control over the behavior of the microservices in an architecture.

Best Practices**:**

- Start with Least Privilege: Always begin with restrictive policies and gradually open up permissions as necessary to minimize exposure.
- Regular Audits: Conduct regular audits of your network policies and service mesh configurations to ensure they are still aligned with your security objectives.
- Compliance Checks: Ensure that all policies adhere to regulatory compliance standards, especially when dealing with sensitive data

Testing & Monitoring

Testing these policies in production is a critical step to ensure that they work as intended and do not disrupt legitimate traffic. The following is a two-phase approach to validate new and existing policies:

Monitoring and Observability Phase:

Before applying any new policy, ensure that monitoring and observability tools are in place. With a service mesh, it is possible to leverage its inherent telemetry features to observe the traffic patterns. When applying the new policies, monitor these dashboards closely for any denied traffic or unexpected behavior. Set up alerts for unusual spikes in error rates, latency, or CHD leakage that could indicate configuration issues.

Implement extensive logging. When a network policy denies traffic, it should be logged and reviewed. These logs can offer immediate feedback on the impact of policy changes.

Testing Phase:

1. **Synthetic Transactions:** Generate synthetic traffic that mimics normal user behavior to validate that legitimate requests are allowed by the policies. These transactions should cover various interactions between the front-end and back-end services.

2. **Chaos Engineering:** Introduce controlled failures to see how network policies react. This includes attempting to route traffic from unauthorized services to the back end and ensuring it is correctly denied.

3. **Automated Testing:** Include network policy validation in the CI/CD pipeline. Automated tests should assert that the only traffic allowed to the back-end service is through the front-end service and that direct traffic to the back-end service is blocked.

4. **Staged Rollout:** Use a canary deployment strategy to incrementally apply current policies to a subset of the production traffic. This minimizes the impact of unforeseen issues and allows quick roll back if needed.

5. **Data In-Transit Monitoring:** Implement monitoring tools capable of classifying data in-transit. This will allow confirmation that not only is the traffic flowing according to policy, but that the data content itself adheres to compliance requirements and internal data handling protocols. It is a way to continuously ensure that sensitive data is only accessed and handled by authorized services.

6. **Performance Impact Assessment:** Evaluate the impact of the network policies on the system's performance. This includes measuring the latency and throughput before and after the policy application to ensure there is no degradation.

By closely monitoring the system's behavior and conducting thorough testing it is possible to confidently apply network and service-mesh policies in a production environment. It is important to have a rollback plan ready in case something does not go according to plan.

Documentation

Maintaining detailed documentation for network and authentication policies, outlining their purpose, scope, and expected behavior is crucial for onboarding new team members and facilitating effective troubleshooting throughout the service-mesh platform.

10 Further Resources and Guidance

The following resources and guidance documents are available from the Document Library on the PCI Security Standards Council website.

PCI DSS

PCI SSC Information Supplements:

- *Cloud Computing Guidelines*
- *Guidance for Containers and Container Orchestration Tools*
- *Guidance for PCI DSS Scoping and Network Segmentation*
- *Third-Party Security Assurance*
- *Penetration Testing Guidance*
- *PCI DSS Virtualization Guidelines*

Additionally, common terms in the payment card industry used within this document are listed in PCI DSS v4.0.1. Appendix G PCI DSS Glossary of Terms, Abbreviations, and Acronyms.

Appendix A – Sample Inventory

This sample inventory[5] is intended to illustrate how system components and associated security responsibilities can be identified and organized when using modern network architecture. Entities may use this sample document, in addition to their responsibility matrix (in Appendix B of this document), to support PCI DSS scoping discussions, both internally and with third-party service providers. This type of inventory can help establish, maintain, and communicate system component security ownership and responsibilities specific enough to capture the granular system component types often present in modern network architecture, particularly in virtualization technologies.

Note: *This is intended as a general example only. It may be necessary to reorganize the different technology layers or define additional component characteristics as applicable to a particular environment. Entities may wish to identify responsibilities for each system component in greater detail than provided for here.*

SAMPLE INVENTORY			
Type/Layer *For example: Data, APIs/GUIs, applications, network, physical, processing/memory.* ***Note:*** *Actual layers will vary depending on the structure of provider service offerings. Examples provided only.*			
Component Description/ Purpose *For example: Firewall, OS, application, web server, hypervisor, router, database, etc.*			
Type of Component *For example: Physical, logical, or virtual? Static or dynamic?*			
Number of Components *Number of components used in relation to entity's service*			
Implementation Notes *Defined usage, location, etc., as applicable*			
Responsibility Grouping *For example: Compute (CDE), Compute (Connected-to), Networking, Storage, Security, Undercloud*			
Responsibility for Securing Component *For example: Provider only, customer only, or shared*			

[5] Adapted from PCI SSC Cloud Computing Guidelines Information Supplement, Appendix B: Sample Inventory.

Appendix B – Sample PCI DSS Responsibility Matrix

A PCI DSS responsibility matrix may help to clarify and confirm how responsibilities for maintaining PCI DSS requirements are shared between an entity (Customer) and their third-party service provider (TPSP). Responsibilities should always be defined in written agreements.

The table[6] below is a sample that can be used by providers and customers in hybrid cloud environments to communicate the responsibilities and considerations for each PCI DSS requirement, and includes:

- Does the provider perform/manage/maintain the required control?
- How is the control implemented, and what are the supporting processes—e.g., process for patch updates would include details of testing, scheduling, approvals, etc.?
- What layers of the cloud architecture are covered by the provider for the requirement? What layers of the architecture are not covered by the provider and are specifically the responsibility of the customer?
- How will the provider provide ongoing assurance or evidence to the customer that controls are met—for example, periodic reports, real-time notifications, results of testing, etc.?

Note: *This Appendix is intended for optional use at the discretion of the entity and/or TPSP; completion of this Appendix is not a requirement nor is it necessary for an entity or TPSP to complete this Appendix to meet any PCI DSS requirements.*

PCI DSS Requirement	Responsibility			Specific coverage /scope of entity responsibility	Specific coverage/scope of TPSP responsibility	How and when TPSP will provide evidence of compliance to entity
	TPSP Only	Entity Only	Shared			
1.1.1	☐	☐	☐			
1.1.2	☐	☐	☐			
1.2.1	☐	☐	☐			
1.2.2	☐	☐	☐			
1.2.3	☐	☐	☐			
1.2.4	☐	☐	☐			
1.2.5	☐	☐	☐			
1.2.6	☐	☐	☐			
1.2.7	☐	☐	☐			
1.2.8	☐	☐	☐			

[6] Adapted from PCI SSC Cloud Computing Guidelines Information Supplement, Appendices A and C

Appendix C – Other Modern Network Architecture Implementation Considerations

Implementing modern network architecture within a payment ecosystem involves navigating a complex landscape of security, compliance, and operational challenges—above and beyond scoping and segmentation issues mentioned in previous sections. As organizations evolve to more advanced network topology, they should consider how to ensure the ongoing protection of payment account data and adherence to PCI DSS controls.

In all instances, organizations should carefully manage third-party access. Both the organization and third parties should understand their responsibilities[7] for establishing effective segmentation controls, particularly in cloud environments, to prevent unauthorized access and data breaches. The sample system component inventory in Appendix A and sample responsibility Matrix in Appendix B are effective tools to establish and maintain an understanding of these responsibilities. The two PCI SSC information supplements noted below provide extensive guidance on how to sustain viable TPSP relationships:

- *PCI SSC Third-Party Security Assurance Guidance* Information Supplement
- *PCI SSC Cloud Computing Guidelines* Information Supplement

General considerations include:

Identity and Access management

- Modern network architectures often include intricate permission structures, such that a detailed understanding of their configuration and operation is required to ensure that the principles of least privilege and need to know are implemented accurately and efficiently. Understanding the overall set of permissions in interconnected systems is essential to address issues like permission creep or extraneous access.
- Avoid relying on permission boundaries as segmentation boundaries, as permissions alone may not provide adequate segmentation to secure sensitive data effectively.
- Enforcing configuration access control policies with proper approval methods is essential to prevent unauthorized access to critical systems and data.

Data Sovereignty Management

- Companies should be aware of local data sovereignty laws, which may restrict data transmission and storage locations in CSP environments. This requires careful consideration of data storage decisions, applicable laws, and any resulting potential impacts to the business. Robust data governance programs can help to support these regulatory and sovereignty requirements, addressing data ownership, lifecycle management, and data subject requests.

[7] Refer to PCI DSS v4.0.1. Requirement 12.8 for third-party service provider (TPSP) relationship management and Requirement 12.9 for how TPSPs are expected to support their customers' PCI DSS compliance.

Assessor familiarity with an organization’s modern technology

- Gain assurance that the assessor has a base-level understanding of the tools and technology in use by the organization. Per the QSA Qualification Requirements, each QSA should have sufficient skills and experience to conduct technically complex security assessments, however if the organization has incorporated cutting-edge technology, it may be difficult for that assessor to already be familiar with the controls within that environment.
- Allocate enough time before and/or during a PCI DSS assessment for an assessor to familiarize themselves with the modern technology in use to sufficiently evaluate the security controls involved.

Acknowledgements

PCI SSC would like to acknowledge the contribution of the Scoping and Segmentation for Modern Network Architectures Special Interest Group (SIG) in preparation of this document. This group consists of representatives from the following organizations:

7Security GmbH

Accudata Systems

A-LIGN Compliance and Security, Inc., dba A-LIGN

Amazon

AMR Cybersecurity Ltd

AWS Security Assurance Services LLC

Bank of New Zealand

Banner Health

Barclays Bank Plc

BDO USA, LLC

Bell Canada

Blackbaud, Inc.

Block, Inc.

British Columbia Lottery Corporation

British Telecommunications (BT)

BSI Cybersecurity and Information Resilience Ireland Limited, dba BSI Group

Capital One Financial Corporation

Center for Internet Security

Checkout.com

Cisco

Claranet Limited

Coalfire Systems

ControlGap

Crowe Horwath LLP

CVS Caremark

CyberCX Pty Ltd.

EasyPark

Emirates/Dnata

Enterprise Mobility

Equifax

Ernst & Young LLP

Foregenix

Forvis Mazars, LLP

Gertec Brasil Ltda

Global Payments Direct Inc.

Google

Grubhub Holdings Inc.

Interactive Communications International, Inc.

Intersec Worldwide

INTERVALLE TECHNOLOGIES

JP Morgan Chase

Kirkpatrick Price, Inc. dba Raven Eye

Lattimore, Black Morgan and Cain, PC

LeakSignal

Marriott Vacations Worldwide Corporation

Moneris	Schwarz IT KG
NBCUniversal Media, LLC	Secure Technology Integration Group LTD
NCC Group	Secured Net Solutions Inc.
NCC Services	SecurityMetrics, Inc.
Nelnet, Inc.	SISA Information Security
Oracle Corporation	Smart Payment Association SPA
PagoNxt Merchant Solutions	SRC Security Research & Consulting GmbH
QVC Inc.	Stripe, Inc.
Rackspace	Target Corporation
Redbanc S.A.	TELUS Security Solutions
Resources Connection LLC	Toast, Inc.
Risk Analytics, LLC	U.S. Bancorp
RSI Systems Inc.	usd AG
RSM US LLP (formerly McGladrey LLP)	VikingCloud
RubinBrown LLP	Walmart, Inc.
SAP SE	Weaver and Tidwell, L.L.P.
Saudi Payments	Worldline SA
Schellman Compliance, LLC	Zeta FZ-LLC DMCC

About the PCI Security Standards Council

The PCI Security Standards Council (PCI SSC) is a global forum for the ongoing development, enhancement, storage, dissemination and implementation of security standards for account data protection. Our role is to enhance global payment account data security by developing standards and supporting services that drive education, awareness, and effective implementation by stakeholders. To learn more, please visit pcisecuritystandards.org.