# Architecting on Amazon ECS for PCI DSS Compliance

*July 2020*

# Notices

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

# Contents

# Abstract

Companies are increasingly adopting the use of microservices and containers within AWS to support their sensitive data workloads. This paper outlines best practices for configuring Amazon Elastic Container Service (Amazon ECS) to support customers' Payment Card Industry Data Security Standard (PCI DSS) version 3.2.1 compliance needs. This paper is not comprehensive of all of the PCI DSS controls as some are not applicable to containers and customer environments may vary.

The intended audience is system architects, developers, and security personnel who are interested in architecting their AWS Cloud environments for PCI DSS compliance. This document was developed by AWS Security Assurance Services, LLC (AWS SAS), which is a fully owned subsidiary of Amazon Web Services. AWS SAS is an independent PCI Qualified Security Assessor company (QSAC) that provides AWS customers and partners with specific and prescriptive information for achieving PCI DSS compliance on the AWS cloud. As a PCI QSAC, AWS SAS can interact with the PCI Security Standards Council (SSC) or other PCI QSACs under the confidentiality and contractual framework of PCI Security Standards.

# Introduction

The Payment Card Industry Data Security Standard (PCI DSS) provides technical and operational guidance on securing payment card processing environments that is applicable to people, processes, and technology. Entities that store, process, or transmit cardholder data (CHD) must validate compliance of their cardholder data environment (CDE) against the PCI DSS. Examples of such entities include merchants, payment processors, and service providers.

AWS provides many services that have been attested to have met PCI DSS compliance, and companies can leverage these services to support their own compliance efforts. One area of continued growth is the use of containerized solutions. Containers allow applications to be abstracted from their underlying host. Containers facilitate a run-anywhere, consistent environment that can quickly start and stop based on compute needs.

The benefits of transitioning workloads to container services are well noted and range from platform independence to deployment speed and resource efficiency. Customers should be aware of the ways that they can apply security best practices in their containerized CDEs.

Container functions are typically architected to perform primary tasks, which in turn creates a distributed environment. The services implemented by containers become more network interdependent and require scheduling, scaling, and resource management. Unlike virtual machines, containers share the operating system's kernel. AWS provides strong security isolation between your containers, ensures you are running the latest security updates, and gives you the ability to set granular access permissions for every container. When running multiple containers on a single operating system, all of the containers may share a common network interface.

The considerations for application security still apply in the containerized world, additionally customers should ensure that appropriate protections exist for the management layer. Poorly coded applications running within containers will still exhibit application layer vulnerabilities much like what is defined in the Open Web Application Security Project's (OWASP) Top 10 vulnerability list.

AWS container offerings include Amazon Elastic Container Service (Amazon ECS) and Amazon Elastic Kubernetes Service (Amazon EKS). Each service supports deployment on either AWS Fargate or Amazon Elastic Compute Cloud (Amazon EC2). For Amazon EC2 deployment, customers manage the underlying EC2 instances running the

containers. AWS Fargate is a serverless compute engine and removes the need to provision and manage Amazon EC2 instances.

## PCI DSS Compliance Status of AWS Services

AWS establishes itself as a PCI DSS Service Provider that supports our customers in meeting their compliance requirements. The scope for each service assessed assumes that any data provided by the customer could include primary account numbers (PAN), sensitive authentication data (SAD), or may impact the security of such data. AWS services listed as PCI DSS compliant are assessed as if they store, process, or transmit cardholder data on behalf of customers. This includes physical security requirements for AWS datacenters that support PCI DSS in-scope services.

At the time of writing, AWS completed its most recent PCI DSS assessment in July 2019. The AWS Services in Scope by Compliance Program website lists the AWS services that were included in the annual PCI DSS assessment, along with all other services by compliance program. AWS service compliance is continually maintained and new services are often launched compliant (meaning new services that are a subset of a compliant service will inherit its compliance). Customers can access AWS compliance documentation through the AWS Management Console using AWS Artifact.

A service listed as PCI DSS compliant does not mean that by default the use of that service makes a customer's environment compliant. Rather, it means the service has the ability to be configured to meet PCI DSS requirements. Where parameters are accessible and configurable by customers, it is the customer's responsibility to ensure these parameters are configured to meet compliance requirements. There may be additional AWS services that are not included in the AWS PCI DSS assessment that can still be used to meet PCI DSS controls.

## AWS Shared Responsibility Model

Security and compliance is a shared responsibility between AWS and the customer. The AWS Shared Responsibility Model helps relieve the customer's operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the service operates.

The following figure provides an overview of the shared responsibility model. The line of responsibility may vary depending upon the implemented AWS service. For AWS managed services, AWS assumes more of the customer's operational responsibility.

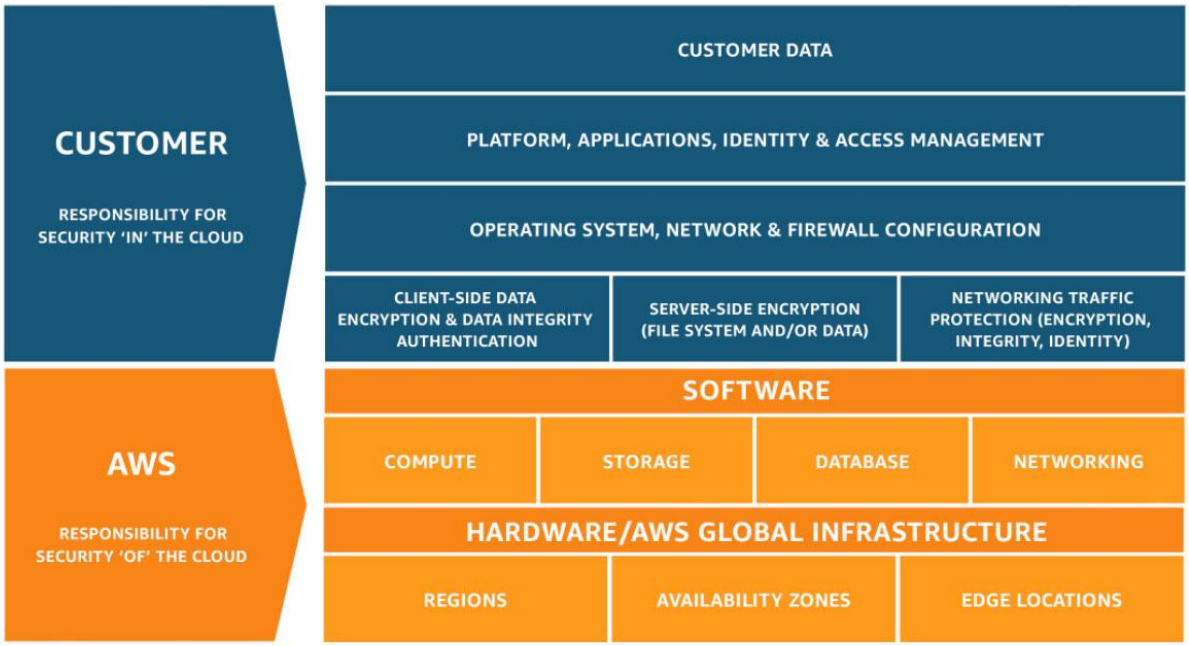


*Figure 1 Shared Responsibility Model*

AWS is responsible for the security and compliance **of** the Cloud, the infrastructure that runs all of the services offered in the AWS Cloud. Cloud security at AWS is the highest priority. AWS customers benefit from a data center and network architecture that are built to meet the requirements of the most security-sensitive organizations. This infrastructure consists of the hardware, software, networking, and facilities that run AWS Cloud services.

Customers are responsible for the security and compliance **in** the Cloud, which consists of customer-configured systems and services provisioned on the AWS. For PCI DSS compliance, customer responsibility is all system components that includes AWS resources included in or connected to their CDE. For abstracted services, such as Amazon S3 or Amazon DynamoDB, responsibility includes customer-configurable controls such as access controls, log settings, and encryption settings.

Depending upon the Amazon service, the level of responsibility depends upon the selected implementation. Amazon ECS is a good example that allows customers to choose a serverless deployment of containers with AWS Fargate, or run containers on Amazon EC2 infrastructure that is accessible by the customer. With AWS Fargate, customers are abstracted from the underlying host and are not responsible for updating or patching the host system, whereas an Amazon ECS on Amazon EC2 deployment

requires customers to take on a greater role of responsibility, such as controlling access and applying security patches.

If a customer can control a service parameter, they are responsible for ensuring it is configured to meet PCI DSS requirements.

## PCI DSS Scope Determination and Validation

It is critical is to understand the complete flow of cardholder data (CHD) within the environment. The CHD flow determines the applicability of the PCI DSS, defines the boundaries and components of a cardholder data environment (CDE), and therefore the scope of a PCI DSS assessment. Accurate determination of the PCI DSS scope is key to defining the security posture and ultimately a successful assessment. Customers must have a procedure for scope determination that assures its completeness and detects changes or deviations from the scope.

The ephemeral nature of containerized applications provides additional complexities when auditing configurations. As a result, customers need to maintain an awareness of all container configuration parameters to ensure compliance requirements are addressed throughout all phases of a container lifecycle.

# Securing an Amazon ECS on Amazon EC2 Deployment

The upcoming sections provide guidance on key topics to consider when architecting a container-based environment for PCI DSS compliance. They are broken down into the following categories.

- Network segmentation
- Host and container image hardening
- Encrypting data at rest and in transit
- Restricting user access
- Event logging
- Vulnerability scanning and penetration testing

Each section provides an overview of the requirements along with best practices recommendations to follow. The guidance is not all inclusive as each customer's

environment is unique. Collectively, the following recommendations provide a defense-in-depth approach to securing a container-based environment.

## Network Segmentation

Controls within requirement 1 of the PCI DSS call for installing and maintaining a firewall to protect cardholder data and require that systems are to be protected from unauthorized access. This requirement means inbound and outbound access must be restricted to only approved ports and services. Although the use of network segmentation is not a PCI DSS requirement, its usage is heavily encouraged so as to reduce the scope of a customer's environment.

Container networking utilizes a software bridge where all of the containers on a host can communicate through a virtual network provided by the container management application. Each container receives a private IP address, but communications with endpoints outside of the instance share the primary elastic network interface of the host on which they are running.

Security groups may be applied to an elastic network interface, but the shared intercommunications among the containers could expose sensitive containerized workloads to out-of-scope services. Such exposure likely increases the scope and impact compliance.

Within AWS, security groups act as a virtual firewall and provide stateful inspection. Security groups can be used to restrict communications by IP address, port, and protocol. It is important to note that, by default, security groups allow all outbound communications. As a result, outbound connection rules must be configured to meet PCI DSS compliance.

Optimally, containerized workloads should be grouped based on data sensitivity levels to more easily facilitate network segmentation. Inter-container communications among containers should be restricted through the use of micro-segmentation. User-defined bridges may be used to restrict inter-container communications within a host, but all communications leaving the host traverse a shared network interface.

Granular control of task communications can also be performed through the use of task networking with `awsvpc` network mode where Amazon ECS tasks are assigned to an elastic network interface. With security groups applied to the interface, network traffic restrictions can be established for a given task. When using `awsvpc` networking mode, the number of tasks run on an instance is limited to the number Amazon EC2 instance network interfaces.

To summarize, consider the following options when working to isolate containerized application communications:

- Isolate containers on separate hosts based on sensitivity of services.
- Implement micro-segmentation to limit inter-container communications.
- Use `awsvpc` networking mode to apply security groups to specific container tasks.

The following section provides suggestions on hardening container-based workloads.

## Host and Image Hardening

Requirement 2 of the PCI DSS emphasizes the need to disable support for vendor supplied defaults for system passwords and other security parameters. Amazon container services like Amazon ECS are run on container-optimized Amazon Machine Images (AMI). These operating systems do not contain additional libraries that are not essential for container deployments, and as a result, help to minimize attack vectors.

Customers are still responsible for maintaining compliance of all configurations and functions at the operating system, network, and application layers. Operating systems should be routinely patched through the use of AWS Systems Manager, whereas non-essential services and libraries should be disabled or removed. Configuration standards should be established that are consistent with industry-accepted system hardening guidelines, such as the Center for Internet Security (CIS) Benchmarks for EC2 instance types. Additional AWS secure configuration standards support is available on the AWS Cloud Security Learning page.

Container builds should be limited to only required resources and adopt a model of microservices where a container provides one primary function. Software architects should ensure that images do not rely on outdated software libraries and applications that may contain known vulnerabilities. A best practice is to rebuild container images in the container registry on a periodic basis to ensure the latest application versions are in use. The use of vulnerable libraries may introduce avenues of attack that are often overlooked.

When managing containers, containers should be immutable and not patched in-place. Customers should create trusted base container images that have been assessed and confirmed to use patched libraries and applications. Use a trusted registry to secure container images, such as Amazon Elastic Container Registry (Amazon ECR). Amazon

ECR provides image scanning based upon the Common Vulnerabilities and Exposures (CVEs) database and can identify common software vulnerabilities.

Customers are responsible for ensuring that their Amazon EC2 instances run appropriate antivirus and file integrity monitoring software. Many container vendors provide solutions optimized for container usage to address these requirements.

## Data Protection

The PCI DSS controls within requirements 3 and 4 are focused on the need to protect sensitive data while at rest and in transit. AWS provides a number of PCI DSS compliant services and features to assist with these compliance efforts.

Workloads that contain sensitive data, such as cardholder data, should secure all storage of data. Data should be stored on secure file stores or databases and not on the underlying container host. System architects should be mindful of volume mounts and sharing of data between containers, such as host file systems and temporary storage.

Sensitive data and environment variables, such as database connection strings that are contained within container build files, must be secured. AWS Secrets Manager and AWS Systems Manager Parameter Store are two services that can be used to secure sensitive data within container build files. AWS Systems Manager Parameter Store provides secure, hierarchical storage of data with no servers to manage. Granular access and audit controls can be established to ensure appropriate restrictions are in place to support meeting compliance requirements. Data stored within AWS Systems Manager Parameter can be encrypted using AWS Key Management Service (AWS KMS).

Similar to AWS Systems Manager Parameter Store, data secured within AWS Secrets Manager also leverages AWS KMS. AWS Secrets Manager provides additional capabilities that includes random password generation and automatic password rotation. AWS KMS is a PCI DSS compliant service that is integrated with many AWS platform services. Users can create and manage cryptographic key material as well as control who can access and use the encryption keys.

For data in transit, sensitive information must be encrypted during transmission over open, public networks. Customers are responsible for configuring strong cryptography and security controls. AWS provides multiple services, such as Amazon API Gateway and Application Load Balancer, that support the use of Transport Layer Security (TLS). Policies can be applied to the services to enforce support of only TLS 1.1 or greater.

Amazon API Gateway and Application Load Balancer also support use of the integrated AWS WAF (web application firewall) to secure communications at the application layer. The AWS WAF protects applications and APIs against common web exploits like those identified within the OWASP Top 10.

# User Access

The controls within requirements 7 and 8 of the PCI DSS are focused on restricting access to authorized personnel and ensuring appropriate access controls are in place. Access to resources should embrace a least privilege model where access is on a need-to-know basis. User access to containers and the underlying host should be authenticated with strong authentication requirements that align with the PCI DSS.

Container images should be run with non-privileged user accounts. For instance, by default, container build files that do not contain defined user credentials run as root. This means that a compromised container service may extend root privileges to an attacker who may use the elevated access to further exploit the underlying host.

Unlike AWS Fargate, where no host access is available, Amazon ECS on Amazon EC2 deployments provide the option to enable secure shell (SSH) access for underlying system management. Consider disabling the use of SSH and instead leverage AWS Systems Manager's Run Command. With Run Command, there are no SSH keys to manage, it is non-interactive, and all invoked operations are auditable within AWS CloudTrail.

In an effort to create and establish secure container images, restrict all access to container images. Container deployments should use a private container registry that restricts access and write permissions, such as Amazon ECR, which integrates with Identity and Access Management (IAM) for access controls. Amazon ECR is a scalable container repository that provides secure storage and transmission of container images. The simplified workflow and integration of Amazon ECR with AWS services also reduces the need for excessively providing credentialed access to container hosts.

# Tracking and Monitoring Access

## Event Logging

The core controls within requirement 10 of the PCI DSS emphasize the need to utilize event logging mechanisms to track, monitor, and alert on potentially anomalous activities.

Leverage AWS event log services to establish event log monitoring at the network, host, and container. Enable VPC Flow Logs to capture network traffic that details packet information, such as the protocol, port, source address, and destination address information. Monitor container hosts to ensure health, efficiency, and availability by ensuring the Amazon CloudWatch agent or Amazon Kinesis Agent are enabled and configured.

Enable event logging capabilities within the containerized applications to capture application and container event log data. Use CloudWatch as a single pane of glass to monitor and alert on all captured event log activity. Store the captured event data securely within encrypted Amazon Simple Storage Service (Amazon S3) buckets to support meeting retention requirements. Amazon Athena and Amazon CloudWatch Logs Insights can be used to query and analyze audit trail logs saved to Amazon S3 from VPC Flow Logs, AWS CloudTrail, and Amazon CloudWatch.

## Network Intrusion Detection

Controls within requirement 11 of the PCI DSS specifies the use of intrusion-detection and/or intrusion-prevention techniques to detect and/or prevent intrusions into the network. The standard requires monitoring of all traffic at the perimeter and critical points of the CDE. With most on-premises environments, the requirements are typically addressed by using Intrusion Detection System (IDS)/Intrusion Prevention System (IPS) appliances. A similar approach can be used within AWS.

When considering containerized environments, inspection of network traffic can be done at the network layer outside of the container host and within the container management software's virtual container network.

There are several options that can be considered for inspection of network data outside of the container host on AWS. Amazon GuardDuty provides threat detection through anomaly detection, machine learning, and threat intelligence of events across AWS data sources, which includes AWS CloudTrail, DNS queries, and Amazon VPC Flow Logs.

When considering a traditional IDS/IPS solution, Amazon VPC Traffic Mirroring can be configured to route a copy of all network communications to a virtual appliance running on one or more Amazon EC2 instances.

Another common solution is to create a transit network architecture that use IP routing to ensure that all network traffic crosses a single network. This architecture allows the use of a virtual IDS/IPS device from the AWS Marketplace to inspect all traffic transiting between networks. It is possible to also use a VPC Gateway to route all traffic to on-

premises IDS/IPS infrastructure. Lastly, host-based IDS or IPS solutions can also be used to inspect traffic as the traffic is delivered to an Amazon EC2 instance.

Inspection of inter-container communications on the virtual container network is another viable option. There are vendors within the AWS Marketplace that provide IDS container solutions, which mostly use a sidecar container to monitor and alert on unusual traffic patterns. Agent-based solutions are also available that use machine learning to detect anomalous communication patterns among the containers.

The security measures put into place depend heavily upon the architecture of the environment. Traffic detection at the network layer requires advanced planning of container deployments and traffic patterns.

## Vulnerability Scanning

The PCI DSS requires that organizations regularly test systems and processes to identify vulnerabilities and remediate such findings in a timely manner. While vulnerability scanning should be performed on a quarterly basis as well as after any significant changes to the environment, including vulnerability scanning into the build process makes it easier to more frequently perform the scans. Similarly, penetration testing is to be performed on an annual basis and after any significant environment changes. Penetration testing of AWS resources is allowed for permitted services. The AWS support policy for penetration testing should be consulted for further details. For those service providers that are using network segmentation, these providers are required to test the effectiveness of segmentation controls every six months or after any changes to the segmentation controls.

The scope of the assessment activities includes the CDE and ancillary systems used in support of the CDE. See the PCI DSS Information Supplement: Penetration Testing Guidance for scope and methodology guidance when performing penetration testing.

Depending upon a customer's environment, the test requirements may apply to on-premises, cloud resources, and containerized environments. When deploying Amazon ECS on Amazon EC2 instances, customers must perform vulnerability scanning of the underlying host. Per the PCI DSS, customers are responsible for establishing a process to identify security vulnerabilities and assigning a risk ranking to newly discovered security vulnerabilities. Amazon Inspector is a security assessment tool that helps identify vulnerabilities and prioritizes findings by level of severity. Integration of Amazon Inspector within the DevOps process provides for assessment automation to proactively identify vulnerabilities.

Customers should also look to use container specific scanning tools to scan container images for vulnerabilities. Container scanning identifies non-compliant code, vulnerable libraries, and potentially exposed secrets. Amazon ECR image scanning helps to identify software vulnerabilities within your container images based on the Common Vulnerabilities and Exposures (CVEs) database. Security vendors within the AWS Marketplace also provide solutions capable of scanning systems, containers, and applications.

When performing internal and external penetration testing, assessment activities should be done at both a network and application layer and should target the underlying host and containerized applications. Patch container hosts to address vulnerabilities and update container images to mitigate identified container vulnerabilities. Create hardened container images and securely store them within private container registries, such as Amazon ECR.

# Conclusion

AWS provides multiple services to support customer's containerized workloads, and customers can configure the services to best meet their data processing needs. Because of this flexibility, organizations must maintain an awareness of all compliance requirements throughout the lifecycle of their container deployments. Methods of security mitigation outlined within this whitepaper can help customers to address PCI DSS compliance requirements for their containerized workloads.

# Contributors

The following individuals and organizations contributed to this document:

- Tim Sills, Sr. Assurance Consultant, AWS Security Assurance Services

# Further Reading

For additional information, see:

- Payment Card Industry Data Security Standard (PCI DSS) 3.2.1 on AWS Compliance Guide
- Architecting for PCI DSS Scoping and Segmentation on AWS
- AWS Quick Start Standardized Architecture for PCI DSS Compliance on AWS

- AWS Security Documentation
- AWS Cloud Security
- AWS Security Whitepapers, Technical Guides, and Reference Material
- AWS Well-Architected Framework Security Pillar

# Document Revisions

| Date | Description |
|---|---|
| **July 2020** | First publication |