aqua

Compliance Guide:

# Achieving PCI-DSS Compliance for Containers

Last Updated: July 2017

## Introduction

The PCI-DSS standard was set up by the credit card industry to ensure that organization who handle credit card information – store it, process it, manage it – adhere to strict security measures to protect this data, and can quickly ascertain and react to potential breaches. When a new technology such as containers is introduced, it changes how applications are developed, delivered, and managed – and it may invariably impact the PCI compliance status of organizations that employ it. Organizations are required to adapt their security countermeasures and processes to ensure they remain compliant.

## How Containers Affect PCI-DSS Compliance

Containers introduce dramatic changes into application development. They often drive an increase in the use of open-source components, and they also accelerate the pace of software development, challenging established security check-points to keep up. This new process may also introduce vulnerabilities, and evade vetting processes based on existing version and configuration management.

Additionally, the container stack is radically new. Containers run on a shared kernel, which limits the level of isolation, and they require dynamic networking – both of which make it harder to have visibility and control over the runtime environment. This might also render existing, non-container-native countermeasures – such as IDS/IPS and firewalls – ineffective, in that they have no visibility into the activities of running containers.

Key areas where containers may impact PCI compliance include (though are not exclusive to):

- Vulnerability management
- Data protection
- Network security
- Threat analysis and mitigation
- User access control, segregation of duties
- Real-time visibility and event audit trails

## Overview: How Aqua Helps Address PCI-DSS Requirements

The following table offers a quick summary of how the Aqua Container Security Platform helps to comply with PCI requirements. Each feature is described in detail later in this document – click the link in the right column to jump to the description.

| PCI-DSS Requirement | Aqua Feature Addressing the Requirement | |
|---|---|---|
| **1.1.2** Current network diagram that identifies all connections between the cardholder data environment and other networks, including any wireless networks | Automated mapping of container network connections | ⬇ |
| **1.1.4** Requirements for a firewall at each Internet connection and between any demilitarized zone (DMZ) and the internal network zone | Container-level firewall policies and nano-segmentation | ⬇ |
| **1.2** Build firewall and router configurations that restrict connections between untrusted networks and any system components in the cardholder data environment | Container-level firewall policies and nano-segmentation | ⬇ |
| **1.2.1** Restrict inbound and outbound traffic to that which is necessary for the cardholder data environment, and specifically deny all other traffic | Container profile network rules, and container firewall rules | ⬇ |
| **1.3.4** Do not allow unauthorized outbound traffic from the cardholder data environment to the Internet | Container profile network rules, based on labeling PCI-related services | ⬇ |
| **2.2** Develop configuration standards for all system components. Assure that these standards address all known security vulnerabilities and are consistent with industry-accepted system hardening standards. | Enforcement of CIS Benchmark and best practices, image vulnerability scanning | ⬇ |
| **2.2.2** Enable only necessary services, protocols, daemons, etc., as required for the function of the system. | Behavioral container security profiles that whitelist legitimate activities and enforce least functionality | ⬇ |
| **2.2.3** Implement additional security features for any required services, protocols, or daemons that are considered to be insecure | Global security controls that enforce configuration, container-specific threat mitigation capabilities | ⬇ |
| **2.4** Maintain an inventory of system components that are in scope for PCI DSS. | Automated registry scanning, views of all running containers, and compliance reports | ⬇ |

| PCI-DSS Requirement | Aqua Feature Addressing the Requirement | |
|---|---|---|
| **3.6.2** Secure cryptographic key distribution | Secrets management that securely injects secrets into running containers | ⬇ |
| **6.1** Establish a process to identify security vulnerabilities, using reputable outside sources for security vulnerability information, and assign a risk ranking (for example, as “high,” “medium,” or “low”) to newly discovered security vulnerabilities. | Image scanning for vulnerabilities, vulnerability severity ranking, and CVSS scores. | ⬇ |
| **6.4.1** Separate development/test environments from production environments, and enforce the separation with access controls. | Label-based management combined with RBAC | ⬇ |
| **6.4.2** Separation of duties between development/test and production environments | Label-based management combined with RBAC | ⬇ |
| **6.5.6** All “high risk” vulnerabilities identified in the vulnerability identification process (as defined in PCI DSS Requirement 6.1). | Image scanning and severity ranking, and continuous monitoring of running containers for newly discovered vulnerabilities | ⬇ |
| **7.1** Limit access to system components and cardholder data to only those individuals whose job requires such access:<br>**7.1.2** Restrict access to privileged user IDs to least privileges necessary to perform job responsibilities.<br>**7.1.3** Assign access based on individual personnel’s job classification and function<br>**7.2.2** Assignment of privileges to individuals based on job classification and function.<br>**8.1.1** Assign all users a unique ID before allowing them to access system components or cardholder data | Container-level RBAC, Aqua system roles and integration with Active Directory / LDAP | ⬇ |
| **8.2.1** Using strong cryptography, render all authentication credentials (such as passwords/phrases) unreadable during transmission and storage on all system components | Secrets management that securely injects credentials into running container | ⬇ |
| **10.1** Implement audit trails to link all access to system components to each individual user | Container-level events are logged to establish an audit trail, including named user traceability., integration with 3rd party SIEM / analytics | ⬇ |

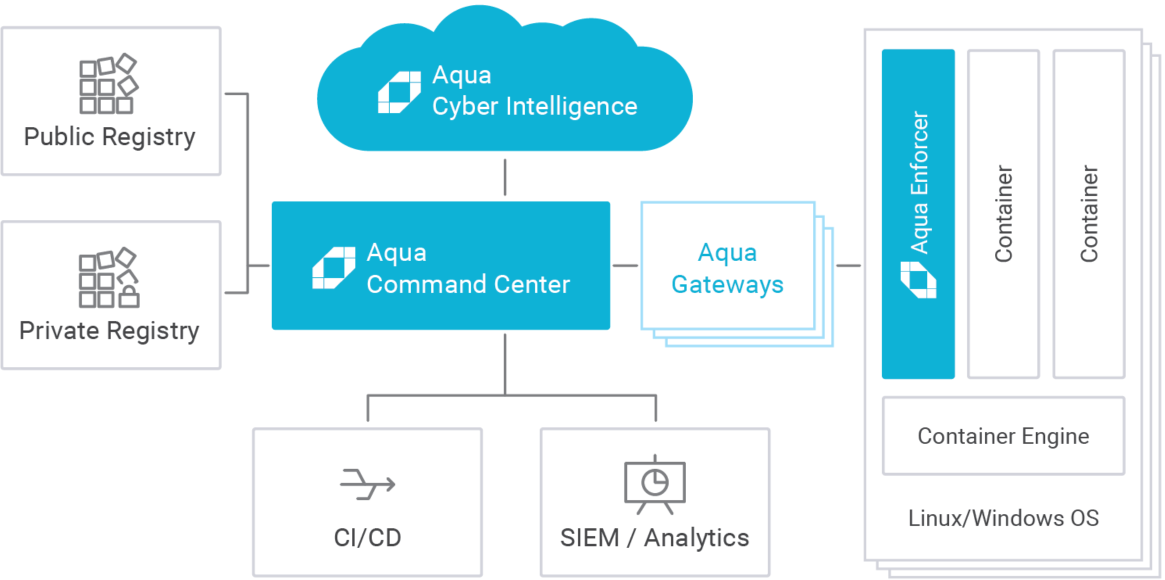
| PCI-DSS Requirement | Aqua Feature Addressing the Requirement | |
|---|---|---|
| **10.2** Implement automated audit trails for all system components to reconstruct the following events:<br>**10.2.2** All actions taken by any individual with root or administrative privileges | Container-level events are logged to establish an audit trail, including named user traceability. All events can be exported to 3rd party SIEM/analytics tools such as Splunk, ArcSight, LogRhythm, etc. | ⬇ |
| **10.3** Record at least the following audit trail entries for all system components for each event:<br>**10.3.1** User identification<br>**10.3.2** Type of event<br>**10.3.3** Date and time<br>**10.3.4** Success or failure indication<br>**10.3.5** Origination of event<br>**10.3.6** Identity or name of affected data, system component, or resource | Container-level events are logged to establish an audit trail, including named user traceability. All events can be exported to 3rd party SIEM/analytics tools such as Splunk, ArcSight, LogRhythm, etc. | ⬇ |
| **11.5** Deploy a change-detection mechanism (for example, file-integrity monitoring tools) to alert personnel to unauthorized modification (including changes, additions, and deletions) of critical system files, configuration files, or content files; and configure the software to perform critical file comparisons at least weekly. | Tracking and monitoring of container images, as well as allowed executables within images. | ⬇ |

# Aqua Container Security Platform: Quick Overview

Aqua’s platform is a container-native, full lifecycle solution for securing container-based applications.

The Aqua CSP comprises three main components:

1. **Aqua Command Center:** A central management component, can be deployed on multiple instances for high availability. It provides policy management, image scanning, image lifecycle controls, monitoring, and reporting. It integrates with image registries for image scanning, with CI/CD tools for security testing as part of the build, and with SIEM/analytics to output audit and alert data. The Command Center exposes full API access as well as a management console UI.
2. **Aqua Enforcer:** The Aqua Enforcer is itself deployed as a container, in charge of monitoring and controlling container operational actions on every protected node. It has visibility into the activity of other containers on the node, and employs multiple methods to stop specific activities that do not comply with policy.
   The Enforcer communicates the event stream back to the Command Center, which in turn publishes the policy out to the Enforcers. All communications are encrypted using SSH and use mutual authentication protocols to prevent spoofing.
3. **Aqua Cyber Intelligence:** Cloud-based service that supplies container vulnerability and threat intelligence to Aqua deployments. The service relies on multiple public and proprietary source, and provides “virtual patching” for several attack vectors, all of which are updated regularly.

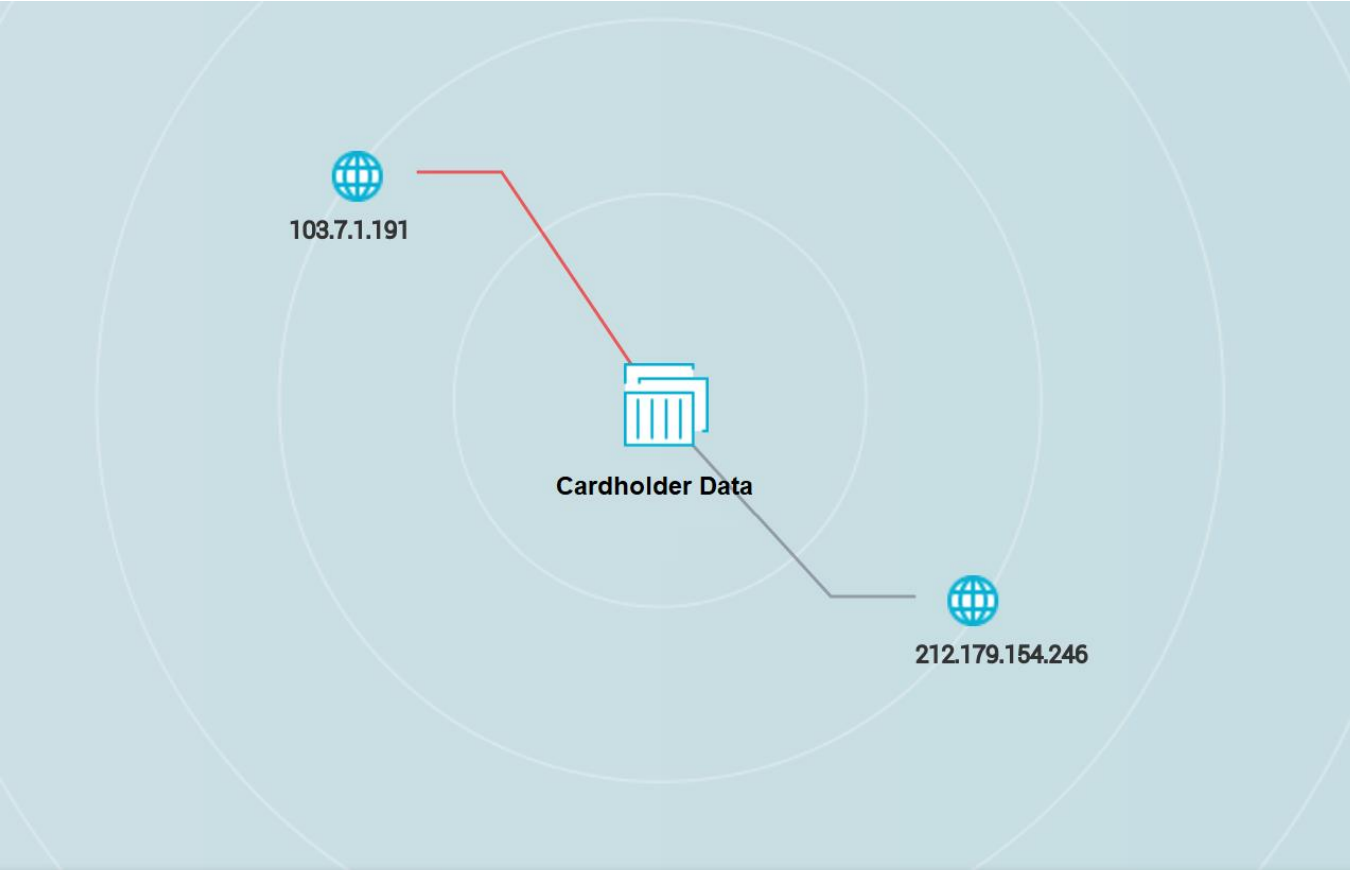


# Using the Aqua CSP for PCI-DSS Compliance

Following are detailed explanations and examples of how the Aqua Container Security Platform addresses PCI requirements.

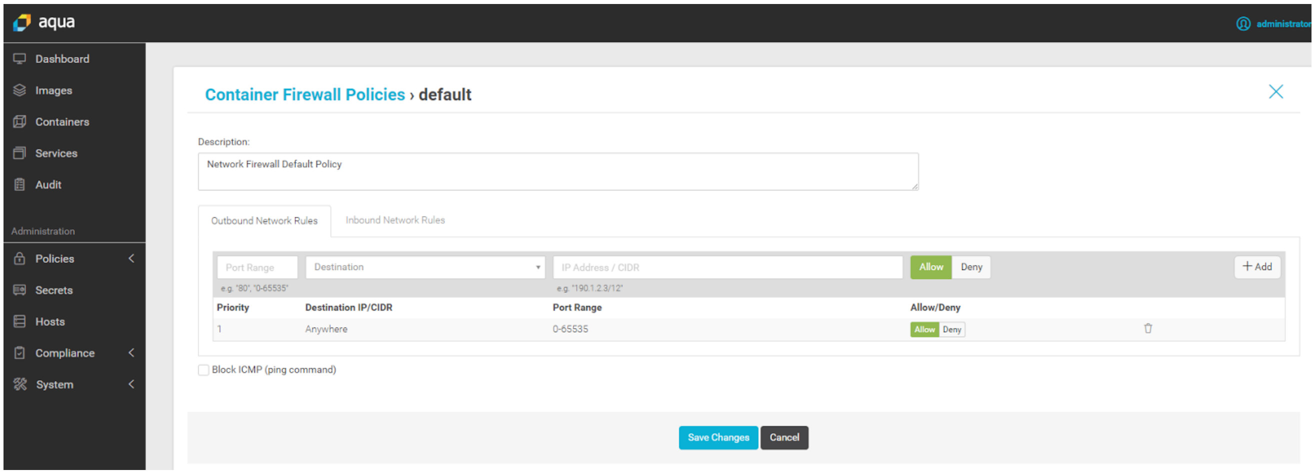
## **1.1.2** Current network diagram that identifies all connections between the cardholder data environment and other networks, including any wireless networks

Aqua's nano-segmentation feature start with automated monitoring and visualization of container networks across application services, regardless of actual network infrastructure used, and including networking within and across hosts. Admins can view a visual representation of the network topology and associated connections of a service. The map enables admins to capture monitored connections and use them as a base for the container firewall policy. Any blocked connections will appear with a red line on the map.



### **1.1.4** Requirements for a firewall at each Internet connection and between any demilitarized zone (DMZ) and the internal network zone

With Aqua’s Container Firewall (Nano Segmentation), admins can limit the network connectivity between containers by applying a firewall-like concept for the container environment. Aqua admins can deny inbound and/or outbound communications from/to containers in the network section when creating a new runtime profile for container.



Container Firewall Policies › default

Description:
Network Firewall Default Policy

e.g. "80", "0-65535"
e.g. "190.1.2.3/12"

| Priority | Destination IP/CIDR | Port Range | Allow/Deny |
|---|---|---|---|
| 1 | Anywhere | 0-65535 | Allow Deny |

□ Block ICMP (ping command)

### **1.2** Build firewall and router configurations that restrict connections between untrusted networks and any system components in the cardholder data environment

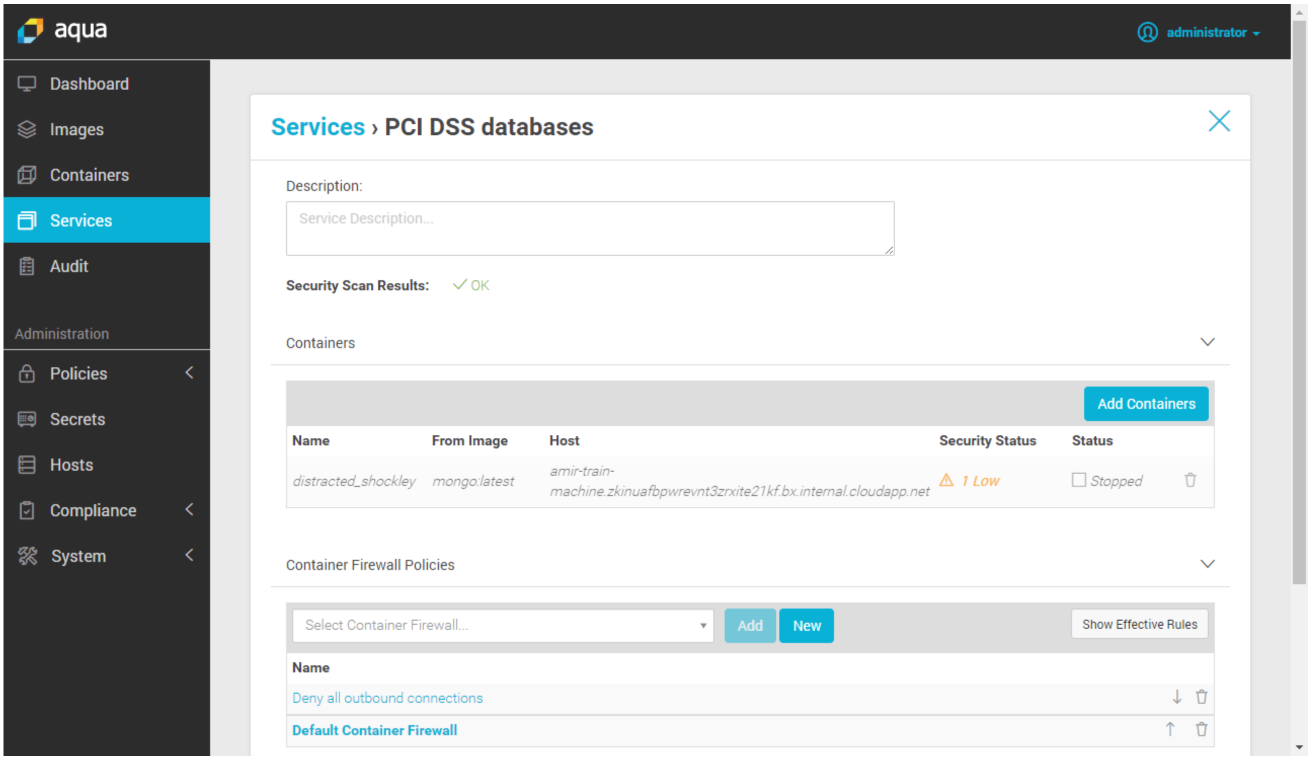
With Aqua’s Container Firewall (Nano Segmentation) admins can limit the network connectivity between services by applying a firewall-like concept for the container environment. This capability allows creating network boundaries across services, where admins can control which networks are accessible for each service.
Aqua’s label-based management allows to labels groups of containers as PCI-sensitive, either within the Aqua console or by inheriting security group definitions from orchestrators.

### **1.2.1** Restrict inbound and outbound traffic to that which is necessary for the cardholder data environment, and specifically deny all other traffic

Aqua provides several ways to limit container network traffic. In addition to nano-segmentation (explained above in 1.1.2, 1.1.4, and 1.2), the security profiles of containers can categorically deny outbound or inbound connections. For example, a database container will typically not require outbound connectivity, so it will be automatically denied.

## **1.3.4** Do not allow unauthorized outbound traffic from the cardholder data environment to the Internet

Aqua's container firewall functionality can also be used to create global rules that would prevent containers tagged with a certain label, for example – PCI-DSS compliance, from having outbound or inbound connections, or only permit them to access specific IP addresses.



Services › PCI DSS databases

Description: Service Description...

Security Scan Results: ✓ OK

Containers

| Name | From Image | Host | Security Status | Status |
|---|---|---|---|---|
| distracted_shockley | mongo:latest | amir-train-machine.zkinuafbpwrevnt3zrxite21kf.bx.internal.cloudapp.net | ⚠ 1 Low | □ Stopped |

Container Firewall Policies

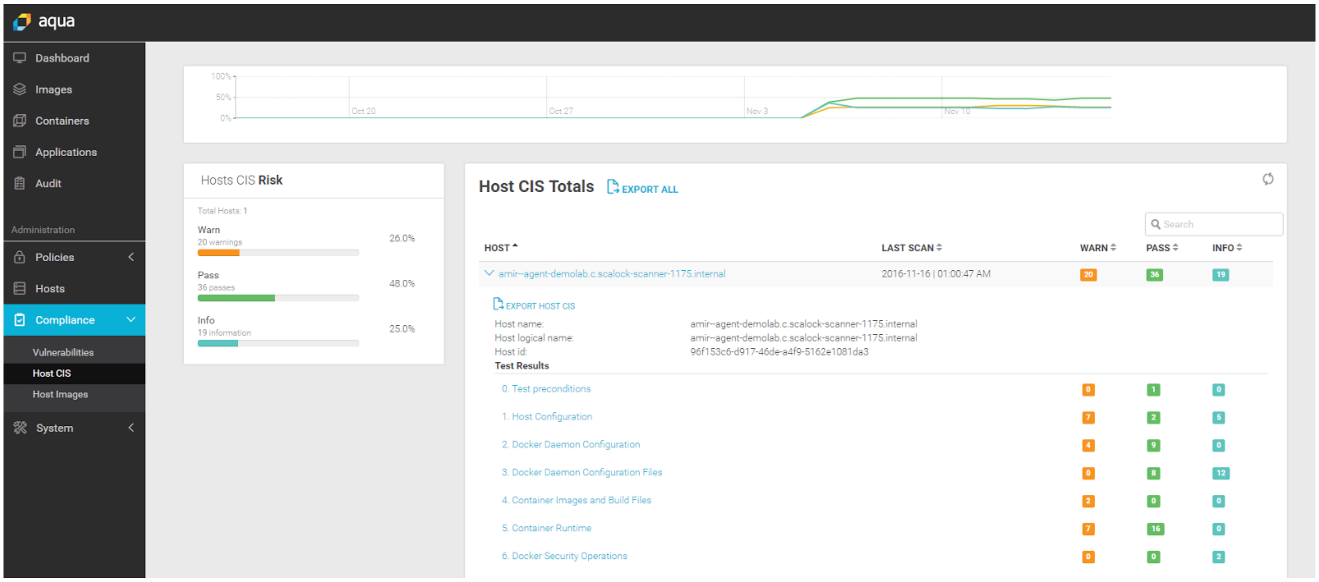
| Name |
|---|
| Deny all outbound connections |
| Default Container Firewall |

## **2.2** Develop configuration standards for all system components. Assure that these standards address all known security vulnerabilities and are consistent with industry-accepted system hardening standards

Aqua supports the CIS Docker benchmark for host hardening, compliance reports per host, and a comprehensive image vulnerability report. The Host CIS screen provides information regarding compliance with the CIS Benchmark, which has best practices for Docker engine configuration, as well as container runtimes, host configurations, etc.

Aqua administrator can build policies based on the guidance provided in CIS Docker benchmark and review compliance with Aqua Host CIS reports.

Additionally, Aqua's image scanning provide deep analysis of known vulnerabilities in image, including binaries and packages, supporting multiple programming languages.



Hosts CIS **Risk**

Total Hosts: 1

| | |
|---|---|
| Warn<br>20 warnings | 26.0% |
| Pass<br>36 passes | 48.0% |
| Info<br>19 information | 25.0% |

**Host CIS Totals**

| HOST | LAST SCAN | WARN | PASS | INFO |
|---|---|---|---|---|
| amir--agent-demolab.c.scalock-scanner-1175.internal | 2016-11-16 \| 01:00:47 AM | 20 | 36 | 19 |

Host name: amir--agent-demolab.c.scalock-scanner-1175.internal
Host logical name: amir--agent-demolab.c.scalock-scanner-1175.internal
Host id: 96f153c6-d917-46de-a4f9-5162e1081da3

**Test Results**

| | | | |
|---|---|---|---|
| 0. Test preconditions | 0 | 1 | 0 |
| 1. Host Configuration | 7 | 2 | 5 |
| 2. Docker Daemon Configuration | 4 | 9 | 0 |
| 3. Docker Daemon Configuration Files | 0 | 8 | 12 |
| 4. Container Images and Build Files | 2 | 0 | 0 |
| 5. Container Runtime | 7 | 16 | 0 |
| 6. Docker Security Operations | 0 | 0 | 2 |

## **2.2.2** Enable only necessary services, protocols, daemons, etc., as required for the function of the system

Aqua allows admins to easily enforce least functionality with automatic creation of a runtime profile for containers. Aqua Enforcer profiles the behavior of a given container, it analyzes and reports on the security profile by noting file access, resource usage, network settings, namespace settings, and executable, baselining all the legitimate container activity and using machine learning to create a whitelisting policy. Admins can also manually tweak the profile parameters for specific parameters.
On top of that, With Aqua image assurance policy users can block images that have not been vetted by Aqua (images that were not scanned), meaning that only approved images will be allowed to run.

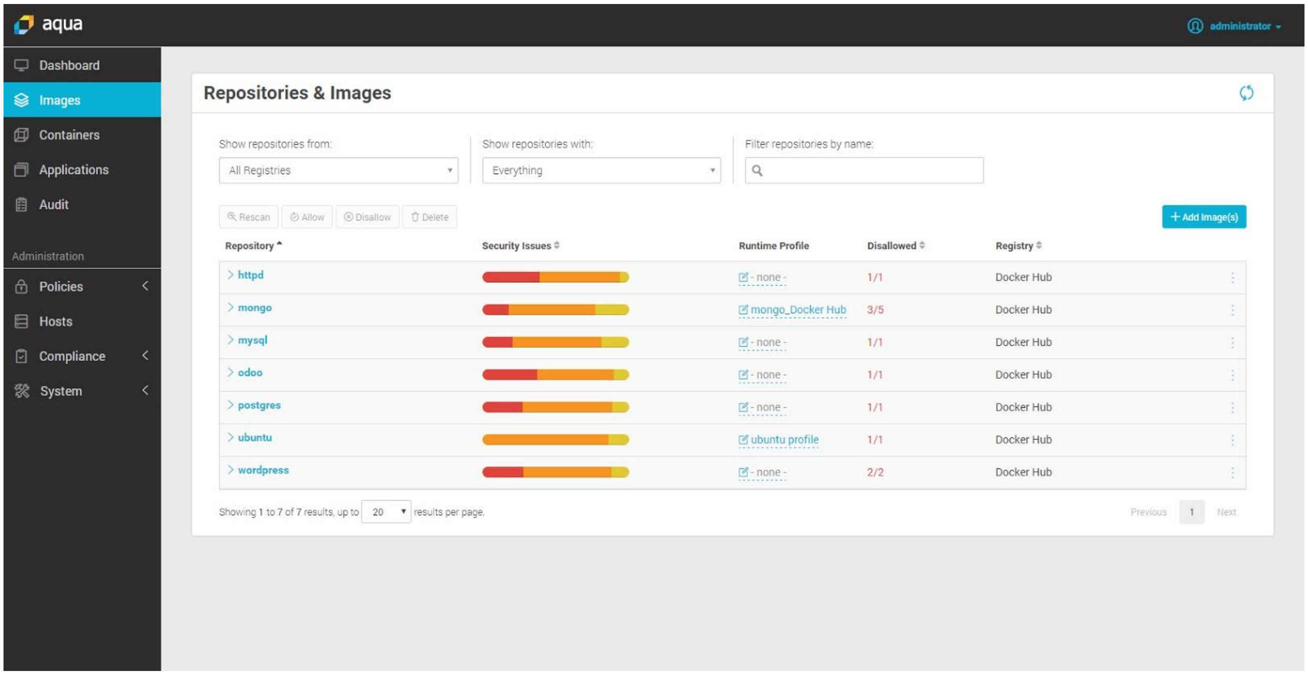
## **2.2.3** Implement additional security features for any required services, protocols, or daemons that are considered to be insecure

Aqua provides numerous settings that can enforce best practices, such as the use of sanctioned base images, prevent containers from running as root, prevent containers from running executable not in the original image, and preventing image drift, i.e. the use of unsanctioned image versions. Additionally, any behavior not allowed by the container behavioral profiles described in 2.2.2 will result in alerts being generated and logged.
Additionally, Aqua's security research team provides threat mitigation protections ("IPS for containers") that block specific behaviors that are indicative of attacks, such as fork bombs and attempts to access malicious IP addresses.

## **2.4** Maintain an inventory of system components that are in scope for PCI DSS

Aqua provides an inventory of containerized applications, covering the different repositories, images, containers, and hosts in the organization. Aqua connects to image registries and enumerate all images stored in them. For each image, it creates an inventory of the installed packages. Aqua will also process all images stored on the hosts, that were not pulled from an image registry, and create a package inventory for every image.

Aqua also provides a view of all running containers and their originating images, including package inventory for every running container, and with Aqua labels users can attach a simple name to any resource to easily create a label based rule for operations or inventory purposes.



Repositories & Images

Show repositories from: All Registries

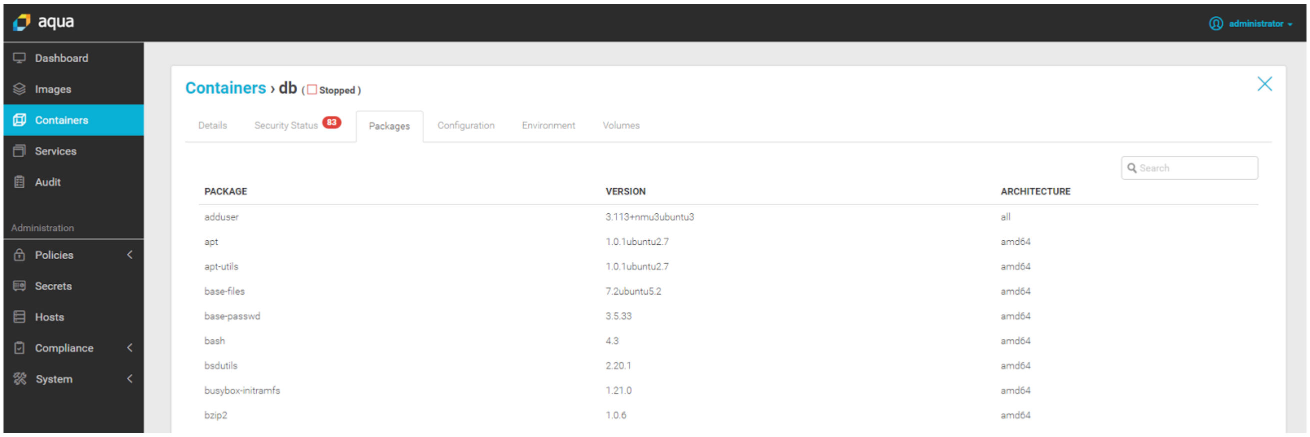
Show repositories with: Everything

Filter repositories by name:

| Repository | Security Issues | Runtime Profile | Disallowed | Registry |
|---|---|---|---|---|
| httpd | | - none - | 1/1 | Docker Hub |
| mongo | | mongo_Docker Hub | 3/5 | Docker Hub |
| mysql | | - none - | 1/1 | Docker Hub |
| odoo | | - none - | 1/1 | Docker Hub |
| postgres | | - none - | 1/1 | Docker Hub |
| ubuntu | | ubuntu profile | 1/1 | Docker Hub |
| wordpress | | - none - | 2/2 | Docker Hub |

Showing 1 to 7 of 7 results, up to 20 results per page.

Previous 1 Next



Containers › db ( Stopped )

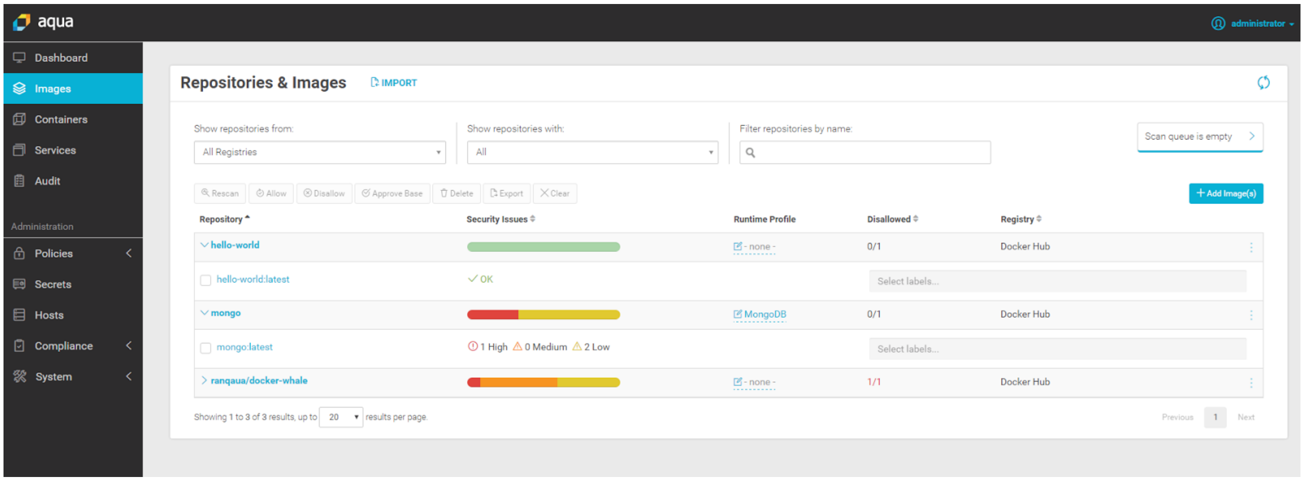
| PACKAGE | VERSION | ARCHITECTURE |
|---|---|---|
| adduser | 3.113+nmu3ubuntu3 | all |
| apt | 1.0.1ubuntu2.7 | amd64 |
| apt-utils | 1.0.1ubuntu2.7 | amd64 |
| base-files | 7.2ubuntu5.2 | amd64 |
| base-passwd | 3.5.33 | amd64 |
| bash | 4.3 | amd64 |
| bsdutils | 2.20.1 | amd64 |
| busybox-initramfs | 1.21.0 | amd64 |
| bzip2 | 1.0.6 | amd64 |

### **3.6.2** Secure cryptographic key distribution

Aqua provides central management and secure distribution of secrets and cryptographic keys into running containers. Admins can define a secret in the Aqua Management console, and assign access control policies that authorize users or groups to run containers that make use of the secret. When a secret is used, its value will be automatically injected into the container, either as environment variable or file. The value of the secret will not be visible outside the container, is encrypted in transit, and does not persist on disk – meaning it disappears once the container stops running.
Aqua integrates with several secret stores, including HashiCorp Vault, Amazon KMS, Azure Vault and CyberArk, to allow organizations to leverage these central stores and extend them for use with containers.

## **6.1** Establish a process to identify security vulnerabilities, using reputable outside sources for security vulnerability information, and assign a risk ranking (for example, as “high,” “medium,” or “low”) to newly discovered security vulnerabilities

Admins can scan images and repositories for known vulnerabilities that could impact the system information assurance and security, each vulnerability is ranked from low to high based on their Common Vulnerability Scoring System (CVSS).
The scanning process is accomplished by integrating the Aqua Command Center with a private image registry.
Each image is being scanned for vulnerabilities both in its OS packages and development language files. All identified vulnerabilities, new and old, are audited and can be automatically mitigated by creating image assurance policies.
Aqua is constantly updated with new vulnerabilities from several sources (commercial, public and proprietary) with Aqua Container Cyber Threat Intelligence feed for continuous improvement and rechecks registries and images for continues monitoring.



Repositories & Images

Show repositories from: All Registries

Show repositories with: All

Filter repositories by name:

Scan queue is empty

| Repository | Security Issues | Runtime Profile | Disallowed | Registry |
|---|---|---|---|---|
| hello-world | | - none - | 0/1 | Docker Hub |
| ☐ hello-world:latest | ✓ OK | | Select labels... | |
| mongo | | MongoDB | 0/1 | Docker Hub |
| ☐ mongo:latest | 1 High 0 Medium 2 Low | | Select labels... | |
| ranqaua/docker-whale | | - none - | 1/1 | Docker Hub |

Showing 1 to 3 of 3 results, up to 20 results per page.

### **6.4.1** Separate development/test environments from production environments, and enforce the separation with access controls

Using Aqua, organizations can easily separate their container development from their production deployments.

Organizations can integrate Aqua into the development pipeline by adding Aqua as a build step in CI/CD tools such as Jenkins, TeamCity and more. This also ensures that developers are not exposed to Aqua's console, as they continue working with their existing tools.

With Aqua images are scanned for vulnerabilities and security best-practices policies are applied. A developed image will be promoted to production only if it passes all the required tests. Aqua admin can also assign labels to images and create security policy that allows only images with specific labels to enter production.

On top of that Aqua provides access control for Docker administrative operations by determining which user can access specific Docker resources.

Additionally, Aqua's labels-based management can be used to label hosts as dev/test/production, and different RBAC and container policies can be applied by these labels, so that specific users can only access resources with specific labels.

### **6.4.2** Separation of duties between development/test and production environments

Aqua provides RBAC for Docker administrative operations by determining which user can access specific Docker resources. For example, with Aqua you can specify that a user with role 'image builder' role can only build images without having actual access for running containers.

By default, when admins deploy Aqua Enforcer on a container host, the Aqua Enforcer applies "owner-based access control". This means that a user who is a container owner (a user who created and started the container) has full container access. By default, other users do not have the same degree of container access. The default "owner-based access control" behavior might be applicable for most common container use-cases. To extend this control and enable users to access containers they do not own, admins can create policies and explicitly assign user permissions.

Aqua also integrates with the organization's identity management systems such as Active Directory / LDAP, to map container users to the organizational user groups, and SAML for single sign-on. This allows even more granular access control and separation of duties.

### **6.5.6** All “high risk” vulnerabilities identified in the vulnerability identification process (as defined in PCI DSS Requirement 6.1)

As stated in section 6.1 - Admins can scan images and repositories for known vulnerabilities that could impact the system information assurance and security, each vulnerability is ranked from low to high based on their Common Vulnerability Scoring System (CVSS). Additionally, all running containers are continuously monitored and matched against the vulnerability database feed in order to flag any container with a newly identified vulnerability.

The Aqua admin can create custom security policies for image assurance, for example blocking any image with high severity vulnerability, or preventing images with an average score higher than X from running.

## **7.1** Limit access to system components and cardholder data to only those individuals whose job requires such access:

### **7.1.2** Restrict access to privileged user IDs to least privileges necessary to perform job responsibilities;

### **7.1.3** Assign access based on individual personnel's job classification and function;

### **7.2.2** Assignment of privileges to individuals based on job classification and function;

### **8.1.1** Assign all users a unique ID before allowing them to access system components or cardholder data

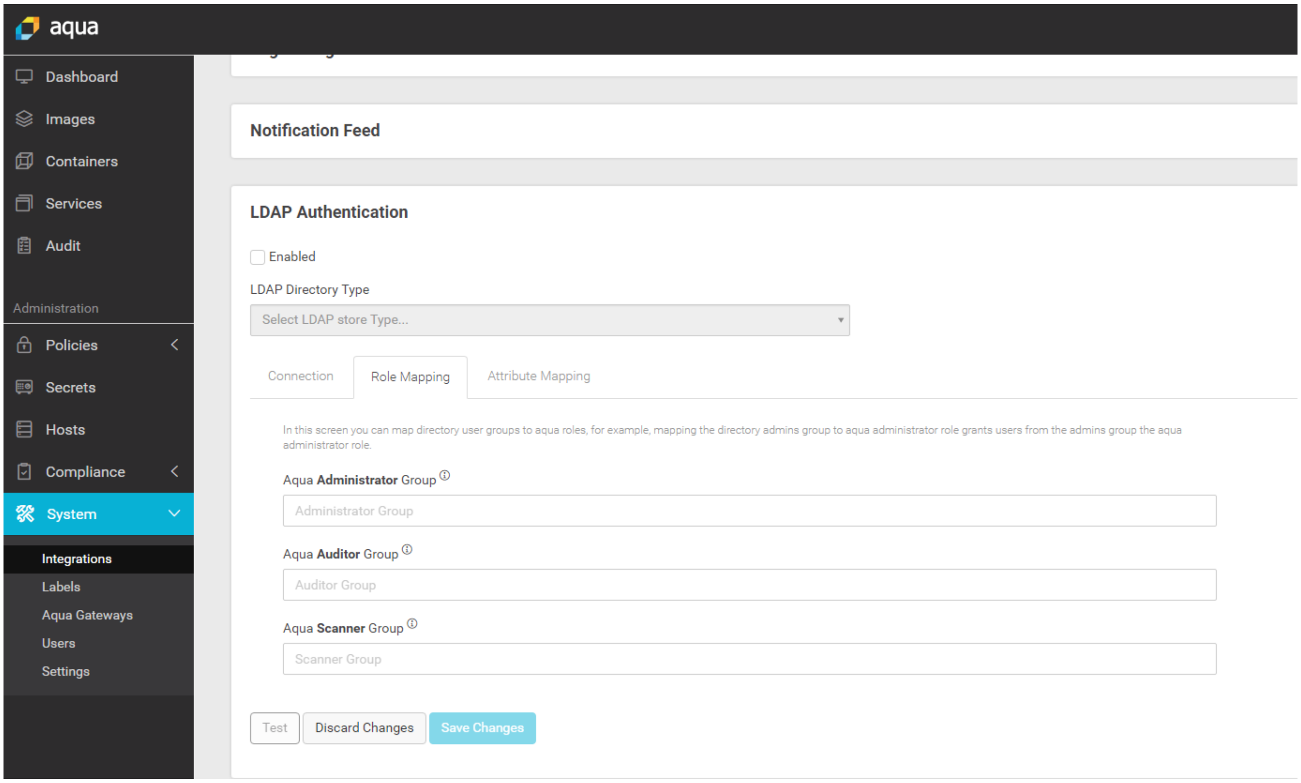
Aqua provides RBAC for administrative container operations by determining which user can access specific container resources.

By default, when admins deploy Aqua Enforcer on a container host, the Aqua Enforcer applies "owner-based access control". This means that a user who is a container owner (a user who created and started the container) has full container access. By default, other users do not have the same degree of container access. The default "owner-based access control" behavior might be applicable for most common container use-cases. To extend or reduce privileges admins can create policies and explicitly assign user permissions.

Aqua can also automatically create an image security profile based on container activity, which Aqua tracks and analyzes. The profiler analyzes and reports on the security profile by noting vital component usage, resources, and network settings. This will create least privileges security runtime profile and Aqua admin can then edit and save it.

To further extend RBAC, admins can use the user access control policies to prevent containers running as root and/or allow container to run with specific users.

Aqua integrates with the organization's identity management systems such as Active Directory / LDAP, to map container users to the organizational user groups, and SAML for single sign-on. This allows even more granular access control and separation of duties.



LDAP Authentication

□ Enabled

LDAP Directory Type

Select LDAP store Type...

In this screen you can map directory user groups to aqua roles, for example, mapping the directory admins group to aqua administrator role grants users from the admins group the aqua administrator role.

Aqua **Administrator** Group ⓘ

Administrator Group

Aqua **Auditor** Group ⓘ

Auditor Group

Aqua **Scanner** Group ⓘ

Scanner Group

## **8.2.1** Using strong cryptography, render all authentication credentials (such as passwords/phrases) unreadable during transmission and storage on all system components

Aqua provides central management and secure distribution of sensitive information like passwords, connection strings or tokens (secrets) into running containers. Admins define a secret in the Aqua Management console, and assign access control policies that authorize users or groups to run containers that make use of the secret. When a secret is used, its value will be automatically injected into the container, either as environment variable or file. The value of the secret will not be visible outside the container, is encrypted in transit, and does not persist on disk. Aqua integrates with several secret stores, this allows Aqua's admin to use secrets that are already defined in the secret store and to distribute values through Aqua's access control system to containers.

**10.1** Implement audit trails to link all access to system components to each individual user;

**10.2** Implement automated audit trails for all system components to reconstruct the following events:

**10.2.2** All actions taken by any individual with root or administrative privileges

All Docker commands are logged by Aqua. Event details include the user, image name, the container name, rule, and the reason for the event severity.

**10.3** Record at least the following audit trail entries for all system components for each event:

**10.3.1** User identification

**10.3.2** Type of event

**10.3.3** Date and time

**10.3.4** Success or failure indication

**10.3.5** Origination of event

**10.3.6** Identity or name of affected data, system component, or resource

Aqua's Audit feature enables admins to view audit events from the Aqua management console. All audit messages generated by Aqua Enforcer are sent to the Aqua Command Center and are available from the console's audit screen. Admins can also configure forwarding of events to external SIEM and analytics tools, like Splunk, ArcSight, SumoLogic and more, this is available under the "Integrations" section in the Aqua management console.
The audit display results appear in table format per time, audit type, and brief description

**11.5** Deploy a change-detection mechanism (for example, file-integrity monitoring tools) to alert personnel to unauthorized modification (including changes, additions, and deletions) of critical system files, configuration files, or content files; and configure the software to perform critical file comparisons at least weekly.

Aqua's image assurance functionality ensures that images cannot be changed without going through the process of scanning and validation. Additionally, executables in images are tracked, and new executables added to a container that were not present in the originating image are blocked.

* * *

Questions? Email us at contact@aquasec.com

Aqua enables enterprises to secure their virtual container environments from development to production, accelerating container adoption and bridging the gap between DevOps and IT security.

The Aqua Container Security Platform provides full visibility into container activity, allowing organizations to detect and prevent suspicious activity and attacks, providing transparent, automated security while helping to enforce policy and simplify regulatory compliance.

For more information, visit www.aquasec.com