A guide to PCI Compliance in Containers and Kubernetes
## Contents

### Introduction
Where does PCI DSS apply? ................................. 4
Containers, Kubernetes and PCI compliance ......................................... 4
PCI DSS Requirements ........................................................................... 5
Feature coverage ....................................................................................... 7

### Requirements and Sysdig Capabilities
Requirement 1: Install and maintain a firewall configuration to protect cardholder data ............... 8
  1.1.2. Current Network diagram .......................................................... 8
  1.1.3. Diagram data flow .................................................................. 10
  1.1.4. Establishing a firewall and a DMZ ........................................... 11
  1.1.5. Description groups, roles, responsibilities management network components .......... 11
  1.1.6.b. Identify insecure services, protocols, and ports allowed ....................... 12
  1.2. Restrict connections to untrusted networks ........................................ 16
  1.3. Examine firewall and router configurations ......................................... 17

Requirement 2  Do not use vendor-supplied defaults for system passwords and other security parameters .............................................................................. 19
  2.2 Configuration standards: CIS, ISO, SANS, NIST .................................... 19
  2.2.a System configuration standards ......................................................... 19
  2.2.1 One function per server isolation (containers) .................................... 24
  2.2.2 Enable only necessary services, protocols, daemons ............................. 28
  2.4 Inventory of system components .......................................................... 31
  2.6 Shared hosting isolation protection ......................................................... 32

Requirement 4: Encrypt transmission of cardholder data across open, public networks ............ 37
  4.0 Strong cryptography for sensitive data .................................................. 37

Requirement 6: Develop and maintain secure systems and applications .................................. 40
6.1 Identify security vulnerabilities with ranking

6.2 Install vendor security patches

6.3 Develop following PCI DSS and best practices

6.4 Separation development / test / production

6.5 Inspect flaws like SQL injection and others

6.6 Review public-facing web at least annually and after a change

Requirement 7: Restrict access to cardholder data by business need to know

7.1 Restrict access to privileged user IDs

7.1.3 Assign access based on an individual personnel’s job classification and function

7.2 Assign privileges to individuals based on job classification and function

7.2.3 Default deny-all setting

Requirement 10: Track and monitor all access to network resources and cardholder data

10.1 Implement audit trails to link access to each individual user

10.2 Implement automatic audit trails to reconstruct events

10.2.1 Of all individual user accesses to cardholder data

10.2.2 Of all actions taken by any individual with root or administrative privileges

10.2.5 Use and change to identification and auth mechanisms

10.2.6 Init, stop or pausing logs

10.2.7 Creation/Deletion system-level objects

10.3 Record audit trail for events

10.5 Logs can not be changed

10.6 Daily review of all security events

Requirement 11: Regularly test security systems and processes

11.4 Network intrusion detection/prevention to monitor traffic

11.5.1 Respond to alerts of change detection
Introduction

Credit card companies previously had to enforce their own version of compliance for all vendors that stored, processed or transmitted cardholder data. Then, in the early 2000s, representatives from American Express, JCB, Visa, Discover and Mastercard combined to form the Payment Card Industry Security Standards Council (PCI SSC). This council created PCI DSS (Payment Card Industry Data Security Service) and released the first set of standards in 2006.

The most recent version of the standard, PCI DSS 3.2.1, came out in May 2018. The standards serve as guidelines and are the starting point for an organization to build their compliance strategy. As applications and technologies change, organizations are required to adapt their compliance strategies to meet the guidelines set by PCI DSS.

Where does PCI DSS apply?

"The PCI DSS security requirements apply to all system components included in or connected to the cardholder data environment."

The cardholder data environment (CDE) is comprised of people, processes and technologies that store, process or transmit cardholder data or sensitive authentication data. “System components” include network devices, servers, computing devices and applications.” Many of these applications are now running directly on containers.

Containers, Kubernetes and PCI compliance

Containers have been adopted faster than any previous enterprise technology, and for good reason. They’re portable, provide better security through isolation, and allow application teams to develop better services faster. However, the quick rise in adoption is a pace that’s hard to match on the compliance side. A great example of this is the Glossary of Terms, Abbreviations, and Acronyms of V3.2 PCI-DSS guidelines. There are definitions for Virtual Machines, Hypervisors and everything you’d need to know for the VM world. However, there are no mentions of Docker, containers, orchestration, Kubernetes, or the (kernel) which becomes even more important when deploying containers.

Containers allow greater degrees of segmentation and isolation across your environment, but their density and ephemeral nature will greatly increase the number of network connections, in addition to making it harder to track what’s connected to what and where. This increase in density will also increase the number of entities that need to be audited and checked for vulnerabilities.
PCI DSS Requirements

PCI DSS 3.2.1 defines 12 requirements categories and 5 appendices:

- **Requirement 1: Install and maintain a firewall configuration to protect cardholder data.** Firewalls are devices that control computer traffic allowed between an entity’s networks (internal) and untrusted networks (external), as well as traffic into and out of more sensitive areas within an entity’s internal trusted networks. The cardholder data environment is an example of a more sensitive area within an entity’s trusted network.

- **Requirement 2: Do not use vendor-supplied defaults for system passwords and other security parameters.** Malicious individuals (external and internal to an entity) often use vendor default passwords and other vendor default settings to compromise systems. These passwords and settings are well known by hacker communities and are easily determined via public information.

- **Requirement 3: Protect stored cardholder data.** Protection methods such as encryption, truncation, masking and hashing are critical components of cardholder data protection. If an intruder circumvents other security controls and gains access to encrypted data without the proper cryptographic keys, the data is unreadable and unusable to that person. Other effective methods of protecting stored data should also be considered as potential risk mitigation opportunities. For example, methods for minimizing risk include not storing cardholder data unless absolutely necessary, truncating cardholder data if full PAN is not needed, and not sending unprotected PANs using end-user messaging technologies, such as e-mail and instant messaging.

- **Requirement 4: Encrypt transmission of cardholder data across open, public networks.** Sensitive information must be encrypted during transmission over networks that are easily accessed by malicious individuals. Misconfigured wireless networks and vulnerabilities in legacy encryption and authentication protocols continue to be targets of malicious individuals who exploit these vulnerabilities to gain privileged access to cardholder data environments.

- **Requirement 5: Protect all systems against malware and regularly update anti-virus software or programs.** Malicious software, commonly referred to as “malware”—including viruses, worms and Trojans—enters the network during many business-approved activities, including employee e-mail and use of the Internet, mobile computers and storage devices, resulting in the exploitation of system vulnerabilities. Anti-virus software must be used on all systems commonly affected by malware to protect systems from current and evolving malicious software threats. Additional anti-malware solutions may be considered as a supplement to the anti-virus software; however, such additional solutions do not replace the need for anti-virus software to be in place.


- **Requirement 6: Develop and maintain secure systems and applications.**
  Unscrupulous individuals use security vulnerabilities to gain privileged access to systems. Many of these vulnerabilities are fixed by vendor-provided security patches, which must be installed by the entities that manage the systems. All systems must have all appropriate software patches to protect against the exploitation and compromise of cardholder data by malicious individuals and malicious software.

- **Requirement 7: Restrict access to cardholder data by business need to know.**
  To ensure critical data can only be accessed by authorized personnel, systems and processes must be in place to limit access based on need to know and according to job responsibilities.

- **Requirement 8: Identify and authenticate access to system components.**
  Assigning a unique identification (ID) to each person with access ensures that every individual is uniquely accountable for their actions. When such accountability is in place, actions taken on critical data and systems are performed by, and can be traced to, known and authorized users and processes.

- **Requirement 9: Restrict physical access to cardholder data.**
  Any physical access to data or systems that house cardholder data provides the opportunity for individuals to access devices or data and to remove systems or hardcopies, and should be appropriately restricted. For the purposes of Requirement 9, "onsite personnel" refers to full-time and part-time employees, temporary employees, contractors and consultants who are physically present on the entity’s premises. A "visitor" refers to a vendor, guest of any onsite personnel, service workers or anyone who needs to enter the facility for a short duration, usually not more than one day. "Media" refers to all paper and electronic media containing cardholder data.

- **Requirement 10: Track and monitor all access to network resources and cardholder data.**
  Logging mechanisms and the ability to track user activities are critical in preventing, detecting or minimizing the impact of a data compromise. The presence of logs in all environments allows thorough tracking, alerting and analysis when something does go wrong. Determining the cause of a compromise is very difficult, if not impossible, without system activity logs.

- **Requirement 11: Regularly test security systems and processes.**
  Vulnerabilities are being discovered continually by malicious individuals and researchers, and being introduced by new software. System components, processes and custom software should be tested frequently to ensure security controls continue to reflect a changing environment.

- **Requirement 12: Maintain a policy that addresses information security for all personnel.**
  A strong security policy sets the security tone for the whole entity and informs personnel what is expected of them. All personnel should be aware of the sensitivity of data and their responsibilities for protecting it. For the purposes of Requirement 12, "personnel" refers to full-time and part-time employees, temporary employees, contractors and consultants who are “resident” on the entity’s site or otherwise have access to the cardholder data environment.

- **Appendix A1: Additional PCI DSS Requirements for Shared Hosting Providers.**
• **Appendix A2**: Additional PCI DSS Requirements for Entities using SSL/Early TLS for Card-Present POS POI Terminal Connections.

• **Appendix A3**: Designated Entities Supplemental Validation (DESV). This Appendix applies only to entities designated by a payment brand(s) or acquirer as requiring additional validation of existing PCI DSS requirements.

• **Appendix B**: Compensating Controls.
  Compensating controls may be considered for most PCI DSS requirements when an entity cannot meet a requirement explicitly as stated, due to legitimate technical or documented business constraints, but has sufficiently mitigated the risk associated with the requirement through implementation of other, or compensating, controls.

• **Appendix C**: Compensating Controls Worksheet.

**Feature coverage**

In this guide we’ll cover PCI compliance related to:

- Network Security
- Data Protection
- Auditing
- User Access Control
- Incident Response & Recovery
- Forensics
- Vulnerability Management

For each specific requirement we’ll cover the guidelines, how to address the requirement for container environments, and how Sysdig can help.
Requirements and Sysdig Capabilities

Requirement 1:
Install and maintain a firewall configuration to protect cardholder data

Firewalls are devices that control computer traffic allowed between an entity’s networks (internal) and untrusted networks (external), as well as traffic into and out of more sensitive areas within an entity’s internal trusted networks. The cardholder data environment is an example of a more sensitive area within an entity’s trusted network.

1.1.2. Current Network diagram

Requirement
Current network diagram that identifies all connections between the cardholder data environment and other networks, including any wireless networks.

Guidelines
Network diagrams describe how networks are configured and identify the location of all network devices. Without current network diagrams, devices could be overlooked and be unknowingly left out of the security controls implemented for PCI DSS, and thus be vulnerable to compromise.

Container Compliance Approach
Your API service is no longer just a collection of a couple nodes, it’s distributed across tens or hundreds of nodes and thousands of containers with other services running on them as well. Keeping track of who is talking to who, and why, is much harder with these distributed containerized services.

Sysdig Capabilities
Sysdig provides automatic discovery of containers and Kubernetes nodes and services with a real-time topology map showing all containers, hosts and processes in both CDE and non-CDE environments. Sysdig monitors all connections in real-time and will discover any new connections to or from containers immediately.
Sysdig will also let you view policies protecting your network and other services based on the physical or logical scoping that is applied to that policy. This makes it much easier to keep track of what policies apply to different areas of your PCI compliance strategy.

<table>
<thead>
<tr>
<th>Runtime Policies</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>K8s activity</strong></td>
</tr>
<tr>
<td>Entire Infrastructure</td>
</tr>
<tr>
<td>Updated 11 days ago</td>
</tr>
<tr>
<td>33 rules</td>
</tr>
<tr>
<td><strong>Malicious Python library jellyfish activities prevention</strong></td>
</tr>
<tr>
<td>kubernetes.pod.name in (&quot;failureservice:76409b916-76409b916&quot;)</td>
</tr>
<tr>
<td>Updated a minute ago</td>
</tr>
<tr>
<td>4 rules</td>
</tr>
<tr>
<td><strong>Suspicious Container Activity</strong></td>
</tr>
<tr>
<td>container.id != &quot;*&quot;</td>
</tr>
<tr>
<td>Updated a minute ago</td>
</tr>
<tr>
<td>9 rules</td>
</tr>
<tr>
<td><strong>Disallowed Container Activity</strong></td>
</tr>
<tr>
<td>container.id != &quot;*&quot;</td>
</tr>
<tr>
<td>Updated a few seconds ago</td>
</tr>
<tr>
<td>1 rules</td>
</tr>
<tr>
<td><strong>User Management Changes</strong></td>
</tr>
<tr>
<td>Entire Infrastructure</td>
</tr>
<tr>
<td>Updated 2 months ago</td>
</tr>
<tr>
<td>1 rules</td>
</tr>
<tr>
<td><strong>Suspicious Network Activity</strong></td>
</tr>
<tr>
<td>Entire Infrastructure</td>
</tr>
<tr>
<td>Updated 2 months ago</td>
</tr>
<tr>
<td>6 rules</td>
</tr>
<tr>
<td><strong>Access Cryptomining Network</strong></td>
</tr>
<tr>
<td>Entire Infrastructure</td>
</tr>
<tr>
<td>Updated 2 months ago</td>
</tr>
<tr>
<td>2 rules</td>
</tr>
<tr>
<td><strong>All K8s Activity</strong></td>
</tr>
<tr>
<td>Entire Infrastructure</td>
</tr>
<tr>
<td>Updated 2 months ago</td>
</tr>
<tr>
<td>1 rules</td>
</tr>
<tr>
<td><strong>All K8s User Modifications</strong></td>
</tr>
<tr>
<td>kubernetes.namespace.name in (&quot;microservices&quot;)</td>
</tr>
<tr>
<td>Updated a few seconds ago</td>
</tr>
<tr>
<td>6 rules</td>
</tr>
</tbody>
</table>

The Runtime Policies list shows a switch indicating which policies are enabled, and under their name, the scope definition specifying where they are being enforced.
1.1.3. Diagram data flow

**Requirement**
Current diagram that shows all cardholder data flows across systems and networks.

**Guidelines**
Teams need to examine data flow diagrams to visualize all cardholder data flows across systems and networks.

**Sysdig Capabilities**
Sysdig automatically discovers real-time network connections between containers and services. Teams can also alert on specific anomalous flows as CDE and non-CDE based on container and Kubernetes metadata/labels.
1.1.4. Establishing a firewall and a DMZ

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

**Guidelines**
Teams can use Kubernetes network policies to restrict inbound and outbound traffic from the cluster.

**Sysdig Capabilities**
Sysdig applies Kubernetes-native microsegmentation to restrict traffic. It uses Kubernetes metadata and application context to define least privilege network policies in Kubernetes.

---

1.1.5. Description groups, roles, responsibilities management network components

**Requirement**
Description of groups, roles and responsibilities for management of network components.

**Guidelines**
Teams need to verify that firewall and router configuration standards include a description of groups, roles and responsibilities for management of network components.
Sysdig Capabilities
Sysdig provides service-based access control called Sysdig Teams to manage groups, roles and responsibilities for PCI containerized environments. LDAP support in the Sysdig software (on-prem version) platform allows user authentication using credentials in a customer’s own directory server.

1.1.6.b. Identify insecure services, protocols, and ports allowed

Requirement
1.1.6.b Identify insecure services, protocols and ports allowed, and verify that security features are documented for each service.

Guidelines
Compromises often happen due to unused or insecure service and ports, since these often have known vulnerabilities and many organizations don’t patch vulnerabilities for the services, protocols and ports they don’t use (even though the vulnerabilities are still present). By clearly defining and documenting the services, protocols and ports that are necessary for business, organizations can ensure that all other services, protocols and ports are disabled or removed.

Container Compliance Approach
Documenting the ports that a database server typically uses is easy. The challenge comes when that host has a load balancer, an application server, and a database, because Kubernetes or some other orchestrator has scheduled them on the same host. Each container will have their own ports exposed to meet their needs, and your team needs to make sure there aren’t any incorrectly exposed ports.
**Prevention**

Sysdig can prevent images from being built or deployed based on the ports that are exposed on that container. Easily choose to whitelist or blacklist ports for an image and evaluate if those are exposed as a step in your CI/CD evaluation.
Monitoring
Sysdig can show what ports a host, container, deployment or any logical service is using, and provide metrics about requests bytes, etc.
**Detection**

After getting visibility into the standard port behavior of a container or a service, you can easily create a policy to detect unexpected inbound/outbound behavior or control what TCP/UDP ports can be opened for listening.

![Runtime Policies](image)

- **Rule Type**: Network Rule
- **Name**: Allow inbound HTTPS connection
- **Description**: Allow inbound TCP connections using port 443
- **Inbound Connection**: Allow
- **Outbound Connection**: Deny
- **TCP**: If Matching: 443
- **UDP**: If Matching: Port numbers...
- **Tags**: PCI x
1.2. Restrict connections to untrusted networks

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

**Guidelines**
Teams can use Kubernetes network policies to restrict inbound and outbound traffic from the cluster.

**Sysdig Capabilities**
Sysdig applies Kubernetes-native microsegmentation to restrict traffic. It uses Kubernetes metadata and application context to define least privilege network policies in Kubernetes.
1.3. Examine firewall and router configurations

**Requirement**
Prohibit direct public access between the Internet and any system component in the cardholder data environment.

**Guidelines**
Teams can use Kubernetes network policies to restrict ingress or egress traffic between the cluster and internet.

**Sysdig Capabilities**
Sysdig applies Kubernetes-native microsegmentation to restrict traffic. It uses Kubernetes metadata and application context to define least privilege network policies in Kubernetes.


**Requirement 2**

Do not use vendor-supplied defaults for system passwords and other security parameters

Malicious individuals (external and internal to an entity) often use vendor default passwords and other vendor default settings to compromise systems. These passwords and settings are well known by hacker communities and are easily determined via public information.

2.2 Configuration standards: CIS, ISO, SANS, NIST

**Requirement Description**

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. Sources of industry-accepted system hardening standards may include, but are not limited to:

- Center for Internet Security (CIS).
- International Organization for Standardization (ISO).
- SysAdmin Audit Network Security (SANS) Institute.
- National Institute of Standards Technology (NIST).

**Guidelines**

There are known weaknesses with many operating systems, databases and enterprise applications, and there are also known ways to configure these systems to fix security vulnerabilities. To help those that are not security experts, a number of security organizations have established system-hardening guidelines and recommendations, which advise how to correct these weaknesses.

2.2.a System configuration standards

**Requirement Description**

Examine the organization’s system configuration standards for all types of system components and verify that the system configuration standards are consistent with industry accepted hardening standards.
Container Compliance Approach
The CIS has published benchmarks for hardening docker and kubernetes. These can be used to verify secure configuration of the docker hosts, daemon, kubernetes services and other critical components of your container stack.

How Sysdig Can Help
Sysdig allows users to schedule the CIS Docker Benchmark and the CIS Kubernetes Benchmark to be run on areas of their infrastructure. Sysdig returns these results in a report format as well as metrics for dashboarding and alerting.

Dashboards
2.1. Kubelet

2.1.1 Ensure that the `--allow-privileged` argument is set to false (Scored)

2.1.2 Ensure that the `--anonymous-auth` argument is set to false (Scored)

2.1.3 Ensure that the `--authorization-mode` argument is not set to AlwaysAllow (Scored)

2.1.4 Ensure that the `--client-ca-file` argument is set as appropriate (Scored)

Remediation
Edit the `/etc/kubernetes/kubelet` file on each node and set the `KUBELET_ARGS` parameter to `"--protect-kernel-defaults=true"`.

2.1.7 Ensure that the `--protect-kernel-defaults` argument is set to true (Scored)

2.1.8 Ensure that the `--make-iptables-util-chains` argument is set to true (Scored)

2.1.9 Ensure that the `--hostname-override` argument is not set (Scored)

2.1.10 Ensure that the `--event-qps` argument is set to 0 (Scored)

2.1.11 Ensure that the `--tls-cert-file` and `--tls-private-key-file` arguments are set as appropriate (Scored)

2.1.12 Ensure that the `--cadvisor-port` argument is set to 0 (Scored)

2.1.13 Ensure that the `--rotate-certificates` argument is not set to false (Scored)

2.1.14 Ensure that the `RotateKubeletServerCertificate` argument is set to true (Scored)

2.1.15 Ensure that the Kubelet only makes use of Strong Cryptographic Ciphers (Not Scored)
2.2.1 One function per server isolation (containers)

Requirement Description
Implement only one primary function per server to prevent functions that require different security levels from coexisting on the same server. For example, web servers, database servers and DNS should be implemented on separate servers.

Guidelines
If server functions that need different security levels are located on the same server, the security level of the functions with higher security needs would be reduced due to the presence of the lower-security functions. Additionally, the server functions with a lower security level may introduce security weaknesses to other functions on the same server. By considering the security needs of different server functions as part of the system configuration standards and related processes, organizations can ensure that functions requiring different security levels don’t coexist on the same server.

Container Compliance Approach
This is an aspect where containers shine! They allow you to separate processes running from each other while worrying less about the physical infrastructure. They also provide an easier and more cost-effective way to isolate workloads by only running one process per container.

How Sysdig Can Help
Using Sysdig Secure, you can build a policy that detects violations against process isolation inside containers, and then can kill the container if that policy is violated.
Process Isolation Example

Runtime Policies > PCI Compliance Control 2.2.1 - Process Isolation

Name: PCI Compliance Control 2.2.1 - Process Isolation

Description: Implement only one primary function per server to prevent functions that require different security levels from coexisting on the same server.

Enabled: [On]

Severity: Medium

Scope: Custom Scope

Rules

<table>
<thead>
<tr>
<th>Name</th>
<th>Published By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detect process not mysqld</td>
<td>Secure UI</td>
</tr>
</tbody>
</table>

Actions

Containers: [Nothing(notify only)] [Stop] [Pause]

Capture: [Off]

Notification Channels: Select notification channel...
Falco runtime detection rules can also implement detection for inbound or outbound traffic not from authorized server process and port.

```yaml
# Rule to detect inbound or outbound traffic not to authorized server process and port

# Security standards that apply to:
# PCI 2.2.1. One function per server isolation (containers)

- macro: restrict_binary_port
  condition: never_true

- macro: restrict_image
  condition: container.image.repository=nginx # change to image to monitor

- macro: authorized_server_binary
  condition: proc.name="nginx" # change to binary to allow
```
- macro: authorized_server_port

  condition: fd.sport="80" # change to port to allow

- rule: Outbound or inbound traffic not to authorized server process and port

  desc: Only authorized process should receive network traffic.

  condition: >

    restrict_binary_port and

    inbound_outbound and

    container and

    k8s.ns.name in (namespace_scope_remote_nodomain) and

    restrict_image and

    (not authorized_server_binary

    or not authorized_server_port)

  output: >

    Network connection outside authorized port and binary

    (command=%proc.cmdline connection=%fd.name user=%user.name
    container_id=%container.id image=%container.image.repository)

  priority: WARNING

  tags: [network, PCI, PCI_DSS_2.2.1, PCI_DSS_2.2.2]
2.2.2 Enable only necessary services, protocols, daemons

Requirements Definition
Enable only necessary services, protocols, daemons, etc., as required for the function of the system.

Guidelines
As stated in Requirement 1.1.6, there are many protocols that a business may need (or have enabled by default) that are commonly used by malicious individuals to compromise a network. Including this requirement as part of an organization’s configuration standards and related processes ensures that only the necessary services and protocols are enabled.

Container Compliance Approach
Containers offer the opportunity to architect your applications with as much isolation as possible. This means running a single process per container and communicating through standards ports with the same network and file patterns everywhere in your infrastructure.

How Sysdig Can Help
Sysdig will look at all activities in your environment to create a baseline of system behavior. From there, we can auto-generate policies and easily detect if there is some unexpected protocol, daemon, process, etc. running on the container.
TCP IN Ports - top ports
size: 2
443
3306

TCP OUT Ports - top ports
size: 1
80

UDP IN Ports - udp ports
size: 1
53

No data found.

Create Policy From Profiles
Also, we can use Falco rules like the previous one to detect a connection outside designated binary and port, as described at 2.2.1, that also helps in this kind of situation.
2.4 Inventory of system components

Requirement Description
Maintain an inventory of system components that are in scope for PCI DSS.

Guidelines
Maintaining a current list of all system components will enable an organization to accurately and efficiently define the scope of their environment for implementing PCI DSS controls. Without an inventory, some system components could be forgotten and be inadvertently excluded from the organization’s configuration standards.

Container Compliance Approach
Often, containers are deployed with an orchestrator. This means that an individual is no longer in control of what containers are being deployed, where. It also increases the velocity at which containers are introduced into your environment. To maintain strong compliance, you need to have a good understanding of what is running now, as well as what ran in the past.

How Sysdig can help
Sysdig comes with an explore view that will give a user an overall view of all hosts and containers running on their system. They can use this table to slice and dice all system components however they
choose. By using the time controls at the bottom of the table, users can always see what containers were running on specific physical infrastructure at any point in time.

2.6 Shared hosting isolation protection

**Requirement Description**
Shared hosting providers must protect each entity’s hosted environment and cardholder data.

**Guidelines**
This is intended for hosting providers that provide shared hosting environments for multiple clients on the same server. When all data is on the same server and under control of a single environment, the settings on these shared servers are typically not manageable by individual clients. This allows clients to add insecure functions and scripts that impact the security of all other client environments, and thereby make it easy for a malicious individual to compromise one client’s data gaining access to all other clients’ data.

**Container Compliance Approach**
One of the largest benefits of containers is the ability to reduce resource consumption by running multiple workloads on the same physical infrastructure. This has complicated the ability to segment data and provide multi-tenant functionality to users.
How Sysdig Helps
Sysdig’s Teams feature is used to segment access to the performance monitoring data we collect from container environments. Financial trading and hosting customers use this to provide data to their customers without giving them access to their entire environments. This can also be used internally for developers to see how a service is performing without giving them access to data from the underlying infrastructure.

A Falco runtime security rule can detect if a user or binary changes thread namespace.

```yaml
- list: user_known_change_thread_namespace_binaries
  items: []
- macro: user_known_change_thread_namespace_activities
  condition: (never_true)
```

# This list allows for easy additions to the set of commands allowed

# to change thread namespace without having to copy and override the

# entire change thread namespace rule.
- list: network_plugin_binaries
  items: [aws-cni, azure-vnet]

- macro: calico_node
  condition: (container.image.repository endswith calico/node and proc.name=calico-node)

- macro: weaveworks_scope
  condition: (container.image.repository endswith weaveworks/scope and proc.name=scope)

- rule: Change thread namespace
  desc: >
  
  an attempt to change a program/thread's namespace (commonly done
  as a part of creating a container) by calling setns.

  condition: >

  evt.type = setns

  and not proc.name in (docker_binaries, k8s_binaries, lxd_binaries, sysdigcloud_binaries, sysdig, nsenter, calico, oci-umount, network_plugin_binaries)

  and not proc.name in (user_known_change_thread_namespace_binaries)

  and not proc.name startswith "runc"

  and not proc.cmdline startswith "containerd"

  and not proc.pname in (sysdigcloud_binaries)

  and not python_running_sdchecks

  and not java_running_sdjagent

  and not kubelet_running_loopback
and not rancher_agent
and not rancher_network_manager
and not calico_node
and not weaveworks_scope
and not user_known_change_thread_namespace_activities

output: >
Namespace change (setns) by unexpected program (user=%user.name command=%proc.cmdline
parent=%proc.pname %container.info container_id=%container.id
image=%container.image.repository)
priority: NOTICE
tags: [process, PCI, PCI_DSS_6.4.2]

A Falco runtime security rule can detect if inbound network traffic comes from outside the local area network for containers that should be isolated.

# Rule to detect network connection outside local subnet
- macro: enabled_rule_network_only_subnet
  condition: never_true
- list: images_allow_network_outside_subnet
  items: []
- macro: scope_network_only_subnet
  condition: >
not container.image.repository in (images_allow_network_outside_subnet)

- macro: network_local_subnet

  condition: >

  fd.rnet in (rfc_1918_addresses) or
  fd.ip = "0.0.0.0" or
  fd.net = "127.0.0.0/8"

- rule: Network connection outside local subnet

  desc: Scoped images should only receive and send traffic to local subnet

  condition: >

  enabled_rule_network_only_subnet and
  inbound_outbound and
  container and
  not network_local_subnet and
  scope_network_only_subnet

  output: >

  Network connection outside local subnet

  (command=%proc.cmdline connection=%fd.name user=%user.name container_id=%container.id
  image=%container.image.repository namespace=%k8s.ns.name
  fd.rip.name=%fd.rip.name fd.lip.name=%fd.lip.name fd.cip.name=%fd.cip.name
  fd.sip.name=%fd.sip.name)

  priority: WARNING

  tags: [network, PCI, PCI_DSS_6.4.2]
Requirement 4:
Encrypt transmission of cardholder data across open, public networks

Sensitive information must be encrypted during transmission over networks that are easily accessed by malicious individuals. Misconfigured wireless networks and vulnerabilities in legacy encryption and authentication protocols continue to be targets of malicious individuals who exploit these vulnerabilities to gain privileged access to cardholder data environments.

4.0 Strong cryptography for sensitive data

Requirement
Use strong cryptography and security protocols to safeguard sensitive cardholder data during transmission over open, public networks, including the following:

- Only trusted keys and certificates are accepted.
- The protocol in use only supports secure versions or configurations.
- The encryption strength is appropriate for the encryption methodology in use. Encrypt transmission of cardholder data across open, public networks.

Guidelines
The intent of this requirement is that organizations can detect if containerized applications or services are communicating securely.

Sysdig Capabilities
Sysdig can detect unencrypted connections not using SSL/TLS, for example, and automatically trigger an alert.
A guide to PCI Compliance in Containers and Kubernetes
A Falco rule to detect creation of an ingress object in a Kubernetes cluster without TLS certificate.

# Applies to standard:
# PCI 4.0. Strong cryptography for sensitive data

- macro: kactivity
  condition: (kevt and consider_activity_events)

- macro: kcreate
  condition: ka.verb=create

- macro: response_successful
  condition: (ka.response.code startswith 2)

- macro: ingress
  condition: ka.target.resource=ingresses

- macro: ingress_tls
  condition: (jevt.value[/requestObject/spec/tls] exists)

- rule: Ingress Object Without TLS Cert Created
  desc: Detect any attempt to create an ingress without TLS certification
  condition: >
    (kactivity and kcreate and ingress and response_successful and not ingress_tls)
  output: >
    K8s Ingress Without TLS Cert Created (user=%ka.user.name ingresses=%ka.target.name
    namespace=%ka.target.namespace)

  source: k8s_audit

  priority: WARNING

  tags: [k8s, network, PCI, PCI_DSS_4.0]
Requirement 6: Develop and maintain secure systems and applications

6.1 Identify security vulnerabilities with ranking

Requirement Description
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.

Guidelines
The intent of this requirement is that organizations keep up to date with new vulnerabilities that may impact their environment.

Sources for vulnerability information should be trustworthy and often include vendor websites, industry news groups, mailing lists or RSS feeds.

Once an organization identifies a vulnerability that could affect their environment, the risk that the vulnerability poses must be evaluated and ranked. The organization must therefore have a method in place to evaluate vulnerabilities on an ongoing basis and assign risk rankings to those vulnerabilities. This is not achieved by an ASV scan or internal vulnerability scan, rather, this requires a process to actively monitor industry sources for vulnerability information.

Classifying the risks (for example, as “high,” “medium,” or “low”) allows organizations to identify, prioritize and address the highest risk items more quickly, and reduce the likelihood that vulnerabilities posing the greatest risk will be exploited.

Container Compliance Approach
It’s easier to patch vulnerability risks in containers because you can move containerized applications through the CI/CD pipeline quicker than a traditional application. To help prevent vulnerabilities from entering, production organizations should scan images for vulnerabilities as part of the CI/CD process, within a registry, and then monitor production containers for vulnerabilities.
How Sysdig Helps

Easily define policies to fail builds if the image being built contains critical vulnerabilities with a fix:

Identify containers that have failed their security scan to drill in to see find out how to mitigate risk:
View reports to see why an image has failed the scanning evaluation:
6.2. Install vendor security patches

**Requirement Description**

Ensure that all system components and software are protected from known vulnerabilities by installing applicable vendor-supplied security patches. Install critical security patches within one month of release.

**Guidelines**

Application security teams often need to ensure they address any high severity CVE with a fix within 30 days.

**How Sysdig Helps**

With Sysdig Secure, you can help bring traditional patch management processes to containers. Teams can set up policies for vulnerability reporting both in the registry and/or running in a particular namespace, cluster or cloud region. You can then query for specific vulnerabilities by advanced conditions like CVE ID, severity, fix, age or any other criteria.
6.3. Develop following PCI DSS and best practices

**Requirement Description**
Develop internal and external software applications (including web-based administrative access to applications) securely, as follows:

- In accordance with PCI DSS (for example, secure authentication + logging).
- Based on industry standards and/or best practices.
- Incorporate information security throughout the software-development life cycle.

**Guidelines**
Without the inclusion of security during the requirements definition, design, analysis and testing phases of software development, security vulnerabilities can be inadvertently or maliciously introduced into the production environment.

**How Sysdig Helps**
Sysdig has a native Jenkins plugin and can integrate with tools like Bamboo, Gitlab or CircleCI to easily integrate image scanning into the software development process. This scanning can help identify vulnerabilities, exposed ports, out of date packages and other image contents that don’t follow security best practices.

6.4.2 Separation development / test / production

**Requirement Description**
Separation of duties between development/test and production environments.
How Sysdig Helps
Sysdig’s Teams feature can be used to segment access to different container environments such as development/test environments. Sysdig supports policy separation between containerized and Kubernetes environments segmented by development, test and production environments. Environments can be scoped by namespaces, images, host, container, etc.

A Falco rule to disallow Kubernetes users.

```yaml
# Generally only consider audit events once the response has completed
- list: k8s_audit_stages
  items: ["ResponseComplete"]

# Generally exclude users starting with "system:"
- macro: non_system_user
  condition: (not ka.user.name startswith "system:" )

# This macro selects the set of Audit Events used by the below rules.
```
- macro: kevt

  condition: (jevt.value[/stage] in (k8s_audit_stages))

- macro: kevt_started

  condition: (jevt.value[/stage]=ResponseStarted)

# If you wish to restrict activity to a specific set of users, override/append to this list.

# users created by kops are included

- list: allowed_k8s_users

  items: ["minikube", "minikube-user", "kubelet", "kops", "admin", "kube", "kube-proxy"]

- rule: Disallowed K8s User

  desc: Detect any k8s operation by users outside of an allowed set of users.

  condition: kevt and non_system_user and not ka.user.name in (allowed_k8s_users)

  output: >

    K8s Operation performed by user not in allowed list of users
    (user=%ka.user.name target=%ka.target.name/%ka.target.resource verb=%ka.verb
     uri=%ka.uri resp=%ka.response.code)

  priority: WARNING

  source: k8s_audit

  tags: [k8s, PCI, PCI_DSS_6.4.2]
A Falco rule to detect a connection to a container from outside the local network.

```yaml
# Rule to detect network connection outside local subnet
- macro: enabled_rule_network_only_subnet
  condition: never_true
- list: images_allow_network_outside_subnet
  items: []
- macro: scope_network_only_subnet
  condition: >
    not container.image.repository in (images_allow_network_outside_subnet)
- macro: network_local_subnet
  condition: >
    fd.rnet in (rfc_1918_addresses) or
    fd.ip = "0.0.0.0" or
    fd.net = "127.0.0.0/8"
- rule: Network connection outside local subnet
  desc: Scoped images should only receive and send traffic to local subnet
  condition: >
    enabled_rule_network_only_subnet and
    inbound_outbound and
    container and
```
6.5.1 Inspect flaws like SQL injection and others

**Requirement Description**
Injection flaws, particularly SQL injection. Also consider OS Command Injection, LDAP and XPath injection flaws, as well as other injection flaws.

**Guidelines**
Injection flaws, particularly SQL injection, are a commonly used method for compromising applications. Injection occurs when user-supplied data is sent to an interpreter as part of a command or query. The attacker’s hostile data tricks the interpreter into executing unintended commands or changing data. That allows the attacker to strike components inside the network through the application to initiate attacks such as buffer overflows, or to reveal both confidential information and server application functionality.

**How Sysdig Helps**
Sysdig looks for fundamentally malicious behavior coming from systems. This covers standard injections and intrusions, but also more difficult behaviors to classify, including users modifying rpm packages, unexpected behavior from a database, or system binaries having network activity.
A Falco rule to detect that a DB program has spawned a shell process.

- **rule:** DB program spawned process

  **desc:**

  a database-server related program spawned a new process other than itself.

  This shouldn't occur and is a follow on from some SQL injection attacks.

  **condition:**
Database-related program spawned process other than itself
(user=%user.name program=%proc.cmdline parent=%proc.pname container_id=%container.id image=%container.image.repository)

priority: NOTICE

tags: [process, database, mitre_execution, PCI, PCI_DSS_6.5.1]

6.5.6. High-risk vulnerabilities

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

How Sysdig Helps
Sysdig automatically scans running containers for vulnerabilities (CVE) and misconfigurations in a single workflow. High-risk vulnerabilities are flagged based on the CVSS score and can be mapped back to specific applications/namespaces at runtime. These high risk vulnerabilities can be prevented by directly integrating scanning policies into the CI/CD pipeline (ex. Jenkins) or via an admissions controller in Kubernetes.
6.5.8. Improper access control

Improper access control, such as insecure direct object references, failure to restrict URL access, directory traversal and failure to restrict user access to functions.

**Requirement Description**

Examine software-development policies and procedures and interview responsible personnel to verify that improper access control—such as insecure direct object references, failure to restrict URL access and directory traversal—is addressed by coding technique that includes:

- Proper authentication of users.
- Sanitizing input.
- Not exposing internal object references to users.
- User interfaces that do not permit access to unauthorized functions.

**Guidelines:**

A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, database record or key, as a URL or form parameter. Attackers can manipulate those references to access other objects without authorization.
Consistently enforce access control in presentation layer and business logic for all URLs. Frequently, the only way an application protects sensitive functionality is by preventing the display of links or URLs to unauthorized users. Attackers can use this weakness to access and perform unauthorized operations by accessing those URLs directly.

An attacker may be able to enumerate and navigate the directory structure of a website (directory traversal), thus gaining access to unauthorized information as well as gaining further insight into the workings of the site for later exploitation. If user interfaces permit access to unauthorized functions, this access could result in unauthorized individuals gaining access to privileged credentials or cardholder data. Only authorized users should be permitted to access direct object references to sensitive resources. Limiting access to data resources will help prevent cardholder data from being presented to unauthorized resources.

**How Sysdig can help**

A Falco rule to detect an anonymous request to administer the cluster that has not been rejected.

```
# Corresponds to K8s CIS Benchmark, 1.1.1.
-rule: Anonymous Request Allowed
desc: Detect any request made by the anonymous user that was allowed

condition: >

  kevt and ka.user.name=system:anonymous and ka.auth.decision!=reject
  and not health_endpoint

output: >

  Request by anonymous user allowed
  (user=%ka.user.name verb=%ka.verb uri=%ka.uri reason=%ka.auth.reason)

priority: WARNING

source: k8s_audit

tags: [k8s, PCI, PCI_DSS_6.5.8]
```
6.6. Review public-facing web at least annually and after a change

**Requirement Description**
For public-facing web applications, address new threats and vulnerabilities on an ongoing basis and ensure these applications are protected against known attacks by either of the following methods:
Reviewing public-facing web applications via manual or automated application vulnerability security assessment tools or methods, at least annually and after any changes.

**Guidelines**
Public-facing web applications are primary targets for attackers, and poorly coded web applications provide an easy path for attackers to gain access to sensitive data and systems. The requirement for reviewing applications or installing web-application firewalls is intended to reduce the number of compromises on public-facing web applications due to poor coding or application management practices.

Manual or automated vulnerability security assessment tools and methods review and/or test the application for vulnerabilities.

**Container Challenge**
The ephemeral nature of containers creates a need to scan your infrastructure on a more frequent basis than annually. This requirement should be met as soon as a new version of a service is deployed, or the scans should be performed on an ongoing basis.

**How Sysdig Helps**
Sysdig provides ongoing monitoring of the containers that are running in your public and internal environments. Sysdig provides real time alerting if the vulnerability risk status falls outside a threshold defined by the organization.
Requirement 7: Restrict access to cardholder data by business need to know

To ensure critical data can only be accessed by authorized personnel, systems and processes must be in place to limit access based on need to know and according to job responsibilities.

7.1.2. Restrict access to privileged user IDs

Requirement Description
Restrict access to privileged user IDs to the least privileges necessary to perform job responsibilities.

Guidelines
Pod Security Policies are actually a threat prevention mechanism. The security constraints they enforce prevent attacks from spreading across the cluster and block the typical container breakout approaches. PSPs can also enforce fine grained runtime security profiles like AppArmor, SELinux, seccomp or Linux capabilities that provide a subset of the available root privileges to a process, all without having full root access.

How Sysdig Helps
Sysdig analyzes the requirements of the Pod spec in your Deployment definition and creates the least privilege PSP for your application. This controls if you allow privileged pods, users to run as the

![Sysdig Pod Security Policies](image-url)
container, volumes, etc. You can fine tune the PSP and define the namespace against which you will run the simulation to validate prior to deployment.

7.1.3. Assign access based on an individual personnel’s job classification and function

**Requirement Description**
Assign access based on individual personnel’s job classification and function.

**How Sysdig Helps**
Sysdig creates the least privilege PSP for your application that is specific to a particular namespace. For example, you can create a permissive PSP as default, and then create specific permissive PSPs for certain namespaces that are more classified/sensitive parts of the application.
7.2.2. Assign privileges to individuals based on job classification and function

Requirement Description
Assignment of privileges to individuals based on job classification and function.
How Sysdig Helps
Sysdig creates the least privilege PSP for your application that is specific to a particular namespace. For example, you can create a permissive PSP as default, and then create specific permissive PSPs for certain namespaces that are more classified/sensitive parts of the application.
7.2.3. Default deny-all setting

Requirement Description
Default "deny-all" setting.

How Sysdig Helps
Sysdig creates the least privilege PSP for your application that can be specified to be very restrictive and follow a deny-all setting. See above example.
**Requirement 10:**
Track and monitor all access to network resources and cardholder data

Logging mechanisms and the ability to track user activities are critical in preventing, detecting or minimizing the impact of a data compromise. The presence of logs in all environments allows thorough tracking, alerting and analysis when something does go wrong. Determining the cause of a compromise is very difficult, if not impossible, without system activity logs.

10.1. Implement audit trails to link access to each individual user

**Requirement Description**
Implement audit trails to link all access to system components to each individual user.

**Guidelines**
It is critical to have a process or system that links user access to system components accessed. This system generates audit logs and provides the ability to trace back suspicious activity to a specific user.

**Container Challenge**
Because containers get isolation built in from cgroups and namespaces within the linux kernel, it is very difficult to see what’s actually happening inside the container. Also, when a user is doing something inside a container, it all looks like root activity so it’s very hard to trace any individual user to specific activity within a container.

**How Sysdig Helps**
Sysdig sits at the kernel level so it can capture all system activity pre, during and post any security event. Sysdig Secure will correlate system activity, such as user commands, including the command arguments, pid, directory and more, and correlate that with Kubectl user session.
An example of a rule to detect if a user modifies a binary dir that has a payment file somewhere below in the directory.
An example of a rule to detect if a user spawned a shell with an attached terminal

| Security events that are triggered from a user spawning a shell in a container and then reading the sensitive PAN data. |
Policy Event Details

When
2/5/2020 10:51:30.261 am (3 hours ago)

Related Resources
Capture and commands will cover 10 minutes around the time of the event.

VIEW CAPTURES 1
VIEW COMMANDS 5

Severity
● High

Triggered Policy
Suspicious access to customer private data

Triggered Rule Type
File System

Scope
1. host.mac: 42:01:0a:80:0f:d8
2. container.id: 18a7c7a44bef

Host
Hostname: gke-vicente-test-default-pool-924c4c96-ck3v
MAC: 42:01:0a:80:0f:d8

Container
ID: 18a7c7a44bef
Name: k8s_server_paymentservice-6c47498cb4-4j2cf_default_9c65de11-4723-11ea-8185-42010a80009b_0
Image: gcr.io/mateo-burillo-ns/paymentservice@sha256:e169bb70e32ea4f5a8be84748009c70e5eee300c2ec2

Actions
1 CAPTURE RECORDED

Summary
fd.name /customers/paymentinfo
proc.cmdline sh
evt.type open
proc.name sh
Within Activity Audit we can tell that a Kubernetes user exec into a pod, ran some commands (curl, bash, etc.) reads a specific file, then kills the container to wipe evidence.
Sysdig Inspect can give compliance and forensics teams a view of everything that was going on in the environment. The sliders can be used to view a specific window at microsecond granularity and will update the visualizations in all the tiles.
Tiles can be drilled down into to show specific executed commands. In this case, we can see where the user used the `cat` command to read sensitive data with PANs.

Using the I/O functionality to specifically pinpoint the data that was read and have the ability to quickly judge the scope of the issue.

Sysdig Secure allows you to forward audit related events to SIEM systems like Splunk.

A Falco rule to detect a terminal shell in a container.

- **rule**: Terminal shell in container
desc: >
A shell was used as the entrypoint/exec point into a container with an attached terminal

condition: >

spawned_process and container

and shell_procs and proc.tty != 0

and container_entrypoint

output: >
A shell was spawned in a container with an attached terminal
(user=%user.name %container.info shell=%proc.name parent=%proc.pname
cmdline=%proc.cmdline terminal=%proc.tty container_id=%container.id
image=%container.image.repository)

priority: NOTICE

tags: [container, shell, mitre_execution, PCI, PCI_DSS_10.1]

10.2. Implement automatic audit trails to reconstruct events

Requirement Description
Implement automated audit trails for all system components to reconstruct the following events.

Guidelines
Generating audit trails of suspect activities alerts the system administrator, sends data to other
monitoring mechanisms (like intrusion detection systems), and provides a history trail for post-incident
follow-up. Logging of the following events enables an organization to identify and trace potentially
malicious activities.

How Sysdig Helps
By its own definition, Sysdig is the tool to detect and audit metrics and security events for infrastructure
or cloud resources, so its whole functionality is aimed towards this security requirement.
In addition to many of the features already shown, we can also add the Kubernetes event audit in secure that will register all cluster related actions, where you can filter by event priority, timeframe or scope (clusters, namespaces, etc.).

Several Falco rules can help track specific security events we would like to audit.

Falco rule to detect all K8s Audit Events.

- **rule**: All K8s Audit Events

  **desc**: Match all K8s Audit Events

  **condition**: kall
Falco rule to detect creation of a ClusterRole with Wildcard.

- **rule**: ClusterRole With Wildcard Created
  
  **desc**: Detect any attempt to create a Role/ClusterRole with wildcard resources or verbs
  
  **condition**: >

  kevt and (role or clusterrole) and kcreate and
  (ka.req.role.rules.resources intersects ")" or
  ka.req.role.rules.verbs intersects ""))

  **output**: >

  Created Role/ClusterRole with wildcard
  (user=%ka.user.name role=%ka.target.name rules=%ka.req.role.rules)

  **priority**: WARNING

  **source**: k8s_audit

  **tags**: [k8s, PCI, PCI_DSS_10.2]
A Falco rule to detect creation of a ClusterRole with Write Privileges.

- **rule:** ClusterRole With Write Privileges Created
  
  **desc:** > Detect any attempt to create a Role/ClusterRole that can perform write-related actions
  
  **condition:** kevt and (role or clusterrole) and kcreate and writable_verbs
  
  **output:** >

  Created Role/ClusterRole with write privileges
  (user=%ka.user.name role=%ka.target.name rules=%ka.req.role.rules)

  **priority:** NOTICE

  **source:** k8s_audit

  **tags:** [k8s, PCI, PCI_DSS_10.2]

A Falco rule to detect creation of a ClusterRole with Pod Exec.

- **rule:** ClusterRole With Pod Exec Created
  
  **desc:** Detect any attempt to create a Role/ClusterRole that can exec to pods
  
  **condition:** >

  kevt and (role or clusterrole) and
  kcreate and
  ka.req.role.rules.resources intersects ("pods/exec")

  **output:** >
Created Role/ClusterRole with pod exec privileges
(user=%ka.user.name role=%ka.target.name rules=%ka.req.role.rules)

priority: WARNING

source: k8s_audit

tags: [k8s, PCI, PCI_10.2]

10.2.1. Of all individual user accesses to cardholder data

**Requirement Description**
All individual user accesses to cardholder data.

**Guidelines**
Malicious individuals could obtain knowledge of a user account with access to systems in the CDE, or they could create a new, unauthorized account in order to access cardholder data. A record of all individual accesses to cardholder data can identify which accounts may have been compromised or misused.

**Container Challenge**
Tying file access back to a user can often be difficult, especially when the action is taken inside the container. Also, with the ephemeral nature of containers, a container can be started, complete a data exfiltration activity and then be killed in a fraction of seconds.

**How Sysdig Helps**
See example from 10.1
10.2.2. Of all actions taken by any individual with root or administrative privileges

**Requirement Description**
All actions taken by any individual with root or administrative privileges.

**Guidelines**
Accounts with increased privileges, such as the “administrator” or “root” account, have the potential to greatly impact the security or operational functionality of a system. Without a log of the activities performed, an organization is unable to trace any issues resulting from an administrative mistake or misuse of privilege back to the specific action and individual.

**How Sysdig Helps**
Sysdig, by default, will capture every action taken by a user on your hosts and inside your containers. These actions can also be viewed based on any piece of host, container or orchestration metadata to view how commands can trigger lateral movement across your infrastructure.

*Filter user commands to isolate all root (uid=0) commands executed.*
10.2.5 Use and change to identification and auth mechanisms

Requirement Description
Usage of and changes to identification and authentication mechanisms — including, but not limited to, creation of new accounts and elevation of privileges — and all changes, additions or deletions to accounts with root or administrative privileges.

Guidance
Without knowing who was logged on at the time of an incident, it is impossible to identify the accounts that may have been used. Additionally, malicious users may attempt to manipulate the authentication controls with the intent of bypassing them or impersonating a valid account.

How Sysdig Helps
We have default policies that track if a privilege container is launched and can easily create custom policies below to look for behaviors of privilege elevation.

A Falco rule to detect launching a privileged container.

```
- rule: Launch Privileged Container
desc: >
  Detect the initial process started in a privileged container.
  Exceptions are made for known trusted images.
condition: >
  container_started and container
  and container.privileged=true
  and not falco_privileged_containers
  and not user_privileged_containers
output: >
  Privileged container started
  (user=%user.name command=%proc.cmdline %container.info
  image=%container.image.repository:%container.image.tag)
```
Falco rule looking if a container is running in privileged mode, for example, if privileged is being passed with a user running docker exec.

Policy event notification detecting a privileged container started in a pod.
10.2.6. Init, stop or pausing logs

Requirement Description
Initialization, stopping or pausing of the audit logs.

Guidelines
Turning the audit logs off (or pausing them) prior to performing illicit activities is a common practice for malicious users wishing to avoid detection. Initialization of audit logs could indicate that the log function was disabled by a user to hide their actions.
How Sysdig Helps
Sysdig, by default, tracks uptime metrics for all entities we monitor. These could be containers, hosts, kubernetes services, cloud regions, etc. We can alert if any of these services go down or are removed.

Alert if specific containers are down, splunk, Sysdig, etc.
This list auto populates and is easy to modify.
The same can be done for processes as well. In many cases, the auditing is done at the host and also consumes container info.
10.2.7. Creation/Deletion system-level objects

**Requirement Description**
Creation and deletion of system-level objects.

**Guidelines**
Malicious software, such as malware, often creates or replaces system level objects on the target system in order to control a particular function or operation on that system. By logging when system-level objects, such as database tables or stored procedures, are created or deleted, it will be easier to determine whether such modifications were authorized.

**How Sysdig Helps**
Sysdig has default policies to monitor if different system binaries and built-in commands are supplanted.

![Policy Events Image]

**Summary**
File below a known binary directory opened for writing (user=root command=cp /usr/bin/wget /usr/bin/ls file=/usr/bin/ls parent=sh pcmdline=sh gparent=\\<NA\\> container_id=1c24c3c691ba image=gcr.io/mateo-burillo-ns/emailexervice)

*We can see from the event details that a user replaced the “ls” facility with “wget”. This means that users can now use “ls” to pull data from the internet.*
A guide to PCI Compliance in Containers and Kubernetes

This second default policy detects that a known system binary (ls) sent network traffic, which should never happen.

A Falco rule to detect modification to binary directories.

- **rule**: Modify binary dirs

  **desc**: an attempt to modify any file below a set of binary directories.

  **condition**: >
  
  (bin_dir_rename) and modify and not package_mgmt_procs and not exe_running_docker_save
A Falco rule to detect creating a directory in binary directories.

```plaintext
- rule: Mkdir binary dirs
  desc: an attempt to create a directory below a set of binary directories.
  condition: mkdir and bin_dir_mkdir and not package_mgmt_procs
  output: >
    Directory below known binary directory created
    (user=%user.name command=%proc.cmdline directory=%evt.arg.path container_id=%container.id image=%container.image.repository)
  priority: ERROR
  tags: [filesystem, mitre_persistence, PCI, PCI_DSS_10.2.7]
```
10.3 Record audit trail for events

Requirement Description
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

Guidelines
By recording these details for the audit trail events at 10.2, a potential compromise can be quickly identified, and with sufficient detail to know who, what, where, when and how.

How Sysdig Helps

Every user event has a full timestamp, down to the syscall level of everything that occurred.
10.5.5 Logs can not be changed

Requirement Description
Use file-integrity monitoring or change-detection software on logs to ensure that existing log data cannot be changed without generating alerts (although new data being added should not cause an alert).

Guidelines
File-integrity monitoring or change-detection systems check for changes to critical files, and notify when such changes are noted. For file-integrity monitoring purposes, an entity usually monitors files that don’t regularly change, but when changed, indicate a possible compromise.

How Sysdig Helps
All file activity can be easily monitored and all I/O activity can also be inspected with advanced Falco rules.
A Falco rule to detect modifying logs.

- list: log_directories
  
  items: [/var/log, /dev/log]

- list: log_files
  
  items: [syslog, auth.log, secure, kern.log, cron, user.log, dpkg.log, last.log, yum.log, access_log, mysql.log, mysqld.log]

- macro: access_log_files
  
  condition: (fd.directory in (log_directories) or fd.filename in (log_files))

# a placeholder for whitelist log files that could be cleared. Recommend the macro as (fd.name startswith "/var/log/app1*"

- macro: allowed_clear_log_files
  
  condition: (never_true)

- macro: trusted_logging_images
  
  condition: (container.image.repository endswith "splunk/fluentd-hec" or
            container.image.repository endswith "fluent/fluentd-kubernetes-daemonset")

- rule: Clear Log Activities
  
  desc: Detect clearing of critical log files

  condition: >

  open_write and

  access_log_files and

  evt.arg.flags contains "O_TRUNC" and
not trusted_logging_images and
not allowed_clear_log_files

output: >

Log files were tampered
(user=%user.name command=%proc.cmdline file=%fd.name container_id=%container.id
image=%container.image.repository)

priority: WARNING

tags: [file, mitre_defense_evasion, PCI, PCI_DSS_10.5.5]

10.6.1 Daily review of all security events

Requirement Description
Review the following at least daily - All security events.

Guidelines
Daily review of security events—for example, notifications or alerts that identify suspicious or anomalous activities—as well as logs from critical system components, and logs from systems that perform security functions, such as firewalls, IDS/IPS, file-integrity monitoring (FIM) systems, etc. is necessary to identify potential issues. Note that the determination of “security event” will vary for each organization and may include consideration for the type of technology, location and function of the device. Organizations may also wish to maintain a baseline of “normal” traffic to help identify anomalous behavior.

How Sysdig Helps
Sysdig has multiple summaries that analysts can view to get an at-a-glance view of all the events that have happened in their systems.
The Sysdig event overview dashboard shows all the events that occurred over the past day from a severity, host, container and service perspective.
Requirement 11: 
Regularly test security systems and processes.

Vulnerabilities are being discovered continually by malicious individuals and researchers, and being introduced by new software. System components, processes and custom software should be tested frequently to ensure security controls continue to reflect a changing environment.

11.4. Network intrusion detection/prevention to monitor traffic

Requirement Description
Use network intrusion detection systems and/or intrusion prevention systems to monitor all traffic in the cardholder data environment, and alert personnel to suspected compromises.

Guidelines
Use intrusion-detection and/or intrusion-prevention techniques to detect and/or prevent intrusions into the network. Monitor all traffic at the perimeter of the cardholder data environment as well as at critical points in the cardholder data environment, and alert personnel to suspected compromises. Keep all intrusion-detection and prevention engines, baselines and signatures up to date.

How Sysdig Helps
All network activity can be easily monitored and inspected with advanced Falco rules, as we have described in previous sections.

Also, Secure network policy rules can be created to allow or deny connections based on protocol (TCP or UDP), port and direction (inbound or outbound).
11.5.1. Respond to alerts of change detection

Requirement Description
Implement a process to respond to any alerts generated by the change detection solution.

Guidelines
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; also, configure the software to perform critical file comparisons at least weekly.

How Sysdig Helps
All process, file, network, container and system call activity can be easily monitored and subsequently, an alert notification can be generated.
All policy events have actions with notification channels to alert of events detection.
Policy event email notification example.
Find out how the Sysdig Secure DevOps Platform can help you and your teams confidently run cloud-native apps in production. Contact us for additional details about the platform, or to arrange a personalized demo.

www.sysdig.com