Security Standards Council

Standard: PCI Data Security Standard (PCI DSS)

Version: 1.0

Date: May 2016

Author: Effective Daily Log Monitoring Special Interest Group
PCI Security Standards Council

Information Supplement:
Effective Daily Log Monitoring

Document Changes

Date	Document Version	Description	Pages
May 2016	1.0	Initial release	All

Table of Contents

1 Introduction

One of the key tenets of almost any information security program is the concept of "defense in depth." Defense in depth is a tactical strategy for preventing the loss or compromise of assets through the implementation of an overlapping system of defenses consisting of multiple protective levels such that the failure of any single defense would not cause the failure of the entire system of defenses.

A defense-in-depth strategy typically involves a combination of preventive, detective, and corrective security measures. A rudimentary example of how defense in depth has been employed historically is a combination of fortress walls (preventative) with watchmen perched atop them at strategic points (detective). While this strategy has proven successful for thousands of years, history has also shown time and time again that attacks and attackers are continuously evolving. At some point, adversaries will develop the capabilities to defeat almost any defensive measure. The ability to quickly detect such circumstances and to adapt defensive tactics to counter attacks is paramount to the ongoing protection of assets. Successful detection of evolving attack techniques is predicated on having actionable intelligence. Having actionable intelligence requires that security defenses and the state of assets be continuously monitored. You would not build fortress walls to keep out intruders and then leave the walls unmanned. If security defenses were not continuously monitored, how would one know if an attack had compromised them? If we do not know the state of our defenses, how can we possibly know the status of our most valuable assets? Simply checking the vault to see whether the assets are still there is no longer sufficient, particularly in an age where a copy of an asset is as valuable as the asset itself, and the loss of the copy is as damaging as—if not more so than—the loss of the original.

1.1 Detective Measures in Information Systems

Since the advent of modern electronic computers, the concept of defense in depth has been widely employed in the protection of information systems. However, similar to the issues affecting historical assets, modern adversaries will eventually develop the capabilities to defeat some information system security defenses. Fortunately, in today's world, detection capabilities are built into most information systems by default through the implementation of logging mechanisms, which can provide organizations the actionable intelligence they need to help defend against evolving attack techniques.

Logging is functionality typically provided by things like operating systems, network devices, and software applications, which generate computerized messages when specific events occur. Those messages are captured in what is generally referred to as a "log" and may reflect a variety of events including the use of specific system resources, system status changes, and general performance issues. Logs are valuable sources of information because they provide a chronological record of events and activities that have taken place on information systems.

Originally created for troubleshooting errors and performance issues, logs have evolved to become the primary source of information on events related to information system security. System or application authentication attempts, file or data accesses, security-policy changes, and user-account changes are all

examples of events that are now captured in security logs[1]. In fact, because of the widespread deployment of networked servers, workstations, and other computing devices, and the ever-increasing number of threats against networks and systems, the number, volume, and variety of security logs have increased substantially (Kent & Souppaya, 2006). This provides organizations with a wealth of information relating to the state and effectiveness of information security measures deployed to protect the organization's information systems.

1.2 The Need for Log Monitoring

Having security logs and actively using them to monitor security-related activities within the environment are two distinctly different concepts. This sounds obvious, but many organizations confuse the former with the latter. Logging system messages and events in security logs may prove helpful—even essential—during post-breach forensic investigations. But having security logs without procedures to actively review and analyze them is of little use in the ongoing management of information security defenses, and is the modern equivalent of fortress walls without watchmen. For security logs to be useful in the defense of information assets, they must be monitored and analyzed—in as close to real-time as possible—so that attacks can be detected quickly and appropriate countermeasures deployed to augment existing defenses when and where necessary. This becomes increasingly important as attacks and attackers become more sophisticated. Without the active monitoring and analysis of security logs, the erosion of information security defenses by capable adversaries will likely go undetected and will eventually result in the compromise of the very assets that require protection.

1.3 Log-Monitoring Challenges

Advancements in technology have enabled those with malicious intentions to improve their craft. As attacks and attackers become more sophisticated and agile, it becomes increasingly important that we as security practitioners become more adept at maintaining and evolving effective measures to protect our information assets. This includes improving our ability to detect attacks and security failures *before* they lead to data breaches. Unfortunately, we do not seem to be very capable of doing that at the moment, as statistics indicate the time between system compromise and detection is averaging weeks and months when it should be measured in hours and days (Ponemon Institute, 2015). This situation is exacerbated by the glut of vulnerabilities that exist in today's information systems and the challenges associated with keeping systems up-to-date on security patches. There are only so many security resources available to perform security-related activities and, in many organizations, other activities including vulnerability management take priority over log monitoring (Black Hat, 2015).

The number of systems generating log data is rapidly expanding as well. The growth in the use of virtualization technologies and the emergence of on-demand scalability of computing resources have allowed many organizations to pack more systems and applications into increasingly smaller hardware architectures. Where there used to be a practical limit on the amount of physical space available to house information systems, virtualization—and cloud-based services in particular—have essentially nullified that issue. The rapid increase in system density has also resulted in exponential growth in the volume of log data that is

[1] For the purposes of this document, the terms "security log," "audit log," and "audit trail" are used interchangeably except where otherwise noted.

produced. This, in turn, has put tremendous pressure on security teams to process increasing volumes of information more quickly without additional resources to assist in the process. Additionally, logs do not necessarily speak the same language. There is no universally adopted standard for structuring or formatting log data. Logs can exist in numerous forms. Some systems and devices generate logs in the form of human-readable text files, while other systems generate log data in machine-readable data files or within relational databases. Some systems may even generate logs in proprietary formats. There is also no consistency in how event information is articulated within log files. The same event occurring on two different systems may be described completely differently by those two systems.

As mentioned previously, these issues place a substantial burden on security practitioners. It's no wonder that—given the amount of overhead seemingly required to manage and analyze log data and the limited number of resources that are available to do this work—many organizations come to the conclusion that the benefits of actively monitoring security logs do not outweigh the costs, and simply choose to devote resources elsewhere. In order to become more effective at log monitoring, organizations need to adopt a structured approach for generating, transmitting, storing, and analyzing security log data in the most efficient manner possible. Log-management processes must align with the organization's risk management strategy so that resources can be best utilized in the most effective and cost efficient manner. The approach must be customized to the organization's specific business mission, and support the culture and technology unique to the organization.

1.4 Guidance in this Document

There are many valuable resources available both in print and on the Internet to help organizations address the challenges of maintaining effective log-management processes. This document seeks to address these challenges by explaining the intent behind PCI DSS Requirements for log monitoring, and providing guidance on the planning, implementation, and application of effective log-monitoring and management practices. However, the primary focus of this document is log monitoring within the context of PCI DSS, and all discussions are intended to provide those with PCI DSS compliance obligations guidance on improving compliance with PCI DSS log-monitoring requirements. Those looking for more general guidance on the topic of logging and log management, please refer to the "References" section at the end of this document for a list of resources that should be considered for further reading.

This document is not intended to be a step-by-step guide for performing log monitoring and management, nor does it guarantee that the implementation of the tools and techniques mentioned herein will result in PCI DSS compliance. This document is intended to provide an overview of the key activities that comprise an effective log-monitoring program. The information in this document is intended as supplemental guidance and does not supersede, replace, or extend PCI DSS requirements.

1.5 Assumptions

The guidance in this document assumes readers are familiar with PCI DSS requirements, testing procedures, and scoping guidance, and possess an understanding of computer information systems, network technologies, and general IT principles and terminology. This document also assumes readers have some experience with security log monitoring as well as popular logging platforms such as Syslog or Windows Event Log.

2 Log-Monitoring Requirements in PCI DSS

The Payment Card Industry Data Security Standard (PCI DSS) is based on the concept of defense in depth and includes a variety of preventive, detective, and corrective information security measures (also called "security controls"). Moreover, PCI DSS includes requirements devoted to the use of log monitoring in the ongoing protection of information assets, addressing the need for proactive monitoring of security logs in Requirement 10.6:

> ***10.6*** *Review logs and security events for all system components to identify anomalies or suspicious activity*

The key elements of PCI DSS Requirement 10.6 are listed in the following three sub-requirements:

> ***10.6.1*** *Review the following at least daily:*
>
> - *All security events*
> - *Logs of all system components that store, process, or transmit CHD and/or SAD*
> - *Logs of all critical system components*
> - *Logs of all servers and system components that perform security functions (for example, firewalls, intrusion-detection systems/intrusion-prevention systems (IDS/IPS), authentication servers, e-commerce redirection servers, etc.)*
>
> ***10.6.2*** *Review logs from all other system components periodically based on the organization's policies and risk management strategy, as determined by the organization's annual risk assessment.*
>
> ***10.6.3*** *Follow up exceptions and anomalies identified during the review process*

On the surface, these requirements are straightforward. However, several terms require clarification before we can better understand the intent behind these requirements.

2.1 Key Terms

2.1.1 Security Event

The PCI DSS Glossary defines a security event as "an occurrence considered by an organization to have potential security implications to a system or its environment. In the context of PCI DSS, security events identify suspicious or anomalous activity." Examples of security events include attempted logons by non-existent user accounts, excessive password authentication failures, or the startup or shutdown of sensitive system processes. Unfortunately, determining the specific types of events and activities that should constitute security events is largely dependent on each individual environment, the systems resident in that environment, and the business processes served by that environment. Therefore, each organization must define for itself those system events and activities that represent "security events."

PCI DSS provides some insight into those activities that might constitute a security event by defining seven high-level activities that must be tracked:

> ***10.2.1*** *All individual user accesses to cardholder data*
>
> ***10.2.2*** *All actions taken by any individual with root or administrative privileges*
>
> ***10.2.3*** *Access to all audit trails*
>
> ***10.2.4*** *Invalid logical access attempts*
>
> ***10.2.5*** *Use 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*
>
> ***10.2.6*** *Initialization, stopping, or pausing of the audit logs*
>
> ***10.2.7*** *Creation and deletion of system-level objects*

Any of the activities described above that are performed without proper authorization would likely constitute a security event. For example, unauthorized individuals accessing cardholder data, audit trails, or making modifications to user settings or system-level objects might reflect an unauthorized elevation of privileges. Similarly, the frequent and successive occurrence of invalid logical access attempts might also reflect an attempt to brute force passwords.

How the activities described above and the potential security events they may represent map to specific messages within security logs will depend primarily on the information system generating the security log, the function that the system serves, the capabilities of the system's logging mechanisms, and how those logging mechanisms are configured. System-specific expertise will be needed to map system messages back to high-level security-related activities. You should consult the system administrators responsible for managing each system or review any system documentation provided by the systems' vendors for assistance in identifying the appropriate system messages that correlate back to these high-level activities. We will discuss event mappings in further detail when we discuss planning for effective log management later on in this document.

2.1.2 System Components

In this document, we make numerous references to "information systems" or "systems." It is important to point out that an information system may consist of multiple system components. A system component, however, is not necessarily limited to a physical asset—such as a server or a network device. Software applications are also components of information systems. The PCI DSS Glossary defines system components as "any network devices, servers, computing devices, or applications included in or connected to the cardholder data environment."

Many people and organizations often overlook this point and fail to include software and other applications in their log-monitoring processes—particularly things like management consoles, mail servers, anti-virus software, hypervisors, and other software applications resident in the cardholder data environment. PCI DSS provides some additional examples of system components that should be considered when establishing log-monitoring processes in the "Scope of PCI DSS Requirements" section.

2.1.3 Critical System Components

"Critical system components" are those system components that perform functions either vital to the operation or security of an information system or the cardholder data environment, or those components that—if compromised—could result in significant damages (whether financial or reputational) to the organization. Like "security events," determining which system components to classify as "critical" is left to each organization to determine for itself. Examples of critical system components might include databases containing cardholder data, unified storage systems, network authentication systems, or cryptographic systems—such as hardware security modules (HSMs). The system components an organization deems "critical" will likely depend on how the organization analyzes risk and assigns risk ratings to specific systems. Please refer to your organization's system classification policy, or contact your organization's risk management office (RMO) or those individuals responsible for conducting risk assessments for further guidance on determining how to classify systems and system components.

2.2 Requirement 10.6

Now that we have defined some of the key terms associated with Requirement 10.6, let us look at each of its sub-requirement in greater detail.

2.2.1 Requirement 10.6.1

PCI DSS Requirement 10.6.1 provides the foundation for the proactive monitoring of security logs for the occurrence of security events by requiring "daily" reviews of logs for critical system components. However, many people are confused by the use of the term "daily." Some people interpret this as "every business day," while others interpret it as "every calendar day." Keep in mind that for security logs to be useful in the ongoing defense of information assets and cardholder data, they must be reviewed in as close to real-time as possible so that attacks can be detected and countermeasures deployed *before* a data breach occurs. Nevertheless, not every organization has the resources or capabilities to support real-time event detection. Therefore, a reasonable timeline must be defined to allow less capable organizations to perform security log reviews while still enabling the organization to detect malicious or anomalous activity before it can likely escalate. In the case of Requirement 10.6.1, PCI DSS has determined that timeline to be a maximum of 24 hours or one calendar day.

While a 24-hour window was intended to accommodate less capable organizations, many organizations--including those with mature information security strategies—still struggle to meet the stated "daily" log review frequency. This then raises the issue of what it means to "review" a security log. Many organizations interpret this to mean a manual analysis of information contained within a security log. But daily log reviews need not be manual. There are numerous commercial and/or open-source tools available to help automate "reviews."

In fact, the note in Requirement 10.6 provides some examples of such automated tools that may be used to meet the intent of this requirement as well as the other sub-requirements under Requirement 10.6:

> ***10.6*** *Review logs and security events for all system components to identify anomalies or suspicious behavior.*
>
> ***Note:*** *Log harvesting, parsing, and alerting tools may be used to meet this Requirement.*

In addition to specifying that "daily" log reviews must be performed, PCI DSS Requirement 10.6.1 identifies what must be reviewed on a daily basis. The best way to clarify what must be reviewed daily is to look at each bullet individually. In the first bullet, the term "security event" is meant to represent those pre-defined events and activities that the organization has identified as being potentially "malicious or anomalous," including those specified in Requirements 10.2.1 through 10.2.7. Consequently, Requirement 10.6.1 requires the review of all security events on a daily basis. As mentioned previously, this may be accomplished through the use of alerting mechanisms. We will discuss alerting mechanisms in further detail later in this document. The latter three bullets then define *where* we are to look for instances of security events. However, the latter three bullets under Requirement 10.6.1 are not necessarily exclusive of one another. In many environments, "critical system components" might include "all system components that store, process, or transmit CHD and/or SAD." The same thing is true for "all system components that perform security functions." The intent of specifying the logs that must be reviewed in this manner is to ensure that regardless of how an organization defines critical system components, the organization includes logs from such systems in the daily log review.

2.2.2 Requirement 10.6.2

Requirement 10.6.2 is complementary to Requirement 10.6.1 and covers all of the other system component logs that are not addressed in Requirement 10.6.1. Like Requirement 10.6.1, the main focus of Requirement 10.6.2 is to review logs for all "security events." However, the frequency with which all other system component logs are to be reviewed is left up to the organization itself to define. Remember, a "system component" is a term used in PCI DSS to denote something that resides in or is connected to the cardholder data environment. Therefore, Requirement 10.6.2 is intended to cover all of the other "in-scope" systems. Systems that are neither located in nor connected to the cardholder data environment—or otherwise are not defined in the organization's PCI DSS scope—are not expected to be reviewed in accordance with Requirement 10.6.2.

PCI DSS Requirement 10.6.2 was introduced in version 3 of the Standard to provide organizations more flexibility in performing log reviews, by allowing them to define how often logs are reviewed for systems that do not fall under Requirement 10.6.1. This does not absolve organizations from having to review other "less-critical" system logs—as noted in the Requirement 10.6. It simply allows the organization to focus efforts on the highest-risk systems first.

2.2.3 Requirement 10.6.3

Requirement 10.6.3 is one of the most important requirements in all of PCI DSS for the ongoing protection of cardholder data, and is an often-overlooked element of log-monitoring processes. It requires follow-up on all exceptions and anomalies identified during the review processes identified in both Requirement 10.6.1 and 10.6.2. When security events are detected in system component logs, it is essential that those events be investigated further to determine whether the occurrence of each event actually represents something malicious or anomalous.

Deciding whether an event represents normal user activity or potentially malicious activity requires analysis of the context in which the event occurred. The occurrence of an event must be verified to represent something concerning. This is called "event validation." Let us consider an example: An event is generated that shows a database administrator successfully logged into a database server containing cardholder data. In most cases, this would not be a big deal since database administrators would be authorized to access such databases in order to maintain them. Now consider if the successful login was immediately preceded by five invalid login attempts by that same database administrator. It is this context in which this event has occurred (after five invalid login attempts) that gives new meaning to an otherwise normal activity. Therefore, the context in which a security event occurs must be evaluated further to help organizations differentiate between normal and abnormal activity. This is what Requirement 10.6.3 is intended to cover.

2.3 Other Important PCI DSS Requirements Related to Log Monitoring

2.3.1 Requirement 12.10

The whole point to the proactive monitoring of security logs is to facilitate timely response to potentially malicious activities *before* they become a big problem. How (and how quickly) an organization responds to a confirmed malicious event may be the difference between a minor security event and a major breach of cardholder data. Organizations must be prepared for such instances, and have appropriate response procedures and countermeasures prepared—in advance—to respond in a timely and efficient manner. PCI DSS Requirement 12.10 is intended to facilitate the advance planning of such incident response procedures.

> ***12.10*** *Implement an incident response plan. Be prepared to respond immediately to a system breach.*

Incident response planning is a critical component of any defense-in-depth strategy and is a topic that warrants extensive discussion and analysis. Unfortunately, providing guidance on defining incident response procedures would likely command its own Special Interest Group (SIG) and is, therefore, a bit beyond the scope of this document. There are, however, elements of incident response that warrant some discussion here.

2.3.2 Requirement 12.5.2

> ***12.5.2*** *Monitor and analyze security alerts and information, and distribute to appropriate personnel.*

Someone needs to be assigned responsibility for monitoring and analyzing security alerts. Requirement 12.5.2 is intended to ensure those responsibilities are formally assigned. In addition, Requirement 12.5.2 is intended to facilitate the incident response process in the event suspicious or abnormal behavior is detected by ensuring appropriate individuals are notified when such events occur.

The PCI DSS Glossary defines "monitoring" as "the use of systems or processes that constantly oversee computer or network resources for the purposes of alerting personnel in the case of outages, alarms, or other predefined events." Once an event has been confirmed to represent concerning or even known malicious behavior, it is the responsibility of log-monitoring personnel (as illustrated in the definition above) to alert the appropriate individuals so that suitable countermeasures can be deployed. Whether those individuals are other system or security administrators, management personnel, legal representatives, or even the payment brands will depend on the context in and extent to which the malicious activity has occurred. But there needs to be a well-defined "handoff" of responsibility between the log-monitoring function and the incident response function.

Although the intent here is to distinguish between the log-monitoring function and the incident response function, these need not be different individuals or teams. It is entirely possible—even likely in many scenarios—that log-monitoring personnel will be assigned responsibility for incident response in addition to their monitoring duties. The key point here is that in the case of a confirmed malicious event, there will be some point or "trigger" at which time the decision to initiate some type of response will become necessary.

2.3.3 Requirement 12.10.3

> ***12.10.3*** *Designate specific personnel to be available on a 24/7 basis to respond to alerts.*

Attackers do not operate on any particular schedule. Attacks and intrusions can occur at any time. Therefore, it is necessary that organizations have personnel assigned to respond to attacks 24-hours a day, 7 days a week. Requirement 12.10.3 ensures that personnel are available 24/7 to respond to security events and to initiate formal response procedures when required.

2.3.4 Requirement 12.10.5

> ***12.10.5*** *Include alerts from security monitoring systems, including but not limited to intrusion-detection, intrusion-prevention, firewalls, and file-integrity monitoring systems.*

The occurrence of malicious activity does not necessarily mean a data breach has occurred, but formal response procedures are still necessary to ensure consistent and appropriate response to such instances. Requirement 12.10 is intended to cover multiple scenarios, up to and including a confirmed breach of

cardholder data. Determining whether certain elements of an incident response plan require invocation or whether certain individuals need to be notified will, once again, depend on the context in and extent to which malicious activity has occurred. PCI DSS Requirement 12.10.5 is intended to ensure that, in addition to defining processes for responding to a confirmed data breach, incident response procedures[2] also include guidance for responding to "lesser" events, such as instances of known malicious activity.

2.4 Section Summary

The PCI DSS recognizes the importance of proactive monitoring of security logs in the detection of attacks on information assets and the protection of those assets from compromise. PCI DSS Requirements for log monitoring are covered under Requirement 10.6. Requirement 10.6 and its sub-requirements call for the review of system security logs to identify the occurrence of security events. Requirement 10.6.1 specifies daily log reviews for critical systems while 10.6.2 permits more infrequent log reviews for other less-critical systems. Requirement 10.6.3 then stipulates that detection of any security event be investigated further to confirm the occurrence of known malicious activity. Upon detection of known malicious activity, formal response procedures are necessary to respond to such events and to mitigate any further damages. Requirements 12.5.2 and 12.10.3 ensure that appropriate personnel are assigned responsibility for monitoring alerts and responding to security events on a 24/7 basis. Finally, Requirement 12.10.5 ensures that incident response procedures include guidance on handling instances of known malicious behavior. These requirements are intended to define a framework for the timely detection of potentially malicious behavior and the incident response procedures that should be performed to protect cardholder data in the event of an attack.

[2] In many organizations, incident response plans tend to be executive-level documents for dealing with major business disruptions such as earthquakes, hurricanes, terrorism, civil unrest, or other potentially catastrophic events. Although executive-level incident response plans may also include procedures for responding to data breaches, for the purposes of PCI DSS and this document, it is not necessary that PCI DSS requirements for incident response procedures be captured in these executive-level documents. It is up to the organization to determine how and where to document these procedures.

3 Planning for Effective Log Monitoring

Effective log-monitoring practices start with effective planning of log-monitoring needs and activities. To be most effective at log-monitoring (and to meet the intent of PCI DSS Requirements for log monitoring), organizations must have thorough understanding of their legal, regulatory, business, and operational requirements. In addition, they must understand the technical capabilities of the systems that need to be monitored, the technologies available to assist with monitoring processes, and the technical capabilities of other individuals and teams within the organization who can assist in developing effective and efficient log-monitoring practices.

3.1 Determine Your Logging Requirements

The key to ensuring the success of any log-monitoring strategy is to know what you want to accomplish before you start to build out the processes and the infrastructure to support it:

- What is it you want or need to monitor?
- Which systems and system components need to be included in the monitoring strategy?
- What information do systems need to capture in security logs?
- How do you want to go about capturing and analyzing security logs?
- How frequently should you review security log data?
- How long do you need to retain security log data?

To answer these questions, organizations need to first consider the applicable laws and regulations to which the organization is expected to adhere. Careful consideration should also be given to any existing organizational policies or risk management strategies employed by the organization. Since the primary focus of this document is log monitoring within the context of PCI DSS, we will focus on PCI DSS Requirements as the basis for defining logging requirements. However, organizations with other legal, compliance, or operational obligations should analyze their own security needs—in addition to those specified in PCI DSS—in order to ensure the log-monitoring and management processes they define meet all the needs of the organization.

3.2 Define the High-Level Activities You Wish to Monitor

The starting point for any entity looking to establish or improve existing log-monitoring processes is to define—at a high level—the types of activities that the organization would like to track as possible indicators of potentially “malicious or anomalous” behavior. The Security Information and Event Management (SIEM) industry often refers to these activities as “events of interest.”

As we noted back in Section 2.1.1, PCI DSS is specific when it comes to defining the basic high-level activities that must be logged and tracked, and lists those in Requirements 10.2.1 through 10.2.7:

> ***10.2.1*** *All individual user accesses to cardholder data*
>
> ***10.2.2*** *All actions taken by any individual with root or administrative privileges*
>
> ***10.2.3*** *Access to all audit trails*
>
> ***10.2.4*** *Invalid logical access attempts*
>
> ***10.2.5*** *Use 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*
>
> ***10.2.6*** *Initialization, stopping, or pausing of the audit logs*
>
> ***10.2.7*** *Creation and deletion of system-level objects*

In addition to the events defined above, PCI DSS identifies other activities that should be logged and tracked. Unfortunately, these are not always explicitly defined in the actual requirement language. For instance, Requirement 5.2 calls for all anti-virus mechanisms to "generate audit logs," but the language in the requirement does not identify the activities that must be captured in those logs. It is in the guidance for this requirement where the intent to log and track all "virus and malware activity and anti-malware reactions" is defined. Events from IDS/IPS systems (Requirement 11.4) and file-integrity monitoring (FIM) systems (Requirement 11.5) are also expected to be logged and tracked. In the case of Requirement 11.5, any modification (changes, additions, and deletions) of critical system files, configuration files, or content files should be monitored.

The purpose of defining security events of interest at a high-level is to provide security administrators flexibility in associating specific system messages or alerts to these high-level events. As we discussed earlier in this document, there is no universally adopted language for log messages. The same high-level event may be reflected differently in the security logs of different systems. Therefore, it is important from a planning standpoint to discuss logging requirements at a high-enough level such that security personnel can determine appropriate system-level messages that relate the high-level events described above. Once those system-level messages are defined, a mapping exercise should be performed to map system-specific messages back to high-level events. We will discuss that mapping exercise later in this section.

There may be other events of interest that organizations wish to define in addition to those specified in PCI DSS requirements. Examples of such events may include the detection of active malware on a web server, the presence of CHD in an unauthorized location, or multiple attempts to connect to a database server containing CHD from an unauthorized source (such as an external IP address). How additional events of interest are defined by the organization will depend greatly on the environment, the technologies in use, and the organization's risk management strategy. Additionally, while PCI DSS provides some structure and guidance on what should be logged, tracked, and monitored to meet PCI DSS compliance requirements, it is

up to the organization to determine for itself which activities and events should be monitored to meet the organization's *security* objectives. While PCI DSS is intended to ensure effective security controls are implemented to protect CHD, organizations should consider their security needs beyond PCI DSS to ensure all of their information security objectives are properly addressed, not just those related to CHD.

3.3 Identify Potential Log Sources

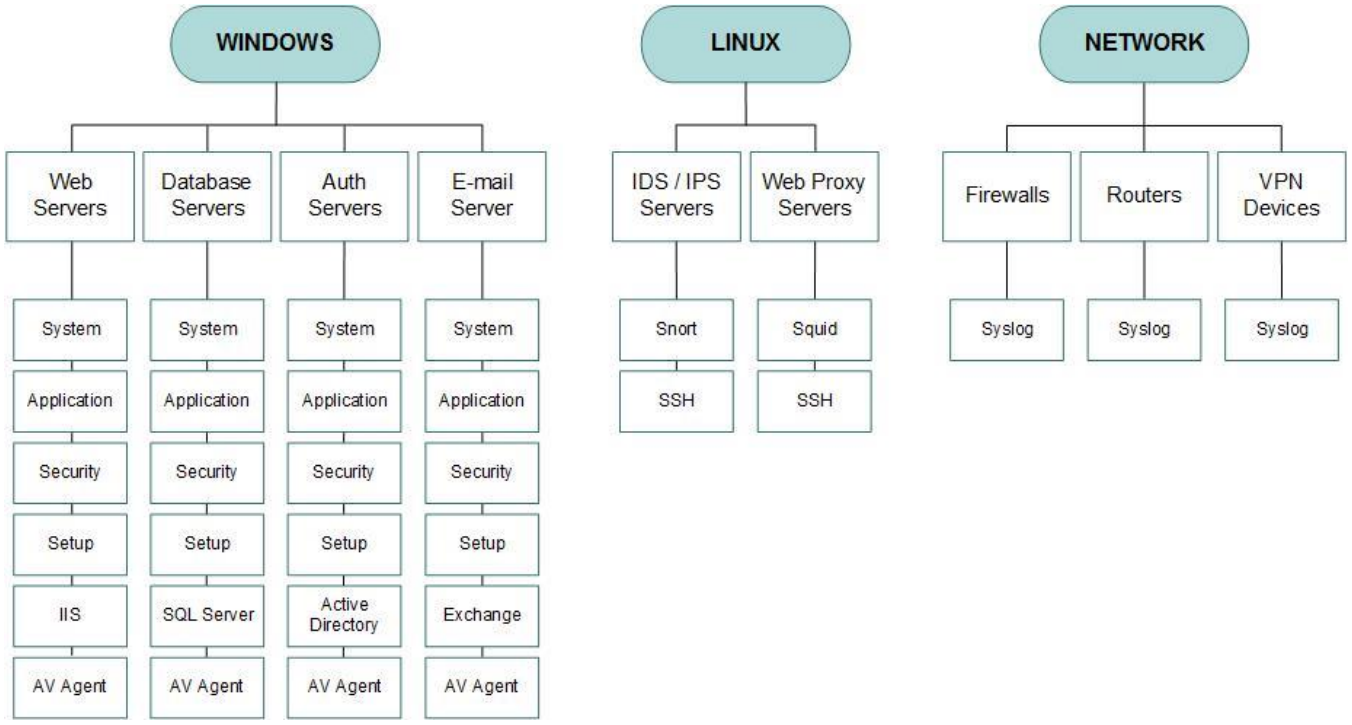
The next step in building an effective log-monitoring program is to understand the environment you're looking to monitor and the logging capabilities of the systems within that environment. As part of the PCI DSS scoping exercise, you should have already identified all of the in-scope systems. Now you need to look at each system individually and identify the various logs generated on those systems.

Very few systems today are incapable of generating logs. In fact, most Operating Systems generate multiple logs. Windows-based systems, for example, typically generate four distinct types of logs: System, Security, Application, and Setup logs. Linux-based systems are known to generate even more logs, many of which are located in the "/var/log" directory. There may be other logs generated by systems depending upon the function the system serves and the software running on them. For example, web servers running Microsoft Windows with Internet Information Services (IIS) will generate IIS logs in addition to the operating system logs mentioned above. Likewise, a Linux system running Apache web server will generate log messages in Apache-specific log files. Other types of software they may generate additional logs include database server software, anti-malware software, intrusion-detection/prevention software, and e-mail server software. The number of logs generated and the location of these logs will depend upon the configuration of the operating system as well as the software running on it. You will want to consider all software running on each system as you identify potential log sources. Refer to documentation provided by the vendors of the software running on those systems to determine the logging capabilities and locations of potential log sources.

Once you have identified all of the potential log sources on each in-scope system, it may be helpful to represent potential log sources in a tree map (Frye, 2009). To do so, categorize systems by operating system, and then break those systems down further by the function of the system[3] (web-server, database server, etc.), and then the logs available on those systems to provide a general overview of the overall logging infrastructure (see Figure 1 for an example).

[3] Keep in mind that PCI DSS Requirement 2.2.1 mandates that only one primary function per server be implemented to prevent functions that require different security levels from co-existing on the same server.

Figure 1 - Log Source Tree Map



3.4 Document Log Source Characteristics

Understanding log formats, as well as what information is captured within each log, is the next step in the planning process. As we mentioned earlier in this document, log sources do not conform to a standard log format. To be able to extract information from potential log sources, you need to know the names of the logs, where the logs are located, the formats those logs are in (e.g., human-readable, machine-readable, or proprietary), the tools necessary to be able to read the logs, and the type of information captured within those logs. Vendor documentation may be required in order to determine this information. If vendor documentation is not available, ask the system administrators responsible for managing each system and/or software component for assistance in specifying these details.

To conduct this exercise effectively, it is necessary to ensure that all logging mechanisms on a particular system are configured and running. It is also important that the logging mechanisms be configured to log at the highest verbosity setting feasible so that all information the logs are capable of capturing can be evaluated (Frye, 2009). You can scale back logging verbosity settings after you have completed this exercise and once you have documented the information available in each log and the minimum log settings capable of generating the information needed to capture the occurrence of security events. You should refer to system and/or software documentation to determine how to modify logging verbosity settings for each application or logging mechanism and the appropriate settings for your environment.

After logging mechanisms have been properly configured and activated, you will need to look at each log individually to identify the type of information captured within each log. For the purposes of PCI DSS compliance, the logs should capture the key data points specified within PCI DSS Requirement 10.3 at a minimum:

> ***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*** *Identify or name of affected data, system component, or resource*

The actual data fields in each specific log may differ from what is specified in Requirement 10.3. However, most logs should have data fields that generally map to the data points defined in Requirement 10.3. For example, most logs have fields for *Source IP* and *Destination IP*. These fields generally map to data points specified in Requirements 10.3.5 and 10.3.6 respectively. The same goes for *Timestamp* fields and Requirement 10.3.3. Some logs, however, may not have specific fields to map back to those data points in Requirement 10.3. However, that does not mean the information is not available. That information may be obtained in other fields. One common example involves *Success or failure indication* in Requirement 10.3.4. Many logs do not have a specific "status" field. However, this information is often provided as part of a *System Message* field, which may contain information on multiple data points, including the type of event, success or failure, the specific user ID involved, etc. Each log is different so you will want to evaluate each log type (see Section 3.3) to determine what information and data points are available. Once again, use vendor documentation and/or internal system expertise to help you identify the relevant data points.

Lastly, you should document the characteristics of each log so that the information can be referenced in later planning activities. It is recommended that, at a minimum, the following characteristics be captured for each log type (Frye, 2009):

▪ Log Name	▪ Application Name (which generates the logs)
▪ Description	▪ Application Version
▪ Location (file, path, database table, etc.)	▪ Operating System
▪ Log Type / Format (text, XML, etc.)	▪ Relevant Data Points

These characteristics can be documented in a number of different ways, including policy documents, spreadsheets, databases, XML files, Visio diagrams, etc. The important thing is that they be documented somewhere as a permanent and accurate reference for log-monitoring personnel.

3.5 Identify and Map System-Level Event Messages to High-Level Events

For each of the high-level events of interest you defined earlier, you will need to investigate the logs from each log source type to determine how the logs identify those same events within system-level log messages. This may require you to analyze logs from a range of dates in order to identify the relevant system-level messages. If you're using a third-party tool such as a SIEM solution, many SIEM tools come pre-installed with templates that already map certain system-level events from many common log sources (such as network equipment, operating systems, databases, etc.) to common events of interest, such as "failed user logons" and "user account changes." Therefore, there may only be limited manual analysis that is required to map system-level events from sources not supported by the SIEM. For those who are not using a third-party tool or solution, there may be some heavy lifting required up front to investigate each log to identify and map all the relevant system-level messages back to your high-level events of interest. Fortunately, there are many resources available on the Internet that help you identify what you should look for in your logs[4].

As you work through each of the high-level events of interest, you should document the mapping between the high-level events and the corresponding system-level messages for those log source types unique to your environment. For example, you may want to keep a spreadsheet that includes information similar to the information below:

Event Type	Message Summary	Event Source
Failed User Logon	*An account failed to failed logon*	Microsoft Windows
	Event ID = 4025	Microsoft Windows
	Authentication failure	Linux
	Failed user authentication	Microsoft IIS

Documenting the mapping between high-level events of interest and system-level messages provides other log-monitoring personnel with a valuable reference should new systems be added or upgraded, or when scripts need to be modified to support a new log source or log source type.

3.6 Prioritize Log Sources

As we noted earlier in this document, very few systems today are incapable of generating logs. However, that is not to say that all logs from all systems need to be collected and analyzed with the same frequency and rigor. Additionally, system components residing on the same system may not need to have both sets of logs captured if there is redundancy between the information generated by the two[5]. In order to be the most effective and efficient at log analysis, we need to focus first on those systems and system components that are the most important and represent the most risk to stakeholders. PCI DSS Requirement 10.6.1 provides some direction in determining the most critical system components by identifying the types of system components that require daily log reviews: "All system components that store, process, or transmit CHD

[4] https://zeltser.com/media/docs/security-incident-log-review-checklist.pdf

[5] For further information, please refer to FAQ #1081, titled "Does PCI DSS Requirements 10.2 and 10.3 mean that both database and application logging are required?" available on the PCI SSC website at: https://www.pcisecuritystandards.org/faqs

and/or SAD" should probably be considered among the most important. The same likely goes for "all servers and system components that perform security functions (for example, firewalls, intrusion-detection systems/intrusion-prevention systems (IDS/IPS), authentication servers, e-commerce redirection servers, etc.)." Organizations should evaluate the results of their last risk assessment (as specified in PCI DSS Requirement 12.2) to help determine which systems the organization deems most critical, and whether any of those systems should be included with the previously described systems in the daily review process. Additionally, system component inventories (Requirement 2.4), and network and cardholder data flow diagrams (Requirements 1.1.2 and 1.1.3, respectively) may also help identify those system components that meet the criteria specified above. For all other in-scope systems, the organization should define criticality based on its risk assessment framework[6].

As you evaluate each system and system component, do not overlook other software or system components that comprise a particular system. Systems should be classified according to the highest classification of any system component comprising that system. For example, let us consider a server containing virtualization software is comprised of a hypervisor and several virtual machines (VMs). One of those VMs includes a database containing cardholder data. Naturally, the database containing cardholder data would be considered a "critical system component" per Requirement 10.6.1. But many people fail to properly classify the VM and the overarching hypervisor as critical system components as well. Compromise of either the VM or the hypervisor would likely lead to the compromise of the resident database. Therefore, logs from those systems should also be captured and be included in the daily log review process.

Careful consideration for each system component and the relationship it has with other system components is essential in evaluating potential threats to information assets. Additionally, system and system component classifications and justification for those classifications should be documented. One logical place to do so might be in the system component inventories described in Requirement 2.4. Having documented justification for all system classifications can help defend any decisions to exclude certain systems from daily log review requirements.

3.7 Determine Who to Notify When Security Events Occur

Once you have defined the key events of interest, and have identified and prioritized the systems you wish to monitor, you need to determine which individuals and/or teams should be notified when such events occur. PCI DSS Requirement 12.5.2 ensures that appropriate individuals are formally assigned responsibility for monitoring and analyzing security alerts and information, and Requirement 12.10.3 ensures such personnel are made available on a 24/7 basis. In many cases the personnel that are assigned responsibility for responding to alerts will depend on the specific event, and/or the system(s) on which the event(s) occurred.

For each of the events identified by the organization and those defined in Requirement 10.2, consider the potential impact of that event occurring on each system and system component. Consideration may need to be given to each individual system's classification, as those resources that require notification may differ from

[6] For additional guidance on classifying systems based on risk, please refer to the PCI DSS Risk Assessment Guidelines Information Supplement available on the PCI SSC website at: https://www.pcisecuritystandards.org/documents/PCI_DSS_v2_Risk_Assmt_Guidelines.pdf

system to system based on the criticality of that system. Additional considerations may need to be given to the "owner" of the system (business owner vs. administrative personnel), the time the event occurs (and whether or not separate "after hours" personnel should be notified), and the location of the system (and whether local or remote personnel require notification). Other organization-specific considerations may be necessary and should be identified and evaluated by each organization on a case-by-case basis.

3.8 Determine What Should Be Done in Response to Security Events

The response procedures for each high-level event should also be specified (see Requirement 10.6.3 and 12.10.5). Once again, this will depend largely on the event, the type of system the event occurred on, and the criticality of that system. Likewise, response procedures may also depend on other factors, including the time of day the event occurred as well as the location of the event. Response procedures need not be highly detailed, as a general description of the procedures should suffice. However, the more detailed the procedure, the less likelihood responsible personnel may misunderstand those procedures. So it may be in the organization's best interest to be as detailed as possible.

3.9 Document Logging Requirements

3.9.1 Logging Policy

Now that you have identified the key requirements for logging and log monitoring, you need to document them in a Logging Policy. A Logging Policy generally describes the business, regulatory, compliance, and/or security requirements for logging and log monitoring. For the purposes of this document, we would want to identify the purpose of logging and log monitoring as it relates to each specific organization (see Requirement 10.1); the roles and responsibilities of personnel tasked with performing functions related to logging and log monitoring (see Requirements 12.5.2 and 12.10.3); the high-level events the organization wishes to track (see Requirement 10.2); and the general systems and system components that should be monitored (see Requirement 10.6). Additional information, such as the specific information that must be captured in logs (see Requirement 10.3), the configuration of specific elements of the logging infrastructure (see Requirements 10.4 & 10.5), and the retention periods for individual logs (see Requirement 10.7) may also be included in a Logging Policy. How and where to document logging and log-monitoring requirements, as well as what is included in those documents, will be up to each organization to determine for itself.

3.9.2 Use Cases

Another helpful method for tracking log-monitoring and management requirements is through documentation of event "use cases." An event use case is mechanism by which personnel and/or automated systems are instructed what to do when a particular event occurs. That mechanism can be anything from a document, a script, or even a rule in an automated system.

Use Cases are most commonly employed in the SIEM industry and generally corresponds to rules or dashboards associated with a particular SIEM solution. However, the concept of use cases can be easily applied to environments that do not leverage SIEM solutions[7].

Use cases are usually organized by the specific high-level events an organization wishes to track. For each high-level event, a separate document would be generated. Within that document, the following information is typically captured:

1. Use Case Name (for example, "PCI Failed Logins")
2. High-level description of the event to be tracked (e.g., "Failed login attempts on all PCI DSS systems")
3. The systems on which the events are to be tracked
4. The individuals/teams that should be notified when such events occur
5. The response procedures associated with the event

Additional information such as the associated PCI DSS Requirements the use case is intended to satisfy, a description of the mechanisms used to notify responsible personnel, and any metrics that can be obtain from the implementation of the use case may also be included in the document. A detailed example use case is provided in Appendix A. Organizations are free to use the example as a template for their own use cases, develop their own templates, or forgo the use of use cases altogether. Use cases are merely another tool—in addition to policy and procedure documents—for documenting and tracking events.

An important point to make about use cases is that they are not static documents. Much like policies, they are living documents and need to be developed and cultivated over time. You will need to revisit any use cases you develop after you've defined your activity-monitoring baseline (described later in this document) to make sure your use cases are still relevant, whether additional information should be captured within the use cases, or whether new use cases need to be developed. In addition, both the logging policy and use cases should be re-evaluated periodically to ensure they remain aligned with changes in organizational risks, business processes, and monitoring needs. As business needs change, so do monitoring needs. Documented monitoring requirements, policies, and processes must be evaluated frequently to ensure they continue to meet the needs of the organization.

[7] Many of the concepts and terms used in conjunction with SIEM technologies are also applicable to non-SIEM environments. PCI DSS does not mandate the use of SIEM or other log-management and analysis technologies However, as noted in this document, it is becoming increasingly difficult to maintain an effective log-monitoring program and, therefore, meet PCI DSS requirements for log monitoring without tools to automate the management and analysis of logs.

4 Preparing for Effective Log Monitoring

Another key component of an effective log-monitoring program is effective log-management practices. Once you have defined your requirements for log monitoring, you will need to define the processes, tools, and infrastructure that will be used to manage logs in accordance with those requirements.

4.1 Identify the Tools & Resources to be Used for Log Management

Although it's not required by PCI DSS, it is becoming increasingly infeasible to be effective at log monitoring and management without some form of automation. Whether you use scripts, native alerting mechanisms, or third-party Log Management, SIEM, or Advanced Analysis solutions, some level of automation is necessary to process, transmit, analyze, and alert on security event information. Even in small environments, the speed and volume with which log data is generated today makes manual processing and analysis completely impractical. It is no longer feasible (not that it ever was, really) to assign people to watch data scroll across a screen and expect them to extract the salient information from those streams in an effective and efficient manner.

Fortunately, there are a wide variety of tools available today, including both commercial and open-source solutions, to help organizations improve and optimize their log-monitoring and management processes. In the case of commercial solutions, many vendors even offer monitoring services to further assist organizations in simplifying the monitoring processes. Nevertheless, before you opt to buy, build, or outsource a log-monitoring solution, you need to assess the tools and resources already at your disposal. Depending upon your log-monitoring needs (see Section 3.1 for Log-Monitoring Requirements), simply having scripting expertise in-house may provide you with all the necessary tools you need. You should talk to other teams within your company to learn whether they may have tools you could leverage. Finally, you need to consider the types of systems in your environment and the personnel available to assist you in configuring the necessary tools. Once you have identified all the tools that are going to be used and all of the resources available to help you manage them, you can start to build out the logging architecture.

4.2 Establish Central Repository for Log Data

Most systems have a limited amount of storage space available for the capture and storage of log data. Even those systems with a seemingly unlimited storage available will likely incur performance problems if logs are not pruned periodically to eliminate stale or outdated information, and to keep logs to a manageable size. However, as suggested by PCI DSS Requirement 10.7, it is often necessary to retain log data for historical purposes:

> ***10.7*** *Retain audit trail history for at least one year, with a minimum of three months immediately available for analysis (for example, online, archived, or restorable from backup).*

Many logging mechanisms will begin to overwrite older log entries after the logs reach a certain point or exceed a preconfigured size. This presents a problem when organizations are required to retain log data for longer periods of time. Furthermore, if log data is overwritten before it can be processed and analyzed by log-

monitoring personnel, it is likely that log-monitoring personnel will be unaware of potential security-impacting activity occurring on such systems. Configuring the logs to overwrite themselves or manually modifying log entries before the logs can be copied or reviewed by log-monitoring personnel is a common tactic employed by intruders and employees alike to conceal malicious activities. For these reasons, it is necessary to transfer logs to a centralized repository where they can be appropriately protected. PCI DSS Requirement 10.5 and its sub-requirements describe the methods that must be employed to protect logs:

> ***10.5*** *Secure audit trails so they cannot be altered*
>
> ***10.5.1*** *Limit viewing of audit trails to those with a job-related need*
>
> ***10.5.2*** *Protect audit trails from unauthorized modifications*
>
> ***10.5.3*** *Promptly back up audit trail files to a centralized log server or media that is difficult to alter*
>
> ***10.5.4*** *Write logs for external-facing technologies onto a secure, centralized, internal log server or media device*
>
> ***10.5.5*** *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).*

There are two key reasons for leveraging a central repository for log data storage. First, as we just discussed, is for the long-term storage of log data for historical purposes. In the event of a breach, access to the original logs (or unmodified copies of the original logs) is critical to determining the root cause of the breach as well as identifying the potential culprit(s). Centralizing of logs simplifies management of those logs. If the logs are kept in a central location, it is far easier to keep track of the mechanisms employed to protect the logs than it is if they are stored on numerous point systems.

The second reason for having a central repository for the storage of logs is so that operations can be performed on the logs in order to extract the information needed to meet log-monitoring requirements. However, you need to be careful that you do not make any modifications to the original logs while performing the necessary operations to extract relevant log data. If the original logs have been modified in any way, the reliance that can be placed on those logs—particularly in a court of law—is diminished. Therefore, strict protection mechanisms, including those specified in Requirement 10.5, are necessary. Depending upon the tools and technologies in use to process log data, it may also be necessary to create a second repository where operations can be performed on the logs in order to extract relevant event information, separate from the long-term storage of the original logs. We will talk about the need for additional repositories when we discuss data normalization in an upcoming section.

4.3 Transport Logs to the Centralized Repository

As is the case with many factors involved in log monitoring, the way in which logs are backed up, transported, or offloaded to the centralized repository will depend on the technologies in use. For Linux and other Unix-based systems—including most network devices—it is likely that Syslog will be used to transport log data to the central repository. For Windows systems, transport methods may involve the use of an agent, WMI calls, SNMP traps, or good old-fashioned file/data transfers using FTPS or SCP. Transferring log information from proprietary sources such as databases or other application logs to a central repository will likely require the use of a third-party tool, custom scripts, or some other mechanism. In fact, the repository itself may be a third-party system or solution, such as a SIEM. Regardless of the mechanism you use to transport the logs to a central repository, make sure the logs are pristine copies of the original log data so you can ensure you are working with accurate information. You may need to refer to the specific system or application documentation, or contact the vendors for information on how to transport log data to the central repository.

How frequently logs are transported to the central repository is an important consideration as well. If you are using the central repository for real-time analysis of events, it must be updated as closely to real-time as possible. The criticality of the system may also factor into the frequency with which logs are polled or transported to the central repository, giving priority to the most critical systems first. For example, an authentication server within the CDE may have its log data harvested every 10-15 seconds while the log data of an internal storage management console may only be harvested every 60 seconds.

4.4 Prepare Log Data for Processing

In order to be as efficient as possible during monitoring and analysis, you need to implement mechanisms or processes to prepare the log data to be processed and analyzed as quickly (and as accurately) as possible. One way to achieve this is to filter (or parse) the logs to focus on only the information that is needed to satisfy operational, security, and compliance needs for log monitoring. Otherwise, you'll spend valuable resources to store and process information that ultimately provides little to no value to your organization

For efficient filtering to take place, log data needs to be converted to a common format such that filtering mechanisms can be applied quickly and consistently. Trying to process and analyze data in different formats and in different locations creates inefficiencies and often requires multiple, redundant processes or mechanisms in order to extract relevant security data. Unfortunately, while many logs support common log formats (like Syslog), there is no universally adopted log-formatting standard. In many cases log formats are proprietary and/or require a third-party tool to view or process information in the logs. This is one of the bigger challenges to effective log monitoring.

Fortunately, many tools exist today to help facilitate the transformation of log data from numerous, heterogeneous sources into a single common format using mapping templates and data conversion mechanisms. This process is called data "normalization." For a sample list of tools available to assist in the process of normalizing log data, please refer to Appendix B. Additionally, if the use of a third-party tool is not an option, consider working with the log source vendors or internal resources to configure logging mechanisms to support as few common formats as possible. It is far easier to support two or three different log formats than it is trying to support five or more. The fewer formats you have to support, the fewer redundant processes are necessary to process and analyze data.

After log data has been normalized, you will need to parse or filter the log data for information related to the specific event messages you have identified in previous activities described in this document. A good resource for helping you with this process is the event-mapping document described earlier in Section 3.5. Using regular expressions with scripting languages such as Perl and Python, or other text processing languages like "sed" and "awk," is one method for filtering log data in text-based logs. Structured Query Language (SQL) queries or pre-existing APIs can be used to extract log data from databases. Filtered events should then be captured into a single repository that represents all events that have occurred on all systems. Centralization of relevant event messages allows for faster processing of event data as well as more complex operations like event correlation[8]. Once again, third party log-management tools are available to assist with these activities. Please refer to Appendix B for a sample list of such tools.

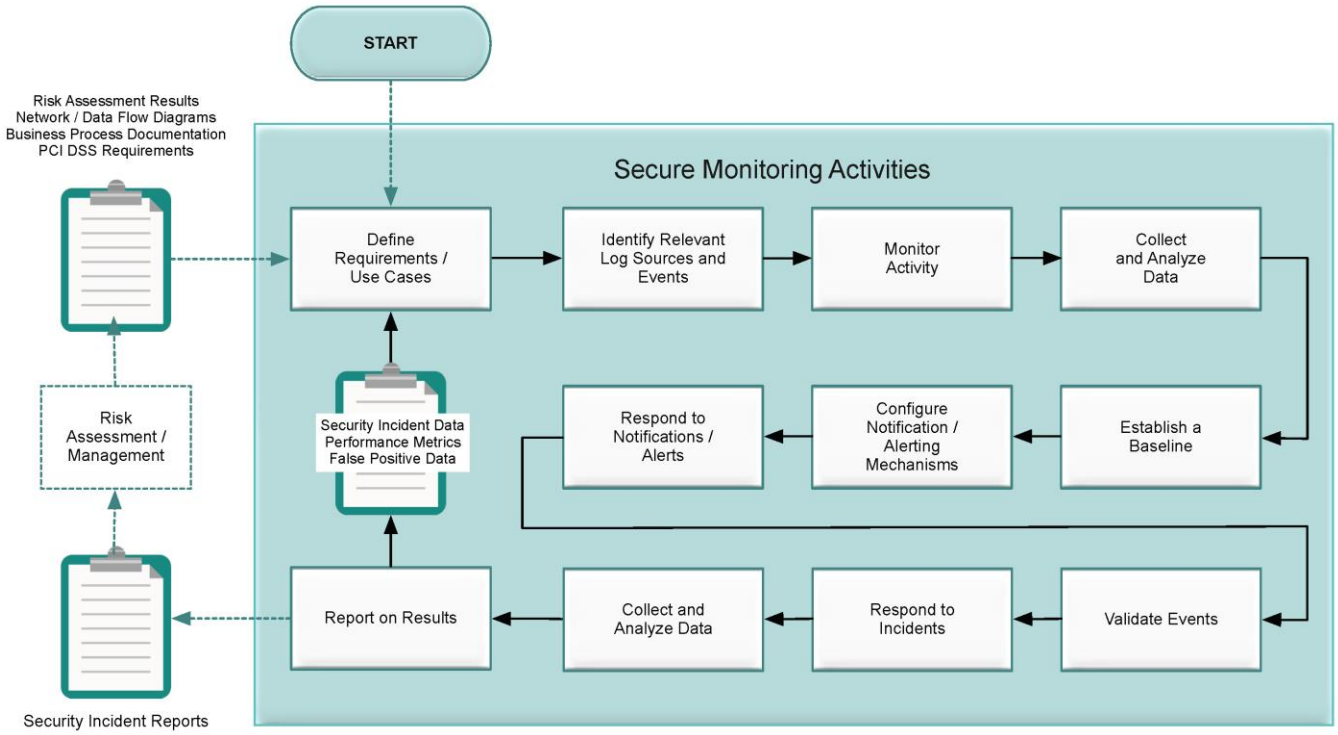
[8] Since event correlation is not specifically required by PCI DSS, it is not covered in detail in this document. For more information on event correlation, please refer to the "References" section for other publications that will likely provide more a detailed explanation.

5 Performing Effective Log Monitoring

You have defined your logging requirements and have designed and implemented your logging architecture. You have all your events captured in one place and in one format. Now it is time to put the log-monitoring process into action.

Effective log monitoring is not a tool or a technology, but rather a process that requires continuous improvement. The key activities associated with that process are described in Figure 2.

Figure 2 – Log-Monitoring Process Cycle



Continuous improvement of the log-monitoring process defined above is achieved through the implementation of its key activities as part of a virtual cycle. Each individual activity provides input into subsequent activities, which, in turn, feed back into the consideration of log-monitoring requirements. Additional inputs are received from parallel or supporting activities such as the risk assessment and PCI DSS scoping exercises.

For the purposes of this document, we have already defined (and hopefully performed) the first three activities in the cycle: requirements definition (Sections 3.1 and 3.2), log source and event identification (Sections 3.3 and 3.5), and activity monitoring (Sections 3.5 and 3.6). Now you need to analyze the activities described in log data to help differentiate between normal activity and abnormal or suspicious activity.

5.1 Collect and Analyze Activity Data

Keeping your log-monitoring processes optimized requires frequent analysis of the activities taking place in your environment. The key to effective analysis is to look at log data within a specific time frame. Keep the time window manageable. Too large of a population will exhaust the limited resources you have available to you to perform analysis. Too small of a population may not reflect some activity persistent in the environment. Appropriate time frames will differ from organization to organization and will be somewhat dependent on the number of systems in the environment and the amount of log data generated by those systems. But a good starting point is to begin with two weeks of data and scale up as needed.

5.2 Establish a Baseline

5.2.1 Look for Patterns

Activity patterns can be good indicators of normal as well as abnormal activity. Log data analysis is intended, in part, to help you differentiate between normal and abnormal behavior. If you have the ability to sort or filter log data by certain data fields within your centralized event repository, you may want to start by sorting activity by Source IP/port and Destination IP/port. Look at the frequency in which certain systems are accessed. Also look for common activities that take place and then factor in where those activities originate and the systems that are targeted by those activities. Some key metrics to consider include:

▪ Avg # of user logins to a specific IP per day	▪ Avg bytes of data transferred per destination IP per day
▪ Avg # of user logins from a specific IP per day	▪ Avg # failed logins per user per day
▪ Avg # of times a specific IP/port is hit per day	▪ Avg # failed logins per user per day per source IP
▪ Avg # of user logins during off hours per host per day	▪ Avg # of times processes stopped/started per day
▪ Avg bytes of data transferred per source IP per day	▪ Avg # error messages per system per day
▪ Typical geographical region data is sent to	▪ Avg # of log messages per day

If you are using a SIEM or other data analytics tool, activity summary reports are a great source for this kind of information. Scripting languages can be used to parse log data to obtain this type of information as well.

Other patterns to consider may be a bit more complex. Once again, the context in which certain events occurred must be considered. For example, multiple invalid logon attempts using an administrator account followed by a successful logon using the same account is certainly one pattern that may warrant investigation. Additionally, the creation of a non-admin user account followed immediately by admin-level privilege escalation is another example of activity patterns that may reflect potentially malicious behavior. The ability to identify these types of patterns depends greatly on the tools and methods used to parse log data. Those organizations leveraging SIEM and data analytics solutions will likely find it much easier to identify these types of patterns than those who are using scripting languages to parse data. However, with that said, it is certainly possible to obtain this type of information without complex analysis tools. It is likely that advanced scripting skills will be required, though.

5.2.2 Consider Other Data to Baseline

In addition to activity captured within logs, it may also be helpful to capture some baseline information about the operating environment in general. Systems that serve a common function (such as webservers) should have almost identical configurations and operating characteristics. The processes running on those servers should be consistent across all like systems. The same is true for all users authorized to access those systems. Any activity or circumstances that extend or contradict the baseline characteristics of any system should be investigated. Examples of potentially suspicious activity include the startup of a new process uncommon to those systems. A new FTP daemon running on a database server might be cause for concern. The creation of a new administrative user on a single system within a collection of like systems is another example of activity that has potentially malicious intentions.

5.2.3 Define Common Activity Constraints and Rules

For those activities that are common, you need to determine at which point those activities become suspicious. In many cases, activities become suspicious if the frequency in which they occur exceeds average frequencies. Using frequency to define activity thresholds as indicators of suspicious activity is one good method for differentiating between normal and potentially abnormal activity.

Time and location constraints should also be considered. For instance, changes to key systems—such as the shutdown of audit logging mechanisms—outside of well-defined maintenance windows could be an indication of potentially malicious behavior. Additionally, activity originating from an IP address located in region of the world where the organization has no business interests or personnel might also indicate suspicious behavior.

5.3 Configure Automated Alerts

Once your baselines have been defined, you will want to create automated alerts to notify appropriate personnel of activity or behavior that exceeds those baselines. You will also want appropriate log monitoring personnel to be alerted of any occurrence of certain events, not just those that exceed baselines. For example, if anti-malware software detects active malware on a critical server containing cardholder data and is unable to clean the malware from the system, then that is probably something that should be addressed as soon as possible.

Automation is a critical component of the notification process and the overarching log monitoring process as well. For logs to be useful in the defense of information assets, they must be monitored and analyzed—in as close to real-time as possible—so that attacks can be detected quickly and appropriate countermeasures deployed to augment existing defenses when and where necessary. Without automated alerting mechanisms, it is almost impossible to identify and alert about such events in near real-time. And without near real-time notifications, the ability to deploy countermeasures to protect information assets before they can be compromised is greatly diminished. Fortunately, as is the case with logging mechanisms, many systems today also support alerting mechanisms. Operating systems, in particular, generally package logging and alerting mechanisms together. Other system components, however, may not support alerting mechanisms. In these situations, custom scripts or third-party tools may be necessary to generate appropriate alerts.

Despite the availability of point solutions, centralization of alerting mechanisms is preferred. Centralized alerting simplifies alert management and allows for more complex rules and alerts to be created to take into account how events may transcend multiple systems (e.g. event correlation). Most tools that are designed to help with log data filtering and normalization also provide centralized alerting mechanisms. Please refer to Appendix B for a sample list of tools and solutions that can be used to assist in the notification process.

5.4 Respond to Alerts

When alerts are generated, they must be responded to in a timely manner in order to be useful. The whole point of log monitoring is to detect malicious behavior *before* it becomes a major issue. PCI DSS Requirements 12.5.2, 10.6.3, and 12.10.3 support the need for timely response to events by requiring that personnel be assigned responsibility for monitoring and analyzing security alerts and information; that exceptions and anomalies identified during the review be followed up on; and that personnel be made available on a 24x7 basis to respond to such events. The longer the period between alert generation and follow-up, the more time a seemingly innocuous event may be permitted to escalate.

However, in addition to ensuring they are followed up on in a timely manner, alerts must also be based on relevant information, meaning they must pertain directly to the risks and concerns of the organization. Alerts created based on non-essential or irrelevant events are just noisemakers and can be detrimental to the overall efficiency and effectiveness of the log-monitoring program. Alerts should only be configured on events that *require* follow up. In other words, alerts should only be created for those events that the organization deems important enough to warrant active (and timely) response. Alerts based in “informational” events serve no practical purpose other than to consume resources that should otherwise be allocated to more critical activities.

5.5 Validate Events

During our discussion of Requirement 10.6.3 in Section 2.2.3, we discussed the concept of event validation. Upon notification that a suspicious event has occurred, the context in which that event occurred needs to be analyzed to confirm whether or not the event indeed represents abnormal or malicious activity. This may require that other data or events related to the system that generated the original event message be analyzed. You should look for similar activity being performed on the system in question. If the initial event that triggered the validation process is associated with a particular user, you should look at other activities associated with that up to and around the time frame in which the initial event occurred. You should also look at what other non-user associated activities may have been going on at the time the initial event was generated, such as system error messages or system-level processes that may have been stopped/started in and around the same time frame.

Event validation may also require that events from other systems also be analyzed to correlate event activity across systems. This is called “event correlation.” Event correlation is a complex exercise that, while still technically possible to do manually, generally requires the use of third-party tools and solutions to be done effectively. Additionally, third-party tools and solutions may employ other methods, such as the use of signatures or automated heuristic analysis mechanisms to help confirm the occurrence of abnormal or malicious activities. For more information on these types of tools, please refer to Appendix B.

5.6 Respond to Incidents

When event activity is confirmed to be—or associated with—known malicious activity, appropriate countermeasures need to be deployed as quickly as possible to isolate or stop the activity and address any potential residual effects or risks presented by the occurrence of that activity. Rapid response requires that procedures be defined in advance of the incident such that any delays in deploying appropriate countermeasures can be minimized. As previously discussed, PCI DSS Requirement 12.10.5 requires that organizations implement incident response procedures to respond to alerts from security-monitoring systems. PCI DSS Requirement 12.10 covers the need to have such incident response procedures formally documented (in advance), and Requirement 12.5.3 requires appropriate personnel be assigned responsibility for ensuring the timely and effective handling of security incidents.

5.7 Collect and Analyze Incident Data

Incident feedback is a critical component of an effective log-monitoring program. Any lessons learned from the handling and response to information security events should be formally documented and used as input when it is time to conduct periodic review of incident response processes. In fact, PCI DSS Requirement 12.10.6 explicitly requires such improvements be made for incident response processes, and the same should be made applicable to other log monitoring processes as well. Any gaps or areas of improvements should be identified and recommendations for addressing those gaps should be documented.

Metrics should also be developed to qualify and/or quantify program performance. Some sample metrics that could be used to describe the effectiveness of the log-monitoring program include the following:

▪ % systems with logging enabled	▪ Avg # of confirmed events per day/week/month
▪ % systems with alerting enabled	▪ # of unknown events per day/week/month
▪ Top 10 events detected	▪ # of baseline violations per day/week/month
▪ Top 10 systems affected (destination IP)	▪ # of detected events vs. # of alerts generated
▪ Top 10 activity sources (source IP)	▪ # of false positives
▪ Top 10 users associated with confirmed events	▪ # incidents vs. # of resolved incidents
▪ Avg # of alerts per system per day/week/month	▪ Avg response delay (from detection to response) in minutes

5.8 Report on Results

Management must be kept aware of the performance of the log-monitoring program to ensure the program continues to receive the oversight and resources necessary to continue to operate effectively. Many third-party monitoring tools and solutions come pre-packaged with report templates that provide log-monitoring performance data, including metrics like those described in the previous section. Where third-party tools are unavailable, scripts can be written to extract similar information from log data repositories. Reports and/or presentations should be created to describe the program's performance to management using metrics as well as other quantitative and qualitative measures. These reports then become a primary source of input when log-monitoring requirements and use cases are re-evaluated to ensure ongoing alignment with updated risk analysis.

5.9 Perform Periodic Program Reviews

The ongoing effectiveness of any log-monitoring program is contingent on its ability to keep up with changing business risks and needs. Therefore, to keep the log-monitoring program operating effectively and efficiently, log-monitoring requirements should be frequently re-evaluated and take into consideration the latest risk analysis information as well as overall program performance information to identify any potential gaps. It is possible, even likely, that updated risk analysis and log monitoring performance data will uncover new events and activities that require monitoring.

For the same reasons why PCI DSS Requirement 12.1.1 requires annual reviews of security policy, periodic reviews of log-monitoring requirements should also be performed. Security threats and protection mechanisms evolve rapidly. If log-monitoring requirements do not keep up with the evolving risk landscape, detection mechanisms may not pick up on new attacks or events that are potential indicators of new attacks.

Periodic reviews of log-monitoring requirements should be performed using intelligence from numerous sources. One such source is the results from the annual risk assessment. Additional sources of intelligence should include log monitoring program performance metrics describe in earlier sections as well as intelligence information from external sources. Consider leveraging log source vendor documentation, information from vulnerability scans, and other threat analysis information that might be available. Don't forget the existing raw log data you have at your disposal as well. While logs have been filtered to extract information related directly to pre-defined events, you should evaluate other events and anomalies that may exist within your event repository as well as in the raw logs themselves. The use of third-party tools and solutions, particularly those solutions with advanced data analytics and "machine learning" capabilities can greatly help with the performance of periodic program reviews and ongoing threat identification and analysis. Please refer to Appendix B for some examples of solutions with advanced data analytics capabilities.

5.10 Make Updates Where Necessary

As new threats and risks are identified, it is necessary that all or some of the log monitoring program activities be re-performed as need. For example, new threats and risks means that new events, log sources, and log messages may need to be identified. New systems may have come online since requirements were initially defined. Changes in the business operations may in turn result in changes in employee behavior. Consequently, activity baselines, event constraints and processing rules may also need to be updated.

All of these activities can be time consuming. However, it is critical to the ongoing effectiveness of the log-monitoring program that these activities be re-performed as necessary. One of the most important elements of an effective log-monitoring program is its ability to remain relevant and operating effectively. Taking the time to review the program periodically and make necessary changes to keep processes relevant is essential to the ongoing success of the log-monitoring program.

6 Applying Effective Log Monitoring

So far, this document has been focused on optimizing log-monitoring processes in order to increase attack detection capabilities in associating with PCI DSS Requirement 10.6. However, there are other opportunities—beyond attack detection—to leverage effective log-monitoring processes. While suspicious events and activity need to be monitored in near real-time, so should the process and controls that are employed to protect information assets from intruders. If all we were to do were focus on attack-related events, we would be ignoring failures in our information security defenses that likely permitted those attacks to be launched in the first place.

6.1 Business-as-Usual Activities

PCI DSS v3 introduced the concept of business-as-usual (BAU) activities as a strategy for ensuring the ongoing effectiveness of PCI DSS security controls. The primary objective of implementing security into BAU activities is to detect the failure of any critical security controls. Some security controls are systems themselves, such as firewalls, intrusion-detection/intrusion-prevention systems, anti-malware, and access control mechanisms. Other security controls are not systems, and detecting potential failures in those controls isn't as simple as monitoring the up/down status of a system or process.

Fortunately, the ability to track the status and effectiveness of security controls is yet another benefit of an effective log monitoring program. Using the processes and the methods previously described in this document, we are able to ascertain important data and metrics, which can provide clear indication of the operational effectiveness of general information security controls. The table below provides a sample of information that can be obtained from log data to monitor the state and effectiveness of PCI DSS security controls:

Req. No.	Requirement	Potential Verification Method
1.3	Prohibit direct public access between the Internet and any system component in the CDE.	Set up alerts/reports to identify activity between IP addresses located in the CDE and IP address ranges not employed by the organization.
2.2.2	Enable only necessary services, protocols, daemons, etc. as required for the function of the system.	Set up alerts/reports to detect when non-approved services, protocols, and daemons are activated.
3.6.4	Replace cryptographic keys that have reached the end of their cryptoperiod.	Set up alerts/reports to detect cryptographic keys that have exceeded their defined cryptoperiod.
5.2	Ensure that all anti-virus mechanisms are kept current.	Set up alerts/reports to detect outdated anti-virus signatures.
6.2	Install critical security patches within one month of release.	Set up alerts/reports to identify systems that have outdated versions of software.
7.1	Limit access to system components and cardholder data to only those individuals whose job requires such access	Set up alerts/reports to identify when users are assigned to groups with access to cardholder data.

Req. No.	Requirement	Potential Verification Method
8.1.5	Monitor activity of user IDs used by vendors	Set up alerts/reports to identify when vendor user IDs are logged on/off.
9.1.1	Use access control mechanisms to monitor individual physical access to sensitive areas.	Set up alerts to identify when users access sensitive areas.

Over time, the frequency with which violations to the rules described above occur should be as close to zero as possible. However, a more practical (and likely) threshold for acceptability may need to be defined by the organization.

Log-monitoring functions themselves can be "monitored" as well to ensure their ongoing effectiveness. One way to do this is to review system status/performance information. For example, system-level processes associated with logging mechanisms should be monitored (using alerts) to detect whether or not those processes are actively running. This is slightly different than what is required by PCI DSS Requirement 10.2.6. Active monitoring or polling of a process' status (e.g., an active PID) can help detect—in near real-time—when such processes fail or are shut down. Additionally, it may be possible to create and automate "fake" events to check the status of detection and alerting mechanisms. One example of this approach in action is to run a scheduled task to periodically (for example, every 1-5 minutes) generate a log message that can be used to verify event detection and alerting mechanisms are operating effectively. Other methods may include leveraging some of the advance capabilities of third-party tools and solutions, including security information and event management systems and advanced data analytics solutions. Please refer to Appendix B for more information on such tools and solutions.

6.2 Summary

Today's IT environments are becoming increasingly more complex. With more complexity comes greater difficulty in managing and securing the information assets within those environments. Add to it the fact that attacks and attackers are becoming increasingly more adept at gaining unauthorized access to such assets, systems and solutions are increasingly riddled with vulnerabilities that may be used by adversaries to gain access to information assets, and companies are devoting fewer and fewer resources to protect those assets, the effective protection of information assets seems like a daunting—if not impossible—endeavor. Despite these complexities, organizations, legal entities, and the general population expect information assets to be protected from unauthorized disclosure. Therefore, in order to have any hope of achieving some level of "security" for information assets, our security methods and practices for protecting information assets must be as effective and efficient as possible.

In order to ensure assets are adequately protected, security defenses must be actively monitored to ensure those defenses continue to work as expected. Likewise, the information assets themselves must also be monitored to ensure they have not been compromised. Nevertheless, the ease with which these tasks can be performed is also somewhat dependent on the complexity of the IT environment. Therefore, in order to be as effective as possible at monitoring the state of security controls and the overall security of information assets, a well-planned, well-structured approach to monitoring is required.

Effective security monitoring requires three distinct elements: Planning, preparation, and performance. Planning involves the identification of relevant legal and regulatory requirements, the analysis of business and IT risks associated with the environment, and the specification of the general activities and specific systems that should be monitored. The preparation phase involves the analysis of the IT architecture and the development/deployment of a monitoring infrastructure to support the IT architecture. The third and final phase is the establishment and performance of effective monitoring processes in a program of continuous improvement.

A structured program for monitoring security event and status information, when implemented properly, allows the organization to focus its efforts on the risks and activities that are the most concerning to the business. This ensures that security and business objectives remained aligned and that security operations remain relevant. This also helps to ensure resources are utilized in the most effective and efficient manner, security controls operate effectively, and information assets are protected at all times. Without a structured approach to security log monitoring, efforts to protect information assets will remain erratic at best. And as history has shown us, at some point someone or something will overcome or circumvent the security controls in place. Without the necessary mechanisms to detect and respond to such instances or such failures in security controls, the whole point of a defense-in-depth security strategy seems to get lost in the shuffle.

Appendix A: Use Case Example

PCI Failed Logon Attempts Use Case

The organization desires to monitor failed logon attempts on all devices in the cardholder data environment.

Addresses the following PCI DSS requirements:

- 10.2.4 Invalid logical access attempts
- 8.1.6 Limit repeated access attempts by locking out the user ID after no more than six logon attempts.

Other PCI DSS Requirements partially met:

- 10.6.1 Review all security events daily.
- 10.6.2 Review other logs periodically.
- 10.6.3 Follow up on exceptions.

Trigger Scenario:

- Describe the event that will trigger the reviewer or SIEM system that suspicious activity has occurred.

Data Sources:

- PCI perimeter firewalls
- PCI Unix systems
- Main POS application
- Database logon logs

Risk Assessment / Gap Analysis

The strengths and weaknesses of this use case are described here:

Strengths

- We can meet the stated PCI DSS requirements.
- A process to record failed logon attempts is available to audit personnel.
- Account lockouts are recorded and validated.
- A follow-up procedure on exceptions is created.

Weaknesses

- Logon failures are locked out after six attempts so there is no immediate risk to the organization.
- The process to follow up with every account lockout is costly.
- System changes within the CDE are not always communicated and the process may fail without validation.
- The main POS application logs need to be modified for better parsing.

Notifications / Escalations

- For PCI perimeter firewall logon anomalies, escalate to the Network Team.
- For PCI Unix devices, follow up with user first, then escalate to the System Admin Team if necessary.
- For POS application anomalies, follow up with the user first, then escalate to the System Admin Team if necessary.
- For Database failed logon anomalies, follow up with user first, then escalate to the Database Administration Team if necessary.

Metrics

- Record the number of lockout events.
- Record the number of escalations by device / device type.

Acknowledgements

PCI SSC would like to acknowledge the contribution of the Effective Daily Log Monitoring Special Interest Group (SIG) in the preparation of this document. The Effective Daily Log Monitoring SIG consists of representatives from the following organizations:

24 Solutions AB
Accudata Systems
Aeris Secure
Agio, LLC
Allstate Insurance
Amazon.com
American Express
Bank of New Zealand
CIPHER
Citigroup Inc.
Compass IT Compliance, LLC
ControlScan
CVS Caremark
Elavon Merchant Services
Equifax
Espion LTD
Games Workshop Ltd
Gemserv Limited
Getnet Technologia em Captura e Processamento De Transacoes H.U.A.H. S/A
GlobalCollect Services B.V.
Grant Thornton
Heartland Payment Systems
Henry Ford Health System
Hewlett-Packard
Hughes Network Systems, LLC
Intellectual Technology, Inc.
Jet Infosystems
Lattimore, Black Morgan and Cain, PC
MegaPath Inc.
NCC Group
NCI Secured Intelligence
Nestle Nespresso SA
Nettitude Ltd.
nGuard Inc.
North Carolina State University
NTT DATA INTELLILINK Corporation
Optomany, LTD.
Patterson Companies, Inc.
Paymetric Inc.
PCI Security Standards Council
Protiviti
RBC Royal Bank
Research in Motion Ltd.
Secured Net Solutions Inc.
Sense of Security Pty Ltd.
ServerChoice
SIX Payment Services Ltd
Sprint Nextel
Starwood Hotels and Resorts Worldwide, Inc.
Stratica International PTY LTD
Sumo Logic
Sword & Shield Enterprise Security, Inc.
SynerComm, Inc.
SystemExperts Corporation
Telstra
Trustwave
TSYS
TUI Travel Plc
U.S. Bancorp
Venafi
Verizon Wireless
Virtual Inc.
Visa
Vodat International Limited
Walt Disney Company, The
Xpient Solutions LLC

References

Black Hat. (2015). *Black Hat Attendee Survey.* Retrieved from https://www.blackhat.com/docs/us-15/2015-Black-Hat-Attendee-Survey.pdf

Frye, D. (2009). *Effective use Case Modeling for Security Information & Event Management.* Retrieved from SANS Institute: https://www.sans.org/reading-room/whitepapers/auditing/effective-case-modeling-security-information-event-management-33319

Kent, K., & Souppaya, M. (2006). *Guide to Computer Security Log Mangement.* National Institute of Standards and Technology. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf

Payment Card Industry Security Standards Council. (2016). *Payment Card Industry Data Security Standard.* Retrieved from https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-1.pdf

Ponemon Institute. (2015). *Cost of Data Breach Study.* Retrieved from IBM: http://public.dhe.ibm.com/common/ssi/ecm/se/en/sew03053wwen/SEW03053WWEN.PDF

Additional Resources

This document draws from the following additional sources of reference. These sources are recommended as additional guidance on building sustainable security and compliance programs.

2015 PCI Compliance Report. Verizon. Van Oosten, C., Baritchi, A., & van Koten, R. (2015). http://www.verizonenterprise.com/resources/report/rp_pci-report-2015_en_xg.pdf

Critical Security Controls. Center for Internet Security. https://www.cisecurity.org/critical-controls/

Cyber Security Monitoring and Logging Guide. CREST. http://crest-approved.org/wp-content/uploads/Cyber-Security-Monitoring-Guide.pdf

Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management. Chuvakin, A., Schmidt, K., & Phillips, C. (2013). Waltham, MA: Syngress.

OWASP Logging Project. The Open Web Application Security Project (OWASP). https://www.owasp.org/index.php/Category:OWASP_Logging_Project.

OWASP Logging Cheat Sheet. The Open Web Application Security Project (OWASP). https://www.owasp.org/index.php/Logging_Cheat_Sheet.

About the PCI Security Standards Council

The PCI Security Standards Council is an open global forum that is responsible for the development, management, education, and awareness of the PCI Data Security Standard (PCI DSS) and other standards that increase payment data security. Founded in 2006 by the major payment card brands American Express, Discover Financial Services, JCB International, MasterCard Worldwide, and Visa Inc., the Council has over 600 Participating Organizations representing merchants, banks, processors, and vendors worldwide. To learn more about playing a part in securing payment card data globally, please visit: https://www.pcisecuritystandards.org.