PCI-DSS 3.0 AND APPLICATION SECURITY

www.quotium.com

Achieving PCI DSS Compliance with Seeker

This paper discusses PCI DSS and the vital role it plays in building secure software applications. It will focus on specific requirements that deal with the protection and transmission of cardholder data, regular testing of security systems and processes, which are all essential in establishing strong application security. It tackles each requirement and explains how Quotium's Seeker helps achieving compliance.

PCI-DSS 3.0 and Application Security

ACHIEVING PCI DSS COMPLIANCE WITH SEEKER

ABSTRACT

The Payment Card Industry Data Security Standard, commonly referred to as PCI-DSS is a leading standard with which organizations that handle payment data such as credit or debit cards are required to comply.

Defined by the Payment Card Industry Security Standards Council, the standard was created to increase controls around cardholder data to reduce credit card fraud through its exposure. Validation of compliance is done annually by an external Qualified Security Assessor (QSA) that creates a Report on Compliance (ROC) for organizations handling large volumes of transactions, or by Self-Assessment Questionnaire (SAQ) for companies handling smaller volumes.

This paper discusses PCI DSS and the vital role it plays in building secure software applications. It focuses on specific requirements that deal with the protection and transmission of cardholder data, regular testing of security systems and processes, all of which are essential in establishing strong application security.

The paper demonstrates how Quotium Technologies' Seeker helps in achieving PCI DSS compliance. It tackles each requirement and explains how Seeker addresses them.
This paper can provide valuable information regarding PCI DSS compliance to:

- Merchants who develop software applications dealing with customer payments

or

- Software companies who build these applications

PCI DSS – STATISTICS

The Payment Card Industry Data Security Standards (PCI DSS) apply to organizations or merchants who accept customer payments through credit or debit cards.
The research firm, Ponemon Institute, has been able to quantify the cost of cyber-attacks, although the financial cost is only one of many.

In its "2013 Cost of Cyber Crime Study", Ponemon found the average annualized cost of cybercrime (per company) in the US to be at $11.6 million per year. That's about 26% more than it was the previous year.

Parallel studies conducted in Germany, Japan, France, and the United Kingdom revealed average annualized costs (in USD) of approximately $7.56M, $6.73M, $5.19M, and $4.72M, respectively. Although all costs are lower than the US figure, these numbers are still large enough to cause significant damage to any business.

By complying with PCI DSS, you will be able to strengthen your defenses, eliminate vulnerabilities, and significantly reduce the chances of a data breach. In fact, you shouldn't comply with PCI DSS just for the sake of compliance; rather, you should comply because it is critical for your business.

PCI DSS – APPLICATION SECURITY

The success and impact of a cyber-attack largely depends on how secure are the organization software applications. When applications have serious vulnerabilities, a cyber-attack will easily succeed and its impact can be considerable.

In the US edition of the Ponemon study mentioned earlier, three of the most expensive types of cyber-attacks, namely malicious insiders, malicious code, and web-based attacks, account for 55% of cyber-crime cost. These three are also the most difficult to resolve, requiring an average of 57.1 days for malicious insiders, 50.3 days for malicious code, and 37.9 days for web-based attacks. The success rate and impact of these particular attacks can be considerably reduced by strong application security.

Because application security plays an important role in countering cyber-attacks, it is given the utmost importance in PCI DSS. Security requirements governing software development are inscribed in major Requirement 6, which charges merchants to “develop and maintain secure systems and applications”.

Other requirements that have implied prescriptions for software development and application security can be found under major Requirements 3, 4, 8, and 11. These requirements specify standards for the protection of stored cardholder data, encryption of cardholder data during transmission, assignment of a unique ID to each person who has computer access, and regular testing of security systems and processes, respectively.

How Seeker Helps You Achieve PCI DSS Compliance

Quotium's Seeker helps you meet PCI DSS requirements with minimal effort. It integrates into the software development lifecycle, identifying vulnerabilities before they become a liability to your organization.

Seeker does this by conducting simulated attacks and analyzing code as it runs in response to those attacks. At the same time, it closely monitors how the code handles sensitive data as the data flows through all application tiers and components. To eliminate false positives and obtain a more accurate assessment of the potential impact and risk to business, the simulated attacks are based on real world exploits.

Seeker is data-centric, meaning all vulnerabilities are assessed in relation to how they affect business critical data. It is therefore the best solution to comply with requirements that concern application data security.

Let us now take a closer look at PCI DSS requirements and discuss how Seeker helps meet them.

Requirement 3: Protect stored cardholder data

Requirement Details	Achieving Compliance with Seeker
3.1 – Keep cardholder data storage to a minimum by implementing data retention and disposal policies, procedures and processes that include at least the following for all cardholder data (CHD) storage • Limiting data storage amount and retention time to that which is required for legal, regulatory, and business requirements • Processes for secure deletion of data when no longer needed • Specific retention requirements for cardholder data • A quarterly process for identifying and securely deleting stored cardholder data that exceeds defined retention.	In order to implement data retention and disposal policies effectively, it is first needed to identify what data needs to be retained or disposed and where these data are located This is no easy task considering the volume of data many organizations handle every day. Seeker automatically identifies payment card information. If further customization is needed, Seeker allows the configuration of user-defined sensitive data. It then uses this knowledge during runtime to monitor the application, seek out the data in question, and track its flow. This makes it easier to know where payment card data is stored and whether data retention and disposal policies are violated.
3.2 - Do not store sensitive authentication data after authorization (even if encrypted). If sensitive authentication data is received, render all data unrecoverable upon completion of the authorization process. **3.2.1- Do not store the full contents of any track** **3.2.2- Do not store the card verification code or value (three-digit or four-digit number printed on the front or back of a payment card)** **3.2.3- Do not store the personal identification number (PIN) or the encrypted PIN block.** **Sensitive authentication data includes:** **Full contents of the magnetic stripe located on the back of the card, which includes the cardholder's name, PAN, expiration date, service code, etc.** **Card verification code or value (CAV2/CVC2/CVV2/ CID)** **Personal identification number (PIN) or PIN block**	PCI DSS storage requirements are not limited to primary storage or data repositories such as databases and flat files (e.g. text files and spreadsheets). It also applies to non-primary storage like backups, audit logs, and exception or troubleshooting logs. Seeker understands how sensitive data is handled by the application - including when stored, transmitted, and in memory. It tracks data throughout its lifespan across application tiers. Seeker's unique technology allows the monitoring of web-service, database and file system calls in the path of sensitive data in order to detect any insecure storage or manipulation.

3.4 – Render PAN unreadable anywhere it is stored (including on portable digital media, backup media, and in logs) by using any of the following approaches: • **One-way hashes based on strong cryptography, (hash must be of the entire PAN)** • **Truncation (hashing cannot be used to replace the truncated segment of PAN)** • **Index tokens and pads (pads must be securely stored)** • **Strong cryptography with associated key-management processes and procedures.**	In addition to being able to identify and monitor PAN and other sensitive authentication data, Seeker can also determine whether they are ever stored unencrypted.

Requirement 4: Encrypt cardholder data over open public networks

PCI-DSS Requirement	Achieving Compliance with Seeker
4.1 Use strong cryptography and security protocols (for example, SSL/TLS, IPSEC, SSH, etc.) to safeguard sensitive cardholder data during transmission over open, public networks, including the following: • **Only trusted keys and certificates are accepted.** • **The protocol in use only supports secure versions or configurations.** • **The encryption strength is appropriate for the encryption methodology in use.**	Seeker identifies sensitive data, monitors its flow, and determines whether it is encrypted or not, regardless of whether the data is at rest or in motion. Seeker alerts to the lack of SSL, but it also alerts specifically to payment card information being transmitted insecurely.

Requirement 6: Develop and maintain secure systems and applications

PCI-DSS Requirement	Achieving Compliance with Seeker
6.1 Establish a process to identify security vulnerabilities, using reputable outside sources for security vulnerability information, and assign a risk ranking (for example, as "high," "medium," or "low") to newly discovered security vulnerabilities.	Seeker provides a continuous vulnerability and remediation process across development projects. Vulnerabilities risk is assessed based on a possible attack outcome rather than on its technical nature.

Note: Risk rankings should be based on industry best practices as well as consideration of potential impact. For example, criteria for ranking vulnerabilities may include consideration of the CVSS base score, and/or the classification by the vendor, and/or type of systems affected.	Vulnerabilities are prioritized according to the risk they pose on user data and classified by industry best practices and standards (OWASP Top10, SANS/CWE, PCI DSS and more) In addition, the user can define internal organizational policies and have Seeker check compliance for those as well.
6.3 Develop internal and external software applications (including web-based administrative access to applications) securely, as follows: • **In accordance with PCI DSS (for example, secure authentication and logging)** • **Based on industry standards and/or best practices.** • **Incorporating information security throughout the software-development life cycle** **Note: this applies to all software developed internally as well as bespoke or custom software developed by a third party.**	Seeker does not need access to the source code to assess its security. It allows organizations to test any internal and outsourced software including third party components integrated into the code. This facilitates testing of open source libraries, components developed by external vendors, off-the-shelf products and more. Whether a component is 3rd party or not is entirely transparent to Seeker. Seeker knows how to differentiate between vulnerabilities in user code and 3rd party code, and proposes remediation to secure their interfaces. It helps to validate software according to best practices and industry standards.
6.3.1 Remove development, test and/or custom application accounts, user IDs, and passwords before applications become active or are released to customers	Seeker alerts the user when it identifies sensitive information such as hardcoded application passwords in database or files
6.3.2 Review custom code prior to release to production or customers in order to identify any potential coding vulnerability (using either manual or automated processes) to include at least the following: • **Code changes are reviewed by individuals other than the originating code author, and by individuals knowledgeable about code-review techniques and secure coding practices.** • **Code reviews ensure code is developed according to secure coding guidelines** • **Appropriate corrections are implemented prior to release.** • **Code-review results are reviewed and**	Seeker integrates into existing R&D methodology and tools, so that every time the code changes, it can carry out security assessment of the changes. Seeker automatically: • Verifies findings by exploitation to eliminate false positives. • Prioritizes vulnerabilities according to their impact and damage potential. This means results are accurate and unbiased. There is no need for a human to go over all results, Seeker does that automatically for the user Detailed remediation instructions are provided to secure the code prior to release (line of code, path through the applications, corrections to apply, videos

approved by management prior to release. **Note: This requirement for code reviews applies to all custom code (both internal and public-facing), as part of the system development life cycle.**	of exploitations of flaws on the tested application)
6.4.5.3 Functionality testing to verify that the change does not adversely impact the security of the system.	With Seeker, developers and testers have security testing in the same place they have functional testing, both to initiate testing and to receive testing results Seeker integrates with automatic testing frameworks like selenium. The security use cases can be mapped to the functional test scenarios.
6.5 Address common coding vulnerabilities in software-development processes as follows: • **Train developers in secure coding techniques, including how to avoid common coding vulnerabilities, and understanding how sensitive data is handled in memory.** • **Develop applications based on secure coding guidelines.** **Note: The vulnerabilities listed at 6.5.1 through 6.5.10 were current with industry best practices when this version of PCI DSS was published. However, as industry best practices for vulnerability management are updated (for example, the OWASP Guide, SANS CWE Top 25, CERT Secure Coding, etc.), the current best practices must be used for these requirements**	With Seeker, developers are trained on the consequences of their insecure coding. On their preferred bug tracking tools they will have access - for each vulnerability reported - to: • The exact location of the vulnerability (including the tier on which the code is found, the code file, and the actual lines of code). • Detailed context-based remediation instructions to guide developers in correcting the vulnerability. • Step-by-step instructions and a video clip showing exploitation of the detected vulnerability so developers can understand and replay the attack themselves. Seeker can be an integral part of security awareness & training processes across teams.
6.5.1 Injection flaws, particularly SQL injection. Also consider OS Command Injection, LDAP and XPath injection flaws as well as other injection flaws	Seeker's ability to spot vulnerabilities that might lead to an SQL injection isn't based on theory or speculation. Seeker actually monitors the application during runtime and observes the malicious input as it traverses the application and arrives at the data layer. Seeker sees the internals of the application during run-time for accurate, false positive free detection. This applies also to LDAP queries, LDAP, XPATH and more. Seeker tracks data throughout application components and tiers and monitors these data as they arrive at the database, directory or file repository calls. Seeker then attempts to exploit this

	access to verify that it could actually be exploited by an attacker.
6.5.2 Buffer overflows	Seeker identifies vulnerabilities by observing code, and memory in response to simulated attack.
6.5.3 Insecure cryptographic storage	Seeker monitors data flow during runtime and reports precisely where information is stored unencrypted. This includes databases, file repositories, debug information, and other repositories.
6.5.4 Insecure communications	Seeker can tell whether and where data are transmitted as clear text, so you will know exactly where data-in-motion encryption is needed.
6.5.5 Improper error handling	Applications sometimes inadvertently leak confidential information through error messages. These leakages may include security configurations, internal workings, or payment card data. Because Seeker can detect a variety of built-in and user-defined sensitive information types, it can check error messages to see if any sensitive information appears there.
6.5.6 All "high risk" vulnerabilities identified in the vulnerability identification process (as defined in PCI DSS Requirement 6.1).	Seeker accurately assesses the impact and classification of each vulnerability's corresponding risk through simulated exploits and data analysis. This feature makes Seeker an invaluable tool in risk-ranking activities and, consequently, in identifying high risk vulnerabilities.
6.5.7 Cross-site scripting (XSS)	XSS allows attackers to execute scripts on a victim's browser and enables them to hijack the user's sessions, alter websites, distribute worms, and perform a host of other malicious activities. Seeker has a unique JavaScript and VBScript analysis engine which identifies cross site scripting and verifies that they an be exploited by using simulated attacks. In addition, by analyzing data, Seeker is able to provide unique insight in testing for Persistent Cross Site Scripting.
6.5.8 Improper access control (such as insecure direct object references, failure to restrict URL access, directory traversal, and failure to restrict	Attackers take advantage of direct object references to gain unauthorized access to other objects. Direct object references are created when developers unwittingly expose references to internal

user access to functions).	implementation objects such as files, directories, keys, or database records through URLs or form parameters. By identifying and tracking data in the system Seeker identifies whether any references affect data and by modifying them if it is possible for an attacker to access privileged information.
6.5.9 Cross-site request forgery (CSRF)	Using CSRF, an attacker can take advantage of a victim's browser by forcing it to automatically send a malicious pre-authenticated request to a web application while the legitimate user is logged on. Seeker detects CSRF vulnerabilities which have an actual impact on application operations (for example vulnerabilities which allow file writing operations, or reading from database tables), and only operations which pose a real threat are then reported to the user.
6.5.10 Broken authentication and session Management	Seeker verifies whether identification and authentication management best practices are in place. It also monitors the length of an idle session and determines whether the idle session limit has been violated.
6.6 For public-facing web applications, address new threats and vulnerabilities on an ongoing basis and ensure these applications are protected against known attacks by either of the following methods: **Reviewing public-facing web applications via manual or automated application vulnerability security assessment tools or methods, at least annually and after any changes.** **Installing a web-application firewall in front of public-facing web applications.**	Web-facing applications are exposed to ongoing threats and can be under attack any time. These attacks often succeed because of insecure coding practices. A regular review on these applications is therefore crucial in preventing attacks from succeeding. Seeker applies a new and highly effective approach of Runtime Code & Data analysis Seeker integrates seamlessly into the software development processes. It becomes part of existing workflow and introduces application security testing as part of ongoing processes. Seeker tracks security flaws at each step of development as well as at every release of the product.

Requirement 8: Assign a unique ID to each person with computer access

PCI-DSS Requirement	Achieving Compliance with Seeker
8.2.1 Using strong cryptography, render all authentication credentials (such as passwords/phrases) unreadable during transmission and storage on all system components.	User passwords are sometimes stored in a database or transmitted over the network as clear text. In these situations, they can be easily obtained by anyone who can penetrate the database or intercept the transmission. Seeker identifies many kinds of sensitive information, including authentication data-like passwords. It can also determine whether the information is encrypted. When passwords are detected, Seeker tracks their flow and maps areas where they fail to be protected by strong encryption.
8.5 Passwords/phrases must meet the following: • **Require a minimum length of at least seven characters.** • **Contain both numeric and alphabetic characters.** • **Alternatively, the passwords/phrases must have complexity and strength at least equivalent to the parameters specified above.**	Seeker verifies whether identification and authentication management best practices are in place. Seeker reports when a weak password policy is being implemented or whether the system accepts weak passwords. Seeker also checks whether the system does not lock user accounts even after a specified number of failed login attempts has been exceeded. It also monitors the length of an idle session and determines whether the idle session limit has been violated.

Requirement 11: Regularly test security systems and processes

PCI-DSS Requirement	Achieving Compliance with Seeker
11.2.1 Perform quarterly internal vulnerability scans and rescans as needed, until all "high-risk" vulnerabilities (as identified in Requirement 6.1) are resolved. Scans must be performed by qualified personnel.	Seeker pioneered IAST (Interactive Application Security Testing), a great new technology that uses a combination of profiling and debugging techniques to analyze how the application behaves under attack. Seeker launches simulated attacks from an attacker point of view and then looks at the application from the inside, observing code, memory and data flow. This allows Seeker to perform an accurate and comprehensive threat identification. Integrated in the software development lifecycle, Seeker test aplication vulnerabilities not on quarterly basis but at each build and each release.
11.2.3 Perform internal and external scans, and rescans as needed, after any significant change. Scans must be performed by qualified personnel.	
11.3.2 Perform internal penetration testing at least annually and after any significant infrastructure or application upgrade or modification (such as an operating system upgrade, a sub-network added to the environment, or a web server added to the environment).	Seeker is like an automated penetration testing tool for applications. From the hacker's point of view to the code, it shows in the same place how a hacker exploits the vulnerability and also where the vulnerability lies in the code.
11.3.3 Exploitable vulnerabilities found during penetration testing are corrected and testing is repeated to verify the corrections	Each vulnerability found by Seeker is provided with everything required to understand and correct the code quickly.
11.3 Perform external and internal penetration testing at least once a year and after any significant infrastructure or application upgrade or modification.	The purpose of conducting a penetration test on a particular environment is to simulate a real world attack scenario and determine the extent by which the attack can penetrate into that environment. Seeker conducts simulated attacks and then analyzes code during run-time to learn how the application responds to these attacks. Furthermore, Seeker integrates into existing R&D methodology, so that every time the code changes, it can carry out the appropriate tests.

CONCLUSION

PCI DSS offers extensive guidance in achieving strong application security. However, it shouldn't be considered the ultimate yardstick. Meaning, even if you have fully complied with all its requirements, that still wouldn't guarantee a fully impenetrable system. Cyber criminals always come up with new kinds of attacks. Application security undertakings should therefore be an ongoing process.

Seeker can play a vital role in that process. As demonstrated throughout this paper, Seeker possesses the necessary elements for achieving PCI DSS compliance. But more importantly, because of Seeker's versatility and ability to closely scrutinize application code and track data flow through all application tiers and components in real-time, it can discover even the most inconspicuous vulnerabilities and help developers build considerably more secure software applications.

About Quotium Technologies

Quotium Technologies specializes in the development of innovative software solutions to guarantee the security and functionality of business-critical applications throughout their life cycle.

Seeker is an Agile security testing solution that allows you to easily automate security as part of existing workflows. Easily integrating with existing software testing processes, Seeker allows developers to efficiently develop secure applications.

For more information www.quotium.com or info@quotium.com