



Establishing a Continuous Process for PCI DSS Compliance

Visa, MasterCard, American Express, and other payment card companies currently require all U.S. merchants accepting credit card payments to comply with the [Payment Card Industry Data Security Standard \(PCI DSS\)](#). A global compliance initiative is targeted for 2010.

The PCI DSS outlines a set of comprehensive requirements to help organizations protect payment card account data from fraud, hacking and various other security vulnerabilities and threats. It requires organizations to not only perform many different tasks, but also to record who performed them and when. Parasoft Concerto, with its business process engine, helps you establish a repeatable process to ensure that the responsible team members are conforming to your organization's prescribed PCI policy.

PCI DSS requirement 6 details how to "develop and maintain secure systems and applications." It promotes a proactive, preventative approach to building security into the application throughout the software development lifecycle (SDLC)—rather than trying to test security vulnerabilities out of the application, one by one.

Parasoft is the industry leader in defect prevention—in fact, we wrote the book on it (*Automated Defect Prevention*, Wiley-IEEE, 2007). With 20+ years of experience helping over half of the Fortune 500 companies incorporate the PCI-mandated practices throughout the SDLC, Parasoft knows what it takes to rapidly bring organizations into compliance with PCI DSS.

Parasoft's PCI DSS Solution significantly reduces the time and cost of achieving PCI compliance by:

- **Delivering the industry's most comprehensive security vulnerability prevention and detection capabilities in an integrated solution:** Parasoft provides out-of-the-box automation of practices essential for achieving PCI DSS 6 compliance, including:
 - **Static analysis**— pattern-based coding standards, data flow analysis, code metrics.
 - **Dynamic analysis**— unit testing, integration testing, functional testing, memory error detection.
 - **Penetration testing**— runtime security policy validation (encryption, authentication, signatures).
 - **Peer code review** (and document review) process automation.
- **Providing out-of-the-box checking for the security issues referenced in PCI DSS requirement 6:** The solution is configured to deliver an instant assessment of compliance with PCI DSS requirement 6 security guidelines across Java, C/C++, .NET, Web language code, and other security-critical application artifacts (e.g., XML configuration files). This enables teams to rapidly assess the level of compliance—without spending time reading the PCI DSS specification and determining how the requirements translate to code.
- **Establishing an automated process that integrates security throughout the SDLC:** Parasoft's automated infrastructure facilitates continued compliance as the application evolves by making compliance to PCI-mandated practices an unobtrusive part of the team's existing workflow.
- **Facilitating issue remediation, not just issue detection:** Each issue detected is prioritized, automatically correlated to the developer who introduced it, then distributed to his or her IDE with direct links to the problematic code. Eventually, developers start writing compliant code as a matter of habit.
- **Delivering extensive reporting for documentation and process improvement:** Our centralized reporting system provides real-time visibility into overall security status and processes

Using Parasoft's integrated solution, organizations not only gain a fast track to PCI DSS 6 compliance, but also establish a process for ensuring that all of the mandated PCI DSS tasks are performed and documented as expected.

Parasoft Concerto Capabilities versus PCI DSS Requirements

PCI DSS Requirement	Parasoft Concerto																										
	Policy Center			Process Center			Project Center			Test Center											Report Center						
	PCI Policy	Policy Management	Policy Best Practices	Business Process Model	Business Rules/Notifications	Process Monitoring	Requirements Management	Task/Iteration Management	Task IDE Integration	Code Analysis - Pattern	Code Analysis - Flow	Coding Metrics	Automated Unit Testing	Functional Unit Testing	Runtime Error Detection	Message/Protocol Testing	Penetration Testing	Web Application Testing	End-To-End Testing	Manual Testing (UAT)	Business Process Testing	Load/Stress Testing	Change Based Testing	PCI Compliance Reporting	Policy Reporting	Process Visibility	Quality Visibility
6.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
6.2	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
6.3	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
6.4	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
6.5	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
6.6	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Parasoft Support for PCI DSS 6

PCI DSS Requirements	Testing Procedures	Parasoft Capabilities
<p>6.1 Ensure that all system components and software have the latest vendor-supplied security patches installed. Install critical security patches within one month of release.</p> <p><i>Note: An organization may consider applying a risk-based approach to prioritize their patch installations.</i></p>	<p>6.1.a For a sample of system components and related software, compare the list of security patches installed on each system to the most recent vendor security patch list, to verify that current vendor patches are installed.</p> <p>6.1.b Examine policies related to security patch installation to verify they require installation of all critical new security patches within one month.</p>	<ul style="list-style-type: none"> • Policy enforcement notifies development to check for updated patches as part of the SDLC. • Development tasks to check and implement patches are visible and reportable within the context of a project or development lifecycle. • Development tasks to check and implement patches are visible and reportable within the context of a project or development lifecycle. • Automated tests or notifications are created for manual verification of security patches.
<p>6.2 Establish a process to identify newly discovered security vulnerabilities (for example, subscribe to alert services freely available on the Internet). Update configuration standards as required by PCI DSS Requirement 2.2 to address new vulnerability issues.</p>	<p>6.2.a Interview responsible personnel to verify that processes are implemented to identify new security vulnerabilities.</p> <p>6.2.b Verify that processes to identify new security vulnerabilities include using outside sources for security vulnerability information and updating the system configuration standards reviewed in Requirement 2.2 as new vulnerability issues are found.</p>	<ul style="list-style-type: none"> • Automated business processes and workflow allow for human tasks to be executed per your organization's defined policy (or PCI-recommended guidelines). • Automated business processes and workflow allow for human tasks to be executed per your organization's defined policy (or PCI-recommended guidelines).
<p>6.3 Develop software applications in accordance with PCI DSS (for example, secure authentication and logging) and based on industry best practices, and incorporate information security throughout the software development life cycle. These processes must include the following:</p>	<p>6.3.a Obtain and examine written software development processes to verify that the processes are based on industry standards, security is included throughout the life cycle, and software applications are developed in accordance with PCI DSS.</p> <p>6.3.b From an examination of written software development processes, interviews of software developers, and examination of relevant data (network configuration documentation, production and test data, etc.), verify that:</p>	<ul style="list-style-type: none"> • Workflow and task management allows for the orchestration of required PCI DSS security tasks throughout the SDLC. • Software development processes and tasks are reportable in granular form.
<p>6.3.1 Testing of all security patches, and system and software configuration changes before deployment, including but not limited to the following:</p>	<p>6.3.1 All changes (including patches) are tested before being deployed into production.</p>	<ul style="list-style-type: none"> • Automated tests or notifications are created for manual verification of security patches.
<p>6.3.1.1 Validation of all input (to prevent cross-site scripting, injection</p>	<p>6.3.1.1 Validation of all input (to prevent cross-site scripting, injection flaws, malicious file execution, etc.)</p>	<ul style="list-style-type: none"> • Pre-configured PCI DSS test configurations can be leveraged in conjunction with task and

PCI DSS Requirements	Testing Procedures	Parasoft Capabilities
flaws, malicious file execution, etc.)		
6.3.1.2 Validation of proper error handling	6.3.1.2 Validation of proper error handling	process management to execute security tests through multiple layers of the application; this is accomplished through <ul style="list-style-type: none"> • Penetration testing. • Flow-based static analysis. • Pattern-based static analysis. • Unit/component testing. • Functional/integration testing. • User acceptance testing. • Contextual peer code review.
6.3.1.3 Validation of secure cryptographic storage	6.3.1.3 Validation of secure cryptographic storage	
6.3.1.4 Validation of secure communications	6.3.1.4 Validation of secure communications	
6.3.1.5 Validation of proper role-based access control (RBAC)	6.3.1.5 Validation of proper role-based access control (RBAC)	
6.3.2 Separate development/test and production environments	6.3.2 The development/test environments are separate from the production environment, with access control in place to enforce the separation.	
6.3.3 Separation of duties between development/test and production environments	6.3.3 There is a separation of duties between personnel assigned to the development/test environments and those assigned to the production environment.	<ul style="list-style-type: none"> • Role-based duties can be defined and tracked via workflow and task management.
6.3.4 Production data (live PANs) are not used for testing or development	6.3.4 Production data (live PANs) are not used for testing and development, or are sanitized before use.	<ul style="list-style-type: none"> • Tests can easily parameterized with test data
6.3.5 Removal of test data and accounts before production systems become active	6.3.5 Test data and accounts are removed before a production system becomes active.	<ul style="list-style-type: none"> • Automated task and iteration management ensures granular tasks are not overlooked within the SDLC.
6.3.6 Removal of custom application accounts, user IDs, and passwords before applications become active or are released to customers	6.3.6 Custom application accounts, user IDs and/or passwords are removed before system goes into production or is released to customers.	<ul style="list-style-type: none"> • Automated task and iteration management ensures granular tasks are not overlooked within the SDLC. • Automated unit tests or notifications are created for manual verifications that test and verify for PCI DSS requirements.
6.3.7 Review of custom code prior to release to production or customers in order to identify any potential coding vulnerability <i>Note: This requirement for code reviews</i>	6.3.7.a Obtain and review policies to confirm all custom application code changes for <i>internal applications</i> must be reviewed (either using manual or automated processes), as follows: <ul style="list-style-type: none"> ▪ Code changes are reviewed by individuals other 	<ul style="list-style-type: none"> • Automation and management of the peer code review and workflow—including preparation, notification, and tracking of peer code reviews—address the known shortcomings of this very powerful inspection method. We automatically

PCI DSS Requirements	Testing Procedures	Parasoft Capabilities
<p><i>applies to all custom code (both internal and public-facing), as part of the system development life cycle required by PCI DSS Requirement 6.3. Code reviews can be conducted by knowledgeable internal personnel or third parties. Web applications are also subject to additional controls, if they are public facing, to address ongoing threats and vulnerabilities after implementation, as defined at PCI DSS Requirement 6.6.</i></p>	<p>then the originating code author, and by individuals who are knowledgeable in code review techniques and secure coding practices.</p> <ul style="list-style-type: none"> ▪ Appropriate corrections are implemented prior to release. ▪ Code review results are reviewed and approved by management prior to release. <p>6.3.7.b Obtain and review policies to confirm that all custom application code changes for <i>web applications</i> must be reviewed (using either manual or automated processes) as follows:</p> <ul style="list-style-type: none"> ▪ Code changes are reviewed by individuals other than the originating code author, and by individuals who are knowledgeable in code review techniques and secure coding practices. ▪ Code reviews ensure code is developed according to secure coding guidelines such as the <i>Open Web Security Project Guide</i> (see PCI DSS Requirement 6.5). ▪ Appropriate corrections are implemented prior to release. ▪ Code review results are reviewed and approved by management prior to release. <p>6.3.7.c Select a sample of recent custom application changes and verify that custom application code is reviewed according to 6.3.7a and 6.3.7b above.</p>	<p>identify updated code by scanning the source control system or the local file system, matches the code with designated reviewers, and tracks the progress of each review item until closure. This allows teams to establish a bulletproof review process where all new code gets reviewed and all identified issues are resolved.</p>
<p>6.4 Follow change control procedures for all changes to system components. The procedures must include the following:</p>	<p>6.4.a Obtain and examine company change-control procedures related to implementing security patches and software modifications, and verify that the procedures require items 6.4.1 – 6.4.4 below.</p> <p>6.4.b For a sample of system components and recent changes/security patches, trace those changes back to related change control documentation. For each change examined, perform the following:</p>	<p>• Change-based testing helps teams identify and test only the areas directly related to the most recent modifications.</p>
<p>6.4.1 Documentation of impact</p>	<p>6.4.1 Verify that documentation of customer impact is included in the change control documentation for each sampled change.</p>	<p>• Automated business processes and workflow allow for human tasks to be executed per your organization's defined policy (or PCI-</p>

PCI DSS Requirements	Testing Procedures	Parasoft Capabilities	
6.4.2 Management sign-off by appropriate parties	6.4.2 Verify that management sign-off by appropriate parties is present for each sampled change.	recommended guidelines).	
6.4.3 Testing of operational functionality	6.4.3 Verify that operational functionality testing is performed for each sampled change.		
6.4.4 Back-out procedures	6.4.4 Verify that back-out procedures are prepared for each sampled change		
6.5 Develop all web applications (internal and external, and including web administrative access to application) based on secure coding guidelines such as the <i>Open Web Application Security Project Guide</i> . Cover prevention of common coding vulnerabilities in software development processes, to include the following: <i>Note: The vulnerabilities listed at 6.5.1 through 6.5.10 were current in the OWASP guide when PCI DSS v1.2 was published.</i>	6.5.a Obtain and review software development processes for any web-based applications. Verify that processes require training in secure coding techniques for developers, and are based on guidance such as the OWASP guide (http://www.owasp.org). 6.5.b Interview a sample of developers and obtain evidence that they are knowledgeable in secure coding techniques. 6.5.c Verify that processes are in place to ensure that web applications are not vulnerable to the following:	<ul style="list-style-type: none"> Policy-based static analysis allows rapid definition of security policies, automatically monitors policy compliance, and tests the effectiveness of these policies. 	
6.5.1 Cross-site scripting (XSS)	6.5.1 Cross-site scripting (XSS) (Validate all parameters before inclusion.)		
6.5.2 Injection flaws, particularly SQL injection. Also consider LDAP and Xpath injection flaws as well as other injection flaws.	6.5.2 Injection flaws, particularly SQL injection (Validate input to verify user data cannot modify meaning of commands and queries.)		
6.5.3 Malicious file execution	6.5.3 Malicious file execution (Validate input to verify application does not accept filenames or files from users.)		
6.5.4 Insecure direct object references	6.5.4 Insecure direct object references (Do not expose internal object references to users.)		
6.5.5 Cross-site request forgery (CSRF)	6.5.5 Cross-site request forgery (CSRF) (Do not reply on authorization credentials and tokens automatically submitted by browsers.)		
6.5.6 Information leakage and improper error handling	6.5.6 Information leakage and improper error handling (Do not leak information via error messages or other means.)		
6.5.7 Broken authentication and session management	6.5.7 Broken authentication and session management (Properly authenticate users and protect account		
			<ul style="list-style-type: none"> Multiple, complementary technologies automate a broad spectrum of security validation and verification practices for C/C++, Java, .NET, SOA, Web, and RIA– the most comprehensive in the industry. Practices supported "out-of-the-box" include: <ul style="list-style-type: none"> Static code analysis - Coding standards, data flow, metrics—including preconfigured PCI DSS 6 , OWASP, and CWE / SANS Top 25 configurations. Penetration testing - Message layer and web interface. Runtime analysis - Buffer overflows and other memory errors. Unit testing - Verification of input validation methods. Runtime security policy validation - Validates encryption, authentication, signatures. Although automated analysis can identify many vulnerabilities, it cannot detect instances

PCI DSS Requirements	Testing Procedures	Parasoft Capabilities
<p>6.5.8 Insecure cryptographic storage</p>	<p>credentials and session tokens.)</p> <p>6.5.8 Insecure cryptographic storage (Prevent cryptographic flaws.)</p>	<p>where security controls are not designed properly or implemented as expected. Such high-level issues can only be detected when the human brain analyzes code in the context of its requirements and available test cases. This is facilitated through peer review workflow automation and management.</p>
<p>6.5.9 Insecure communications</p>	<p>6.5.9 Insecure communications (Properly encrypt all authenticated and sensitive communications.)</p>	
<p>6.5.10 Failure to restrict URL access</p>	<p>6.5.10 Failure to restrict URL access (Consistently enforce access control in presentation layer and business logic for all URLs.)</p>	
<p>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 <i>either</i> of the following methods:</p> <ul style="list-style-type: none"> ▪ 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 	<p>6.6 For <i>public-facing</i> web applications, ensure that <i>either</i> one of the following methods are in place as follows:</p> <ul style="list-style-type: none"> ▪ Verify that public-facing web applications are reviewed (using either manual or automated vulnerability security assessment tools or methods), as follows: <ul style="list-style-type: none"> - At least annually - After any changes - By an organization that specializes in application security - That all vulnerabilities are corrected - That the application is re-evaluated after the corrections ▪ Verify that a web-application firewall is in place in front of public-facing web applications to detect and prevent web-based attacks. <p><i>Note: "An organization that specializes in application security" can be either a third-party company or an internal organization, as long as the reviewers specialize in application security and can demonstrate independence from the development team.</i></p>	<ul style="list-style-type: none"> • Automated security assessment can be conducted using multiple, complementary technologies that automate a broad spectrum of security validation and verification practices for C/C++, Java, .NET, SOA, Web, and RIA– the most comprehensive in the industry. Practices supported "out-of-the-box" include: <ul style="list-style-type: none"> • Static code analysis - Coding standards, data flow, metrics—including preconfigured PCI DSS 6, OWASP, and CWE/ SANS Top 25 configurations. • Penetration testing - Message layer and web interface. • Runtime analysis - Buffer overflows and other memory errors. • Unit testing - Verification of input validation methods. • Runtime security policy validation - Validates encryption, authentication, signatures. • Manual review can be conducted peer code review workflow automation and management—including preparation, notification, and tracking. We automatically identify updated code by scanning the source control system or the local file system, match the code with designated reviewers, and track the progress of each review item until closure. This allows teams to establish a bulletproof review process where all new code gets reviewed and all identified issues are resolved.

About Parasoft's Application Security Solution

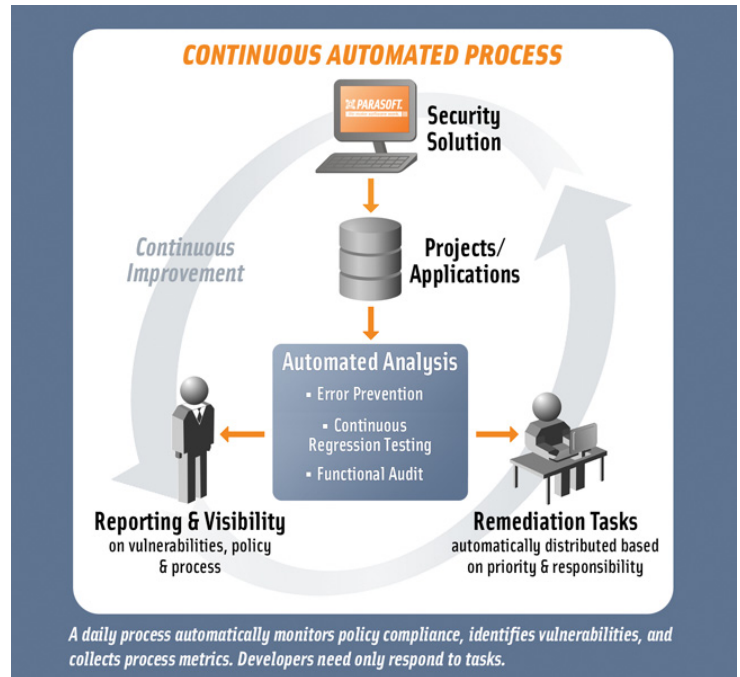
With 20+ years experience helping top organizations implement PCI-mandated practices such as static analysis and code review, Parasoft knows what it takes to ensure adoption and establish a sustainable, repeatable, and predictable process.

To achieve this, Parasoft's Application Security Solution establishes a continuous process that ensures security verification and remediation tasks are not only deployed across every stage of the SDLC, but also ingrained into the team's workflow. The solution automatically monitors policy compliance at all layers of the application stack, identifies vulnerabilities, and collects process metrics.

Developers secure code by simply responding to the reported tasks. To promote rapid remediation, each security issue detected is prioritized, automatically distributed to the developer who introduced it, then distributed to his or her IDE with direct links to the problematic code.

Management gains real-time visibility into overall security status and processes, which allows teams to document improvements as well as determine what additional actions are needed to safeguard security.

For more information, visit http://www.parasoft.com/parasoft_security.



About Parasoft

For 20 years, Parasoft has investigated how and why software errors are introduced into applications. Our solutions leverage this research to deliver quality as a continuous process throughout the SDLC. This promotes strong code foundations, solid functional components, and robust business processes. Whether you are delivering Service-Oriented Architectures (SOA), evolving legacy systems, or improving quality processes—draw on our expertise and award-winning products to increase productivity and the quality of your business applications. For more information visit: <http://www.parasoft.com>.

Contacting Parasoft

USA

101 E. Huntington Drive, 2nd Floor
 Monrovia, CA 91016
 Toll Free: (888) 305-0041
 Tel: (626) 305-0041
 Fax: (626) 305-3036
 Email: info@parasoft.com
 URL: www.parasoft.com

Other Locations

See <http://www.parasoft.com/contacts>