Web Application Penetration Testing for PCI

Michael Hoehl

Web Application Penetration Testing for PCI

GIAC (GWAPT) Gold Certification

Author: Michael Hoehl, mmhoehl@gmail.com
Advisor: Richard Carbone

Accepted: June 19, 2014

Abstract

PCI DSS Requirement 11.3 obligates organizations that process, store, or transport credit card data to implement a methodology for web application penetration testing. This is a recurring commitment—not once and done. This testing must be performed when there is significant change and at least yearly. Merchants as well as payment processors, financial institutions and service providers share this responsibility. This paper proposes a credible method to perform testing to achieve and sustain PCI DSS Requirement 11.3 compliance for web applications.

> Better be despised for too anxious apprehensions, than ruined by too confident security. — Edmund Burke

1. Introduction

The Verizon 2014 Data Breach Investigations Report reported 3,937 total web application related incidents, with 490 confirmed unauthorized data disclosures (Verizon, 2014). These attacks have resulted in negative publicity, revenue loss and significant unplanned costs. The total average liability of these attacks paid by victim organizations increased from $5.4 million in 2012 to $5.9 million in 2013 (Ponemon, 2014). These values do not include indirect costs and lost business an organization suffers as a result of a security incident. For 2013, the Verizon report further reveals that of all incident classifications tracked (e.g., Point-of-Sale System Intrusion, Denial-of-Service, Credit Card Skimming, Insider Misuse, Cyber-espionage, etc.), Web Application attacks remain at the top for all data breaches at 35 percent. Within the retail industry, 95 percent of the reported cyber-attacks had a primary motivation of payment card data theft.

Payment Card Industry Data Security Standard (PCI DSS) was created to provide merchants and service providers guidance safeguarding confidential credit card data. The standard is a result of collaboration between multiple credit card companies including VISA, American Express, MasterCard and JCB. PCI DSS integrates several authoritative guidelines including National Institute of Standards and Technology (NIST) and Open Web Application Security Project (OWASP). This comprehensive standard is intended to reduce the risk of data breaches with specific requirements to protect credit card account and sensitive authentication data. The PCI Security Standards Council (PCI SSC) administers the standard and supporting program. Requirements are defined for security management, policies, procedures, network architecture, software design and other critical protective measures.

PCI DSS is not only a prescriptive industry standard, it is also a contractual obligation for merchants and service providers accepting credit cards for payment. Non-compliance affects fees charged by the acquiring bank with every credit card transaction.

Author: Michael Hoehl, mmhoehl@gmail.com

Several U.S. State breach notification laws (e.g., Nevada, California, etc.) now include in scope loss of credit card data. U.S. State personal information protection laws (e.g., Massachusetts 201 CMR 17.00) and European Union Directives include requirements to safeguard credit card data in a PCI DSS compliant manner.

PCI DSS is comprised of 6 Categories (Goals) including 12 Requirements that apply to the Cardholder Data Environment (CDE). The CDE is comprised of people, processes and technologies that store, process or transmit cardholder data or sensitive authentication data (PCISSC, 2013). The CDE determines the scope of controls that must be sustained by the PCI standard. In some cases, the CDE resides within the merchant private network. In other cases, the CDE is part of a service organization offering (e.g., Private Cloud Computing Infrastructure, Multi-tenant Data Center, etc.). However, a merchant is not required to have all their security controls meet the rigor of the PCI DSS requirements—just those security controls that are part of the CDE.

This paper focuses on PCI DSS Requirement 11.3 as it pertains to penetration testing of web applications within the Cardholder Data Environment (CDE). PCI DSS Requirement 11.3 states merchants must implement a methodology for penetration testing which includes various facets. This is examined in detail in Section 2. It is important to note that a vulnerability scan is not the same as a penetration test. They both have distinct purposes and are both required by PCI DSS. Vulnerability testing required by PCI DSS 11.2 has an important role in validating the presence of configuration and defect weaknesses of a system. This information is used by organizations to help prioritize risk mitigation and reduce the surface of attack. Vulnerability testing is common in the reconnaissance, mapping and discovery phases of penetration testing. PCI DSS Requirement 11.3 penetration testing advances beyond these phases and includes exploiting the discovered vulnerabilities to breach the security of a system. The intent of a penetration test is to simulate a real-world attack with the goal of identifying how far an attacker would be able to penetrate into an environment. This allows a merchant or service organization to gain a better understanding of their potential exposure to a persistent attack and develop a strategy to defend against real attacks (PCISSC, 2013).

Author: Michael Hoehl, mmhoehl@gmail.com

Furthermore, PCI DSS distinguishes between network and application layer penetration testing. Application layer penetration testing targets common coding vulnerabilities including input parameter manipulation, buffer overflow, insecure cryptographic key storage, insecure data transfer or storage, improper error handling, authentication and access control errors, path manipulation and business logic flow errors manifesting from insecure software development processes. Application layer vulnerability and penetration testing can reveal weaknesses to attack that traditional network based security solutions (e.g., stateful firewall, network intrusion prevention system, etc.) can neither detect nor prevent. Application layer exploits are a standard instrument in the hacker toolbox. This form of ethical penetration testing is becoming increasingly important to identify the potential for success with this attack vector and possibility for data exfiltration.

Application layer penetration testing (also known as web application penetration testing) is required for publically facing web applications within the CDE. Penetration testing must be performed at least annually and anytime there is a significant infrastructure or application upgrade or modification (for example, new system component installation, addition of a sub-network or addition of a web server). What is deemed "significant" is highly dependent on the configuration of a given environment (PCISSC, 2008). Organizations may elect to use a third-party company or an internal team to perform the penetration test in as long as the tester is independent from the development team and skilled with testing application security. This paper provides organizations a web application penetration testing methodology for PCI that is credible, repeatable and explainable.

IMPORTANT: This document does not provide legal advice. This paper proposes a credible methodology for performing testing to achieve and sustain PCI DSS Requirement 11.3 compliance for web applications. Do not rely exclusively on this document for guidance about your organization's regulatory and contractual requirements. Consult PCI Security Standards Council, legal counsel, certified PCI assessor, credit card companies and acquiring bank for questions regarding PCI compliance obligations for your organization.

Author: Michael Hoehl, mmhoehl@gmail.com

2. Methodology

PCI DSS 3.0 Requirement 11.3 requires merchants and service providers to implement a methodology for penetration testing that includes the following (PCISSC, 2013):

- Is based on industry-accepted penetration testing approaches (for example, NIST SP800-115);
- Includes coverage for the entire CDE perimeter and critical systems;
- Includes testing from both inside and outside the network;
- Includes testing to validate any segmentation and scope-reduction controls;
- Defines application-layer penetration tests to include, at a minimum, the vulnerabilities listed in Requirement 6.5;
- Defines network-layer penetration tests to include components that support network functions as well as operating systems;
- Includes review and consideration of threats and vulnerabilities experienced in the last 12 months;
- Specifies retention of penetration testing results and remediation activities results.

Note that there is no prescriptive technical details on how the software coding vulnerabilities identified in Requirement 6.5 are to be ethically hacked. Furthermore, there is no elaboration on which tools must be used. NIST SP800-115 is offered as a reference, but is not exclusively required. It is a fair assumption to say that other peer-reviewed, industry-accepted web application penetration testing approaches such as Open Web Application Security Project (OWASP), Open Source Security Testing Methodology Manual (OSSTMM) and Penetration Testing Execution Standard (PTES) can be adopted. This approach provides the flexibility to adopt the most current and effective penetration testing methodology without requiring PCI SSC to publish a completely new version of PCI DSS. The key point is that a proven, repeatable and explainable penetration testing approach is required. This paper proposes an application layer-based penetration testing methodology that is a hybrid of NIST SP800-115

Author: Michael Hoehl, mmhoehl@gmail.com

Technical Guide to Information Security Testing and Assessment and SANS SEC 542 *Web App Penetration Testing and Ethical Hacking*.

2.1. Scoping

Prior to performing the penetration test, selecting the team to perform the test and authoring a vendor Statement of Work, the required scope of the penetration test must be determined. The PCI Security Standards Council provides guidance with *PCI DSS Information Supplement: Penetration Testing (PCISSC, 2008)*: "The scope of penetration testing is the cardholder data environment and all systems and networks connected to it. If network segmentation is in place such that the cardholder data environment is isolated from other systems and such segmentation has been verified as part of the PCI DSS assessment, the scope of the penetration test can be limited to the cardholder data environment." If there is uncertainty, a PCI Qualified Security Assessor (external PCI certified auditor) or Internal Security Assessor (internal PCI certified auditor) is the best resource to confirm CDE and penetration test scope.

PCI SSC offers one exemption from a more traditional penetration test scope. Denial of Service (DoS) is intentionally omitted from the testing scope for PCI DSS Requirement 11. Testing intended to cause service interruption should not be included for consideration by the penetration testing team. These vulnerabilities would presumably not lead to compromise of cardholder data (PCISSC, 2008). The bottom line is that the penetration testing scope is for testing unauthorized access to PCI data—not to determine capacity and resiliency of application infrastructure.

2.2. Planning

Managing a successful assessment begins before the hands-on testing starts. Proper planning is essential for the success of every penetration test. NIST SP800-115 recommends several actions prior to actually performing the penetration test including confirming assessment policy, prioritizing and scheduling assessments, selecting the appropriate assessment approach and addressing logistical considerations. A penetration

testing (or assessment) plan provides prescriptive details as to what is being tested, how, when and where. Additional aspects of the plan should include:

- Testing methodology (e.g., social engineering, remote testing, etc.);
- Complexity (e.g., basic, focused, comprehensive, etc.);
- Testing type (Black Box or White Box);
- High-level calendar of events;
- Security incident handling protocol during testing;
- Communication requirements in advance and during testing;
- Authorization requirements prior to performing tests;
- Removal of tools and data after testing;
- Permitted transmission of assessment data through trusted and untrusted environments;
- Safeguarding of test results;
- Confidentiality of disclosure;
- Distribution of findings.

The organizational assessment policy is an important input to the penetration testing plan. An effective assessment policy has specific attributes that include: sponsorship and authorization from senior management; organizational requirements with which assessments must comply; appropriate roles and responsibilities (at a minimum, for those individuals approving and executing assessments); adherence to established methodology; assessment frequency and documentation requirements, including assessment plans and results (NIST, 2008). These are important considerations necessary for authorized and successful execution of the penetration test.

Obtaining proper permission from Management to perform the penetration test is vital. Verbal consent is not adequate. Ideally, authorization should be in writing with signatures from individuals with sufficient level of authority (e.g., CIO, CISO, CFO, etc.). All stakeholders must be informed of the planned assessment schedule, activities and potential impact on operations. Advanced notification to partner vendors is also

advised. In many cases, multiple vendors will be impacted by a penetration test. This includes Managed Security Service Providers (e.g., firewall, IDS, SOC, SIEM, etc.), Data Center Hosting Providers (e.g., SunGard, IBM, etc.), Internet Access Providers (e.g., AT&T, Level 3 Communications, etc.), Cloud Ecommerce Platform Providers (e.g., Digital River, Demandware, GSI Commerce, Magento, etc.) and On-line Payment Processors (e.g., PayPal, AsiaPay, etc.). There might even be a contractual obligation for advanced notification for planned security control testing. Advanced notification also provides the vendor the opportunity to confirm that threat response procedures are working as planned.

There are legal implications associated with hacking. Laws do not necessarily distinguish ethical from unethical hacking and not all countries have the same laws. Therefore, the Legal Department is an important participant in the planning for penetration testing. In addition to the aforementioned test notification requirements, the Legal Department can provide guidance to avoid potential privacy violations and data handling requirements to ensure data confidentiality during testing. In the event there is a serious weakness discovered or evidence of unauthorized access observed, having the Legal Department engaged early would help expedite the necessary organizational response.

All testing has inherent risk. Therefore, Rules of Engagement are required so that the business can continue to run with minimum interruption. Typical Rules of Engagement discussions include key function contacts (emergency and non-emergency), communication protocol, window of time for testing (and schedule for when testing is not permitted), testing targets and boundaries, test locations, test quality control and incident response protocol. The Rules of Engagement provide guidance to the penetration tester to determine when defined activities can be advanced without the need for additional permissions and when additional permission is required. Additional logistics including who, what, where and when should also be discussed. Operations team management must make available a clear channel of communication to the penetration testing team in order to distinguish real threats from control evaluation during the testing period.

Author: Michael Hoehl, mmhoehl@gmail.com

The last planning consideration is to identify how much information about the target is to be provided to the penetration tester prior to execution of testing. This determines the penetration testing type. There are generally two types of testing: "Black Box" and "White Box". For Black Box penetration testing, very little information is provided to the testers in advance. Source code is not intentionally revealed. The assumption is the tester will have the same information that a malicious attacker would have. However, this assumption can be dangerous in that there is no way to determine what actual information has been (un)intentionally revealed to the malicious attacker. Black Box techniques should be used primarily to assess the security of individual high-risk compiled components, interactions between components and interactions between the entire application or application system with its users, other systems and the external environment (NIST, 2008).

The other extreme is White Box penetration testing (also known as Crystal Box). This is a completely open (source code if applicable is made available) and cooperative (developers and system admins can be questioned) test in which the testing team is provided information prior to the actual testing. Typically, this type of penetration testing is an integral part of the Software Development Life Cycle and is performed at significant release or project management milestones (e.g., QA to SIT, SIT to PROD, etc.). The Open Web Application Security Project (OWASP) proposes that penetration testing occur during the deployment phase of their Testing Framework Workflow (OWASP, 2008). White Box techniques tend to be more efficient and cost-effective for finding security issues than Black Box techniques. If it is determined that it would be beneficial for the penetration tester to have prior knowledge, there are several items produced by the other requirements of PCI DSS that generate useful insight for penetration testers (PCISSC, 2008):

- Results from a Qualified Security Assessor (QSA) review or Self-Assessment Questionnaire (SAQ);
- Quarterly testing for the presence of wireless access points (PCI DSS Requirement 11.1);
- A network diagram (PCI DSS Requirement 1.1.2);

Author: Michael Hoehl, mmhoehl@gmail.com

- Results from quarterly external and internal vulnerability scans (PCI DSS Requirement 11.2);
- Results from the last penetration test (PCI DSS Requirement 11.3);
- Annual identification of threats and vulnerabilities resulting in a risk assessment (PCI DSS Requirement 12.1.2);
- Annual review of security policies (policies that need to be updated may identify new risks in an organization) (PCI DSS Requirement 12.1.3).

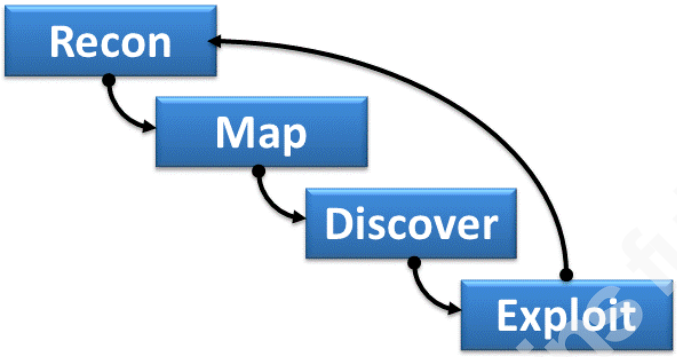
PCI DSS Requirement 11.3 does not explicitly state how much information must be shared with the penetration tester or which testing type must be used. This decision is left to a risk assessment and PCI Qualified Security Assessor (QSA)/Internet Security Assessor (ISA).

2.3. Execution

Now that there is a validated scope, completed assessment plan, established Rules of Engagement, required notification, selected test type and written authorization, hands-on penetration testing may begin.

SANS course *SEC542: Web App Penetration Testing and Ethical Hacking* proposes a four step attack methodology as shown in the following figure:

Figure 1: Four Step Attack Methodology from *SANS SEC542: Web App Penetration Testing and Ethical Hacking.*



The proposed attack methodology is intended to be iterative such that each step is an input to the next step. Therefore, the order of execution is important. In addition, the process is cyclical. Once vulnerability has been successfully exploited, the penetration test advances by returning to the beginning of the attack cycle (Recon) and proceeding to the next target. This approach is useful when there are multiple layers of security controls that the test must pass through. The objective of this approach is to find as many exploitable vulnerabilities as reasonably possible given the resource, time and scope constraints. Whereas an actual malicious attacker might only look for a single vector of attack, the penetration tester is attempting to find all possible vectors of attack. In this way, the surface of attack is reduced as much as possible to actual malicious attackers.

Reconnaissance is the first step of the test and provides the foundation of a credible and efficient attack. The attack target is not engaged directly at this phase. Mature reconnaissance can reveal many weaknesses that become the focus of attack in later phases. Skipping this step might be considered a means of saving time. Unfortunately, this might result in a high number of failed exploit attempts and waste of time later in the testing. Successful ethical (and unethical) hackers focus much of their energy on reconnaissance instead of launching attacks blindly. For a Black Box penetration test, this phase is typically the longest duration. A White Box penetration test reduces the duration of this phase as compared to Black Box since the tester is provided information including network diagrams, platform details, data flows, prior vulnerability scan results and source code.

Author: Michael Hoehl, mmhoehl@gmail.com

Mapping enables the attacker to understand all the facets of the target application and infrastructure. Component relationships, logic flow, software and versions are all examined. For web applications, the site is usually downloaded and inventoried using a spidering tool (also known as web crawler). Popular community source tools that perform spidering include Burp Spider, SprAJAX, WebScarab, Paros and wget. When using a commercial application penetration testing tool (e.g., CoreImpact, AppScan, etc.), this spider functionality is typically built-in. These tools allow the mapping to occur off-line and in one place without disturbing the target system. Lastly, authentication mechanisms and session handling are examined to identify potential vulnerabilities.

Discovery is the third step in the SANS penetration test methodology. At this point, the assessor is actively examining the target application(s) in depth looking for additional information and potential vulnerabilities. According to SANS SEC 542, this phase will focus on finding common applications, user interfaces, information leakage, authentication systems and error messages (SEC542, 2013). Tools and scripts for exploit are prepared during this step. From a hacking perspective, the exploiting has not yet begun. This is still an information gathering and attack preparation phase. However, from a legal perspective, the actions in this phase might be considered a threat and a form of attack. Therefore, the Rules of Engagement must be followed as agreed.

Exploit is the last step in this iterative approach. All the information gathered, all the tools selected and all the scripts prepared are used to exploit flaws that allow security controls to be circumvented. As mentioned earlier, the success of this step is highly dependent on the level of effort devoted to the previous three steps. Poor preparation in advance of this step will typically result in failed exploit attempts by the assessor and create a false sense of security. If a flaw in the application does provide unauthorized access or control, then this methodology proposes returning to the Reconnaissance step to advance the simulated attacks to the next layer of the application. This cyclical approach is recommended so that the penetration tester has the best possible success in pivoting through the application and infrastructure. Remember, this is not a capture the flag exercise. The purpose of this testing is to find as many vulnerabilities as reasonably possible given time and financial constraints.

Author: Michael Hoehl, mmhoehl@gmail.com

PCI DSS Requirements 6.5.1-6.5.10 provides a list of common application vulnerabilities that are to be targeted for penetration testing (PCISSC, 2013).

Table 1: PCI DSS 3.0 Requirements 6.5.1-6.5.10 from PCI Security Standards Council.

6.5.1	Injection flaws, particularly SQL injection. Also, consider OS Command Injection, LDAP and XPath injection flaws as well as other injection flaws.
6.5.2	Buffer overflows
6.5.3	Insecure cryptographic storage
6.5.4	Insecure communications
6.5.5	Improper error handling
6.5.6	All "high risk" vulnerabilities identified in the vulnerability identification process (as defined in PCI DSS Requirement 6.1).
6.5.7	Cross-site scripting (XSS)
6.5.8	Improper access control (such as insecure direct object references, failure to restrict URL access, directory traversal and failure to restrict user access to functions).
6.5.9	Cross-site request forgery (CSRF)
6.5.10	Broken authentication and session management

PCI SSC also states, "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 Testing Guide, SANS CWE Top 25, CERT Secure Coding, etc.), the current best practices must be used for these requirements" (PCISSC, 2013). In other words, these ten stated common application coding vulnerabilities serve as a minimum baseline standard. OWASP Testing Guide v3.0 provides a more comprehensive set of active web application penetration tests in 9 sub-categories for a total of 66 controls. Some of the sub-categories are out of scope (e.g., Denial of Service) while others might not be appropriate for a web application platform (e.g., AJAX). Testers are advised to discuss with the QSA or ISA the flaws that should be included in the testing scope based on current testing practices, recent risk assessment and application platform.

Lastly, several infrastructure components should be carefully considered for eligibility in the testing scope. These include the HTTP Server, Web Application Engine, Database Server, SSL Accelerator\Load Balancer\Proxy, Web Application Firewall,

Database Firewall, Logging Server, Tape Backup Server and Authentication Server. These infrastructure components are necessary for the application and might serve as unintended pivot points that can be used by attackers to eventually exfiltrate credit card account information and sensitive authorization data. Some of the infrastructure components might be in scope for the network penetration testing. Harmonizing the web application penetration testing with the network penetration testing is vital for these infrastructure components so that no attack vector is mistakenly missed from examination.

2.4. Post-Execution

At this point, active penetration testing is complete. However, there remains a lot of work to do. All the testing actions must be documented. This is required so that the testing is repeatable and the findings explainable. As previously mentioned, the penetration tester will pivot through the application and infrastructure. The techniques used at each pivot point must be adequately described so that the tested attack vectors are clear. This includes both successful and unsuccessful testing techniques. Example findings at each pivot point and the data revealed therein should also be collected. Without this supporting documentation, the test might be challenged as inadequate or not credible.

A wiki or similar collaborative content management system is very helpful for organizing and archiving the tester's work. Ideally, the wiki is structured in a manner that aligns with the testing methodology and easily maps to the required reporting format. A common commercial product that serves this purpose is Microsoft SharePoint. Data that might be found in a penetration testing wiki includes the letter of testing authorization, project plan, Rules of Engagement, scope statement, change tickets, inventory of testing tools, scripts, diagrams, screenshots, description of vulnerabilities found, sample test results, executive report, etc. The wiki contains the "recipes for destruction", so appropriate security controls must be put in place to safeguard the confidentiality of this penetration testing data. It would be ideal for a malicious attacker

to discover this wiki with all the heavy work having already been done by the penetration tester.

NIST SP800-115 recommends a three-step approach that organizations can use to translate their findings into actions for improving security. The first step is the analysis of all the findings. Validation is performed during this analysis and risk mitigation options are researched. A popular motto is, "bring me solutions—not just problems". After a thorough analysis and validation of findings, the assessor will be expected to make recommendations for improvement. The recommended actions may be technical (e.g., control configuration change) or procedural (e.g., tracking vendor notifications of new security patches) in nature. Moreover, there might be multiple options to decrease, avoid, assign or accept risk. Management will want to understand these options and their associated investment.

Second, a penetration test report is created and presented. Though NIST SP800-115 elaborates on the value of this report, it does not provide prescriptive guidance for the content of a penetration test report. SANS course *SEC542: Web App Penetration Testing and Ethical Hacking* provides additional guidance with regards to the content of an ideal penetration testing report. SANS proposes five key sections: executive summary, introduction, methodology, findings and conclusions (SEC542, 2013).

The Executive Summary is typically brief (less than two pages) and contains critical findings, remediation recommendations and recommended timelines for change. The Introduction clearly explains the objective, target, scope, timing, restrictions and key participants. The Methodology focuses on explaining the testing approach so that a competent penetration tester or auditor can repeat the testing in the future. This section is necessary to demonstrate to the QSA or ISA the adequacy of the testing necessary for complying with PCI DSS Requirement 11.3. PCI DSS also requires that a follow-up test be performed after vulnerability has been fixed to demonstrate that remediation has been successful. This section is used to guide this quality assurance effort. Depending on the depth of the penetration test, this section is typically 3-10 pages. The Findings section is typically the largest as it contains both the risks and recommendations. Likelihood of

attack, business impact and benchmarking information (industry peers that have suffered from a similar attack) are all included. Because of the amount of information in this section, it is usually categorized by risk level and targets. The Findings section length can vary because of the scope and its findings. The last section is the Conclusion which is similar to the Executive Summary in that it is brief and highlights key risks. It provides the closing remarks that are intended to make the biggest and most necessary impression. Appendices are also typically provided which include a list of the tools used, evidence of findings, chain of evidence (if appropriate) and example code for remediating vulnerabilities.

Lastly, a plan of action and milestones document is required for executing recommended mitigation activities. This document identifies: the tasks needing to be accomplished; the resources required to accomplish the elements of the plan; any milestones in meeting the tasks and scheduled completion dates for the milestones (NIST, 2010). This document will be a living document, reflecting progress while executing recommended mitigation activities. In many cases, the organization will not be able to act on all the findings immediately. Limited resources (i.e., time, money, people) will affect how much and how quickly solutions can be implemented. However, mitigation activities should be advanced using a risk-prioritized approach.

3. Selecting Services and Tools

As previously mentioned, PCI DDS does not recommend or endorse specific penetration testing tools. Furthermore, there is no certification and accreditation program administered by PCI SSC for web application penetration testing services and tools. However, this section may offer some helpful tips for selecting appropriate Services and Tools.

Disclaimer: Any product or service mentioned herein is for informational purposes only, it does not imply recommendation or endorsement by the author nor does it imply that the products mentioned herein are necessarily the best available for the stated purpose.

Author: Michael Hoehl, mmhoehl@gmail.com

3.1. Types of Tools

Many community-sourced and commercial tools are available today for the web application penetration tester. Over the last few years, the tools have been differentiated into two main groups, Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST). Gartner differentiates the two as follows: "Static application security testing (SAST) can be thought of as testing the application from the inside out – by examining its source code, byte code or application binaries for conditions indicative of a security vulnerability. Dynamic application security testing (DAST) can be thought of as testing the application from the outside in – by examining the application in its running state and trying to poke it and prod it in unexpected ways in order to discover security vulnerabilities" (Gartner, 2011). DAST is synonymous with the aforementioned Black Box penetration testing. Examples of DAST community source and commercial products include Paros, W3AF, Inguardians Samurai Web Testing Framework, HP WebInspect, IBM AppScan Enterprise and Core Security CoreImpact (see Appendix C for author provided case study).

For organizations interested in White Box penetration testing, SAST tools are typically used in addition to DAST. Moreover, SAST tools examine source code. These tools are typically used during development or when new code passes through early release management phases. For most commercial products, helpful explanations of the defect and example code corrections are provided to the developer for guidance. Examples of SAST commercial tools include Rough-auditing-tool-for-security, Flawfinder, Yasca, HP Fortify, IBM AppScan Source Edition, Veracode Binary Static Analysis and Checkmarx CxSuite.

Typically SAST and DAST tools include prebuilt tests to evaluate compliance requirements for PCI as well as other regulatory requirements including HIPAA, FISMA, GLBA and SOX. In addition, recent versions of SAST and DAST tools offer integration making collaboration between developers and security even easier.

Many of the aforementioned commercial tools are also available using a Software-as-a-Service (SaaS) model. This model is especially beneficial to an

organization that does not want to commit to the complexity and maintenance necessary for web application testing tools. Examples of vendors offering SaaS-based penetration testing tools include Qualys, CheckMarx CxCloud OnDemand and Veracode Cloud-based platforms. Many of the SaaS offerings are subscription based, making the cost of entry much easier as compared to a perpetual license model. To perform testing from within a private network and CDE firewall segment, the SaaS provider (e.g., Qualys, Veracode, etc) supplies a pre-configured Virtual Appliance. Results are automatically uploaded to SaaS web portal for analysis and reporting.

3.2. Types of Services

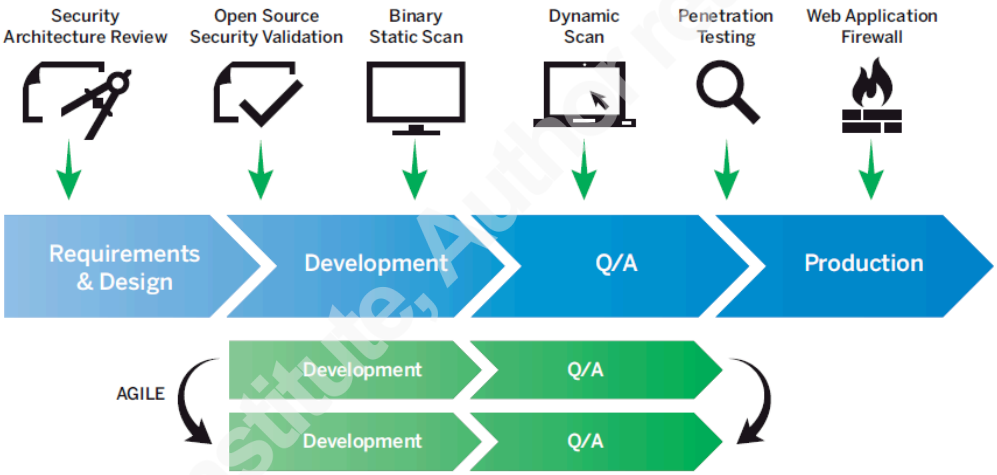
As previously mentioned, PCI DSS permits the use of in-house or vendor-based penetration testers. Although there are a variety of community-sourced and commercial tools available, an organization might elect to engage a vendor to perform the application penetration testing. Unlike the PCI SSC Approved Scanning Vendor (ASV) program, there is no PCI certification and accreditation program administered by the PCI SCC for penetration testing service providers. Many of the ASVs offer penetration testing services, too. Recently, there is a new approach offered by vendors to coordinate penetration testing. Trustwave SpiderLabs Application Security Services, for example, offers on-line penetration testers. Instead of the SaaS model, in which the testing is still performed by in-house staff, testing services are purchased, configured, scheduled and executed entirely on-line. External penetration tests as well as internal penetrations tests within the CDE can be performed (the aforementioned virtual appliance is offered for internal testing). The web portal then stores the results conveniently and securely in a portal that can be reviewed by a QSA, ISA or Acquiring Bank on-demand.

Determining which tools or service to use is a business decision. For some organizations, having the penetration testing performed in-house by employees using community-sourced software is reasonable. Business considerations such as the availability of capital, liability, expertise and resources might dictate that an organization engage service providers to perform the penetration testing. None of these approaches are inherently more secure or compliant from a PCI perspective. However, the timing of

when the tools and services are employed can have an impact on the Software Development Life Cycle and its associated costs.

To gain a better understanding for when the aforementioned security tools and services are optimally used, Financial Services-Information Sharing and Analysis Center Third Party Software Security Working Group offers the following visual guide (FS-ISAC, 2014):

Figure 2: vBSIMM Framework from FS-ISAC *Appropriate Software Security Control Types for Third Party Service and Product Providers* guide.



3.3. Evaluation and Selection Guides

The Web Application Security Consortium (WebAppSec, 2014) and OWASP (OWASP, 2014a and 2014b) offer a comprehensive list of Web Application penetration testing tools for consideration. The list includes both community-sourced and commercial options. The Web Application Security Consortium provides additional guidance for organizations to evaluate application penetration testing tools (WebAppSec, 2009). Not only does it specify the specific scanner functionality that should be evaluated, it provides advice on how to perform a successful "bake-off". Multiple phases are proposed for the evaluation process including preparation, scan testing and results analysis.

For selecting service providers, Information Week journalist Brad Causey provides helpful guidance with his report *Choosing, Managing, and Evaluating A Penetration Testing Service* (InformationWeek, 2013). He advises that the quality of penetration testing services being offered today vary widely. Evaluation criteria of vendor reputation, Security Industry activity, quality of team, financial health and portfolio services offered are suggested when going through the initial discovery process. With this proposed approach, organizations can identify an effective penetration testing provider at a reasonable cost.

4. Integration with Business and Developers

The *OWASP Testing Guide* introduction advises the reader to "Test Early and Test Often". A security logic error is essentially no different from a functional or performance-based logic error. When a bug or flaw is detected early within the Software Development Life Cycle, the fix is typically less effort and lower cost. Rework late in a development project can be costly and materially impact the target completion date. Furthermore, logic errors discovered once the code is made generally available can cause logistic and reputation issues. For these reasons, there is great value integrating security testing with current Business and Developer quality controls.

4.1. Management and Stakeholders

PCI DSS must be sustained—it is not a once a year event. For this reason alone, the "tone at the top" is critical for effective security and continuous compliance. One of the first steps Management should take is to clearly communicate the requirement of penetration testing for achieving business objectives. This communication starts as a formal policy. This empowers developers, project managers, system administrators and security personnel to include security testing as part of their duties. Ideally, the vulnerability management and penetration testing policy is part of the Information Security Policy mandated by PCI DSS Requirement 12.1.

Once the formal policy is in place, metrics should be created to demonstrate that the policy is in place and is tracking risks. A quarterly scorecard can provide insight into

the scope of penetration testing, target dates for testing, results of testing, risk status and remediation target date. It will also show trends over time, revealing to management that information assurance is effective and that risk conditions are improving. Appendix B offers an example scorecard with metrics associated with penetration testing.

These same metrics can be helpful to Management when attempting to determine the total cost of development for a software project. For example, a request for proposal to enhance the web site might have been awarded to the lowest bidder. However, unplanned rework as a result of a penetration test remediation might make the total cost of the project significantly higher. In some cases, the additional unplanned costs might cause the project to exceed the bid of competitors that included code review and other quality controls within their cost of doing business. These metrics are useful to Management (especially Legal, Cost Center Managers and Procurement) for driving down the total cost of application development and liability of vulnerabilities.

4.2. Developers

According to the 2014 Trustwave Global Security Report, 96 percent of applications scanned by Trustwave harbored one or more serious security vulnerabilities (Trustwave, 2014). These vulnerabilities were discovered during vulnerability scanning prior to the penetration testing effort. Clearly, this finding from Trustwave demonstrates the need for security to be integrated into the application development environment.

Integration with Developers begins with security awareness and secure coding practices. PCI DSS Requirement 6.5 states, "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" (PCISSC, 2013). This planned investment is very helpful for avoiding unplanned expenses associated with code rework due to a lack of understanding risk or exploit-based techniques.

Several of the White Box tools and services mentioned in Section 3 of this paper include functionality intended to help developers throughout the multiple phases of the

development lifecycle prior to penetration testing. In addition, many of the commercial penetration testing tools provide example coding techniques to remediate the risks uncovered. Lastly, when evaluating penetration tools and services, be sure to include the development team, as they are a key stakeholder in application security.

4.3. Project Managers and Project Management Office

One of a project manager's primary duties is to manage the project scope. This is accomplished by avoiding activities that have not been approved as part of the project plan. The security team might find, to their chagrin, that penetration testing was not performed because the project manager was not informed early that testing was a necessary activity and in project scope. Furthermore, the project manager might push back on having the penetration testing done as it will affect the project's scope, time and resources. To help ensure that project managers advocate for penetration testing, the security team might want to consider creating a template-based collection of activities with dependencies, level of effort, duration and resources defined. The intention here is to make it easy for the project manager to include these activities as part of the initial approved project plan. For organizations with a Project Management Office (PMO), these activities can become part of the application development project-planning standard. The PMO will audit and examine projects for compliance with the project standard, providing a form of security assurance. An example collection of penetration testing activities is provided by the author in Appendix A: PMO template for PCI Requirement 11.3 Application Testing.

4.4. System Administrators and Change Management

Assuming an organization follows common change management standards, vulnerable code cannot be promoted to production without the assistance of system administrators. ITIL, ISO27002 and PCI DSS Requirement 6.4 require separation of development, testing and operational environments and advise against having developers perform code changes on product systems. All code promotion into production is to be performed by system administrators. With this in mind, integrating penetration testing controls with system administrators can be very effective. As part of the defined release

management and change management standard operating procedures, system administrators are ideal candidates for ensuring penetration testing and necessary remediation has been completed prior to code promotion. System administrators are not performing the penetration tests, just providing the assurance that the security testing was appropriately completed.

5. Remediation

Once penetration testing is done, risk analysis completed, the report on findings presented and the plan of action and milestones approved, the last step is to fix the vulnerabilities. PCI DSS Requirement 11.3.3 requires that "Exploitable vulnerabilities found during penetration testing are corrected and testing is repeated to verify the corrections" (PCISSC, 2013). Therefore, to be compliant with PCI DSS, the organization must not only test for exploitable vulnerabilities, they must demonstrate that the risks were remediated.

In some cases, compensating controls might be in-place for consideration to effectively reduce risk and achieve compliance. PCI DSS requires compensating controls satisfy the following criteria (PCISSC, 2013):

1. Meet the intent and rigor of the original PCI DSS requirement.
2. Provide a similar level of defense as the original PCI DSS requirement, such that the compensating control sufficiently offsets the risk that the original PCI DSS requirement was designed to defend against (see Navigating PCI DSS for the intent of each PCI DSS requirement).
3. Be "above and beyond" other PCI DSS requirements (simply being in compliance with other PCI DSS requirements is not a compensating control).
4. Be commensurate with the additional risk imposed by not adhering to the PCI DSS requirement.

PCI SSC provides a form for documenting compensating controls in Appendix B of the PCI DSS Requirements and Security Assessment Procedures. This must be included with yearly PCI validation reporting.

For those code and control deficiencies that do not have a compensating control, then remediation is required. A Remediation Plan is typically technical and clearly describes the code and control changes that must be performed, the functional managers accountable for the change, resources responsible for performing the change, activities, level of effort, verification methodology and target completion date. This document is intended to be far more prescriptive than either the plan of action or milestones document. The purpose of the Remediation Plan is to ensure that all parties clearly understand in detail what must be performed in order to return to a secure and compliant state. Depending on the penetration testing findings, there might be multiple Remediation Plans for both internal and external (vendors) teams.

6. Conclusion

Most applications change to reflect new business needs and customer requirements. With these changes comes the risk of logic errors and vulnerabilities. PCI DSS Requirement 11.3 obligates organizations to have in-place a methodology for the penetration testing of applications that process, store or transport credit card data. This requirement is intended to identify the vulnerabilities in a manner that simulates a real world, malicious cyber-attack.

Organizations will typically fail to meet the rigors of PCI DSS Requirement 11.3 if compliance is considered simply a hands-on exercise using an open-source application-scanning tool. Tools are just a small part of the web application security testing required for PCI yearly validation. The keys to success with Requirement 11.3 are:

- Implement an Information Security Policy sponsored by Executive Leadership that requires the penetration testing and obtain written authorization to conduct the test;

- Partner with QSA/ISA to properly scope the penetration test;
- Select tools, services and processes that integrate well with all stakeholders (e.g., Security, Developers, Management, PMO, etc.);
- Plan in advance of the penetration test while considering the legal implications, Rules of Engagement and technique (Black Box or White Box);
- Execute the penetration using an industry-accepted penetration testing methodology (e.g., NIST SP800-115, SANS SEC542, OWASP, etc.);
- Ensure the penetration report contains an Executive Summary, Introduction, Methodology, Findings and Conclusion;
- Build a Plan of Action and Milestones document based on reported findings to guide remediation efforts and metrics to track progress.

By incorporating these keys to success, an organization can sustain PCI DSS Requirement 11.3 in a proven, explainable and repeatable manner.

Author: Michael Hoehl, mmhoehl@gmail.com

References

Chuvakin A., Williams, B. (2009, December 1). *PCI Compliance: Understand and Implement Effective PCI Data Security Standard Compliance.* Waltham, MA: Syngress.

Faircloth, Jeremy. (2011). *Penetration Tester's Open Source Toolkit, 3rd Edition*. Waltham, MA: Syngress.

FS-ISAC. (2014). *Third Party Software Security Working Group - Appropriate Software Security Control Types for Third Party Service and Product Providers*. Retrieved from http://www.veracode.com/sites/default/files/Resources/Whitepapers/fs-isac-working-group-whitepaper.pdf?mkt_tok=3RkMMJWWfF9wsRolv63JZKXonjHpfsX77usqX6G3lMI%2F0ER3fOvrPUfGjI4FS8JgI%2BSLDwEYGJlv6SgFTbnFMbprzbgPUhA%3D.

Gartner. (2011). *Static or Dynamic Application Security Testing? Both!*. Retrieved from http://blogs.gartner.com/neil_macdonald/2011/01/19/static-or-dynamic-application-security-testing-both/.

InformationWeek. (2013). *Choosing, Managing and Evaluating A Penetration Testing Service*. Retrieved from http://reports.informationweek.com/abstract/21/11475/Security/Strategy:-Choosing,-Managing-and-Evaluating-A-Penetration-Testing-Service.html.

ISECOM. (2010). Open Source Security Testing Methodology Manual. Retrieved from http://www.isecom.org/research/osstmm.html.

NIST. (2008). *SP 800-64 Revision 1, Security Considerations in the Information System Development Life Cycle*. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-64-Rev2/SP800-64-Revision2.pdf.

NIST. (2008). *Special Publishing 500-115: Technical Guide to Information Security Testing and Assessment*. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf.

Author: Michael Hoehl, mmhoehl@gmail.com

NIST. (2008). *Special Publishing 500-269: Software Assurance Tools: Web Application Security Scanner Functional Specification Version 1.0.* Retrieved from http://samate.nist.gov/docs/webapp_scanner_spec_sp500-269.pdf.

NIST. (2010). *Special Publishing 800-37: Guide for Applying the Risk Management Framework to Federal Information Systems.* Retrieved from http://csrc.nist.gov/publications/nistpubs/800-37-rev1/sp800-37-rev1-final.pdfhttp://csrc.nist.gov/publications/nistpubs/800-37-rev1/sp800-37-rev1-final.pdf.

NIST. (2013). *Web Application Vulnerability Scanners*. Retrieved from http://samate.nist.gov/index.php/Web_Application_Vulnerability_Scanners.html.

OWASP. (2008). *Open Web Application Security Project Testing Guide version 3*. Retrieved from https://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf.

OWASP. (2014a). *Open Web Application Security Project - Phoenix Project.* Retrieved from https://www.owasp.org/index.php/Phoenix/Tools.

OWASP. (2014b). *Open Web Application Security Project - Category:Vulnerability Scanning Tools.* Retrieved from https://www.owasp.org/index.php/Category:Vulnerability_Scanning_Tools.

PCISSC. (2008). *Information Supplement: Penetration Testing.* Retrieved from https://www.pcisecuritystandards.org/documents/information_supplement_11.3.pdf.

PCISSC. (2010). *Navigating the PCI DSS v2.0.* Retrieved from https://www.pcisecuritystandards.org/security_standards/documents.php.

PCISSC. (2013). *PCI DSS 2.0 Cloud Computing Guidelines*. Retrieved from https://www.pcisecuritystandards.org/security_standards/documents.php.

PCISSC. (2013). *PCI DSS 2.0 eCommerce Guidelines*. Retrieved from https://www.pcisecuritystandards.org/security_standards/documents.php.

PCISSC. (2013). *PCI DSS v3.0.* Retrieved from https://www.pcisecuritystandards.org/security_standards/documents.php.

Author: Michael Hoehl, mmhoehl@gmail.com

Ponemon. (2014). *2014 Cost of Data Breach Study - United States*. Retrieved from http://public.dhe.ibm.com/common/ssi/ecm/en/sel03017usen/SEL03017USEN.PDF.

PTES. (2014). *PTES Technical Guidelines*. Retrieved from http://www.pentest-standard.org/index.php/Main_Page.

Scambray, Joel. Etal. (2012). *Hacking Exposed (Web Applications), 3rd Edition*. New York, NY:McGraw-Hill.

SEC542. (2013). *SANS SEC542: Web App Penetration Testing and Ethical Hacking.* Bethesda, MD: SANS Institute.

Tiller, James. (2012). CISO's Guide to Penetration Testing: A Framework to Plan, Manage, and Maximize Benefits. Auerbach Publications.

Trustwave. (2014). 2014 Trustwave Global Security Report. Retrieved from http://www2.trustwave.com/rs/trustwave/images/2014_Trustwave_Global_Security_Report.pdf?aliId=19867330.

Verizon. (2014). *2014 Data Breach Investigations Report.* Retrieved from http://www.verizonenterprise.com/DBIR/2014/.

WebAppSec. (2009). *Web Applications Security Scanner Evaluation Criteria version 1.0.* Retrieved from http://projects.webappsec.org/f/Web+Application+Security+Scanner+Evaluation+Criteria+-+Version+1.0.pdf.

WebAppSec. (2014). *Web Application Security Consortium Web Application Scanner List.* Retrieved from http://projects.webappsec.org/w/page/13246988/Web%20Application%20Security%20Scanner%20List.

Author: Michael Hoehl, mmhoehl@gmail.com

Glossary

Acquiring Bank – bank or entity that a merchant uses to process payment card transactions.

Approved Scanning Vendor (ASV) – entity approved by the PCI Security Standards Council to perform vulnerability scanning. ASV entities are required to follow a specific set of scanning and reporting criteria set forth by the PCI Security Standards Council.

Black Box Penetration Testing – type of penetration testing in which an assessor evaluates security controls by simulating a real attack targeting an application. Black Box techniques assess the security of individual high-risk compiled components; interactions between components and interactions between the entire application or application system with its users, other systems and the external environment.

Cardholder Data Environment (CDE) – people, processes and technologies that store, process or transmit cardholder data or sensitive authentication data.

Dynamic Application Security Testing (DAST) – also known as "Black Box" penetration testing, it examines applications and architecture by simulating attacks that exploit common vulnerabilities (e.g., injection flaws like SQL injection, insecure cryptographic storage, insecure communications, improper error handling, etc.).

Internal Security Assessor (ISA) – internal audit security professional of large merchants, acquiring banks or processors that has attended PCI Standards Security Council training and completed qualification program necessary to perform PCI DSS internal assessments.

Open Source Security Testing Methodology Manual (OSSTMM) – penetration testing guide authored by Peter Herzog and distributed by the Institute for Security and Open Methodologies (ISECOM).

Author: Michael Hoehl, mmhoehl@gmail.com

Open Web Application Security Project (OWASP) – worldwide non-profit charitable organization focused on improving the security of software.

Payment Card Industry Data Security Standard (PCI DSS) – security requirement and assessment standard for organizations that store, process or transmit cardholder data or sensitive authentication data.

PCI Security Standards Council (PCI SSC) – open global forum, launched in 2006, that is responsible for the development, management, education and awareness of the PCI Security Standards, including the Data Security Standard (PCI DSS), Payment Application Data Security Standard (PA-DSS) and PIN Transaction Security (PTS) requirements.

Penetration testing – form of assessment to identify ways of exploiting vulnerabilities to circumvent or defeat the security features of system components. Penetration testing includes network and application testing as well as controls and processes around the networks and applications and occurs from both inside and outside the environment (external testing).

Penetration Testing Execution Standard (PTES) – penetration testing guide authored by Chris Nickerson and a group of information security practitioners from various industries (i.e., Financial Institutions, Service Providers, Security Vendors).

Qualified Security Assessor (QSA) – a security professional that has attended PCI Standards Security Council training and completed the appropriate qualification programs necessary to perform PCI DSS assessments.

Rules of Engagement – defines how penetration testing is to occur so that the business can continue to run with minimum interruption.

Self-Assessment Questionnaire (SAQ) – reporting tool used to document an organization's self-assessment results from an entity's PCI DSS assessment.

Software Development Life Cycle (SDLC) – methodology used from the conception phase through delivery to end-of-life of a software product.

Statement of Work (SOW) – legal instrument that defines activities, deliverables and timeline a vendor must perform to complete services contracted by customer. The SOW usually includes detailed requirements and pricing, with standard regulatory and governance terms and conditions.

Static Application Security Testing (SAST) – also known as "White Box" penetration testing, examines application source code for logic error and omission vulnerabilities. Typically, this form of security assessment is performed during development and QA phases of the Software Development Life Cycle.

Web Application Firewall (WAF) – software or hardware used to enforce a set of security rules to an HTTP conversation between a web application and the client end point.

White Box Penetration Testing – type of penetration testing in which an assessor reviews coding, architecture and build practices to discover vulnerabilities, then tests these vulnerabilities to determine if they are exploitable. White Box assessors, as compared to Black Box assessors, have prior knowledge of the target as a result of analyzing source code, interviewing developers, reviewing prior audits and studying architectural drawings.

Author: Michael Hoehl, mmhoehl@gmail.com

APPENDIX A: Example project template for PCI DSS Requirement 11.3 Application Penetration Testing

1	**Initiate**	
	1.1	Develop Preliminary Project Scope Statement
	1.2	Establish Timeframe for Penetration Testing
	1.3	Identify Executive Sponsor and Stakeholders
	1.4	Create project documents (e.g., Charter, Financial Authorization, etc.)
	1.5	Obtain formal Project Authorization
	A	**MILESTONE - Penetration Testing Project Charter Ratified**
	1.6	Announce Penetration Test
	1.7	Create Project Steering Committee
	1.8	Author Executive Presentation
	1.9	Conduct Project Kick-off Meeting and inform stakeholders
	B	**MILESTONE - Penetration Testing Project Started**
2	**Scope**	
	2.1	Review CDE and testing targets with QSA/ISA
	2.2	Establish Cardholder Data Environment (CDE) Scope
	2.3	Confirm Penetration Test Scope for Project
	2.4	Identify key resources (staff, Service Providers, etc.) that maintain CDE controls
	2.5	Author/Update RACI
	2.6	Review PCI DSS Requirements 6.5 and 11.3 with resources identified in RACI
	2.7	Review legal considerations with Legal and Privacy teams
	2.8	Obtain necessary documentation (Letter of Authorization, contracts, audit reports, SAQ, etc.)
	C	**MILESTONE - Penetration Test Project Scope and Accountabilities Formalized**
3	**Plan**	
	3.1	Determine communication requirements
	3.2	Identify and Engage Penetration Testing Team
	3.3	Establish Document Handling requirements and Book of Evidence
	3.4	Gather Pre-work Artifacts for Penetration Tester
	3.5	Develop Validation Plan and establish testing methodology
	3.6	Review Rules of Engagement (RoE)
	3.7	Identify and engage penetration testing team
	3.8	Obtain formal, written authorization to advance testing
	D	**MILESTONE - Penetration Testing Plan Approved and Key Resources Engaged**
4	**Execute**	
	E	**MILESTONE - Hands-on Penetration Testing Started**
	4.1	Conduct Penetration Testing Team Kick-Off Meeting
	4.2	Finalize Testing Rules of Engagement and Testing Logistics
	4.3	Test and confirm CDE Scope
	4.4	Perform Reconnaissance

Author: Michael Hoehl, mmhoehl@gmail.com

4.5	Perform Mapping
4.6	Perform Discovery
4.7	Execute Exploits
4.8	Document testing actions and findings
F	**MILESTONE - Hands-on Penetration Testing Completed**
5	**Post-Execution**
5.1	Perform initial vulnerability finding(s) assessment and risk review
5.2	Perform Penetration Testing quality control review
5.3	Draft Report on Findings
5.4	Confirm vulnerability findings with accountable teams
5.5	Develop Risk Treatment Plan (Prioritized Plan of Action) to Management
5.6	Present Reports to Sponsor and Stakeholders
G	**MILESTONE - Penetration Testing Results Presented to Management**
5.7	Update PCI Validation Book of Evidence
5.8	Update Security Scorecard
5.9	Develop and Initiate Remediation Plans
H	**MILESTONE - Risk Remediation Started**
6	**Close**
6.1	Advance Data Retention and Destruction Procedures
6.2	Conduct Project Quality Control Review
6.3	Document and discuss Project Lessons Learned
6.4	Review Project Performance and Outcome with PMO
6.5	Formally Release Resources
6.6	Close Project
I	**MILESTONE - Project closed**

APPENDIX B: Example Metrics

"An important part of a good security program is the ability to determine if things are getting better. It is important to track the results of testing engagements and develop metrics that will reveal the application security trends within the organization. These metrics can show if more education and training are required, if there is a particular security mechanism that is not clearly understood by development and if the total number of security related problems being found each month is going down" (OWASP, 2008).

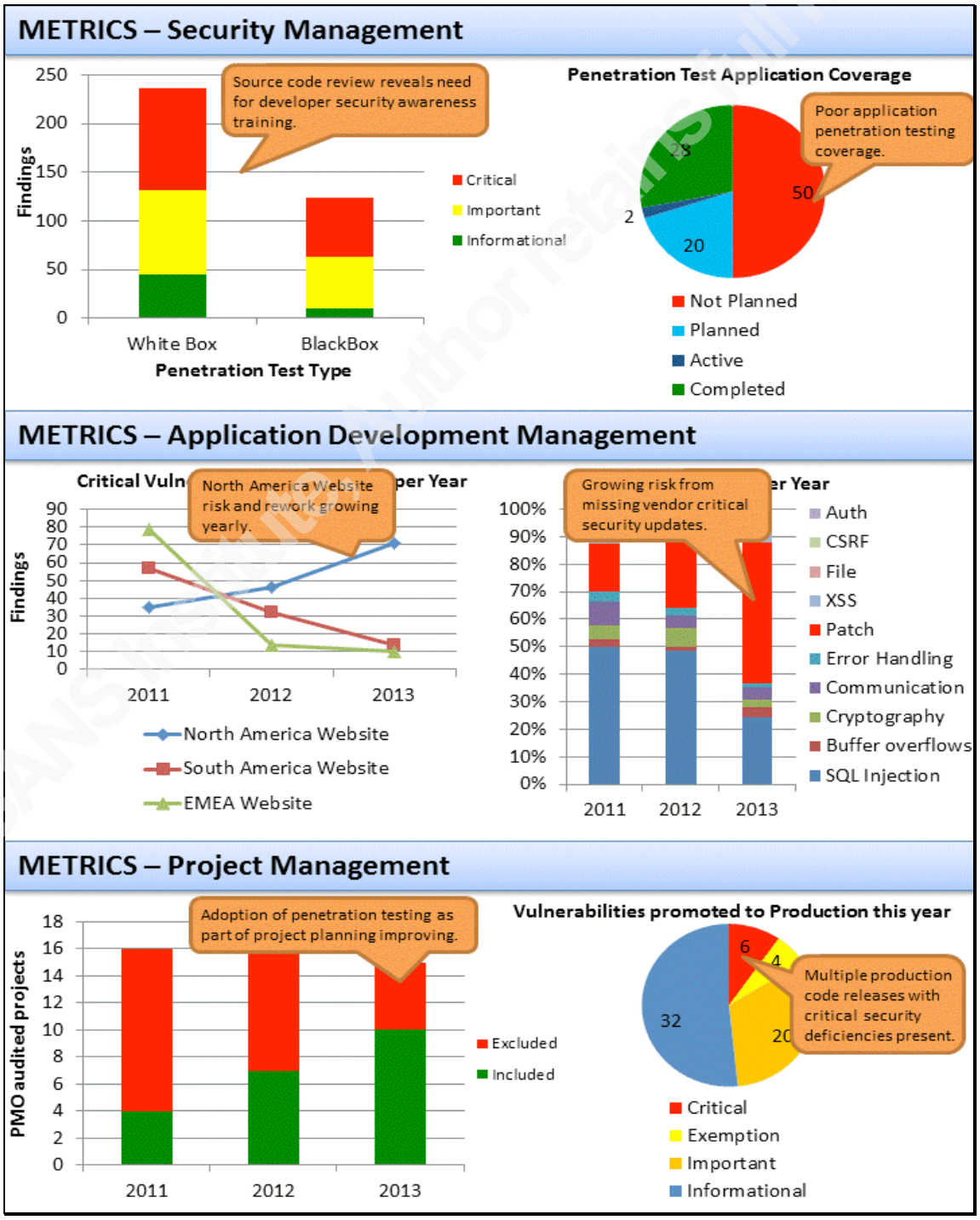
This appendix provides example metrics presented using an executive scorecard for penetration testing, as seen in Figure 3. The metrics are divided into categories reflecting three key stakeholders. The first category is for the Security team and represents them as an internal service provider to the organization. The metrics summarize assessment findings based on risk and the type of penetration test (White Box vs. Black Box). The orange bubbles help to identify key findings from the metrics indicating that investment in developer secure coding training is appropriate. The second metric reveals that a large number of applications are not yet in place for this year's penetration testing.

The second category is intended for the Application Development team. The first metric trends the number of critical vulnerabilities per regional website. The example reveals that the North American website has a growing number of vulnerabilities discovered by penetration testing. The other metric reveals the top vulnerabilities per year, which demonstrates that while vulnerabilities associated with SQL Injection are decreasing (developer secure coding training is working), the number of missing vendor patches is increasing (system administrator security awareness training might be in order).

The third category is intended for the Project Management and Change Management teams. The first metric trended over time demonstrates that project managers are incorporating penetration testing activities into their plans more frequently. This makes it easier for the Security team to perform their services without slowing a

project down. The final metric tracks release management in which code is promoted to production with security vulnerabilities still present. If this continues, a trending graph might be helpful with further data segmentation based on application.

Figure 3: Example Penetration Testing Scorecard.



Author: Michael Hoehl, mmhoehl@gmail.com

APPENDIX C: Case Study using Core Impact Professional

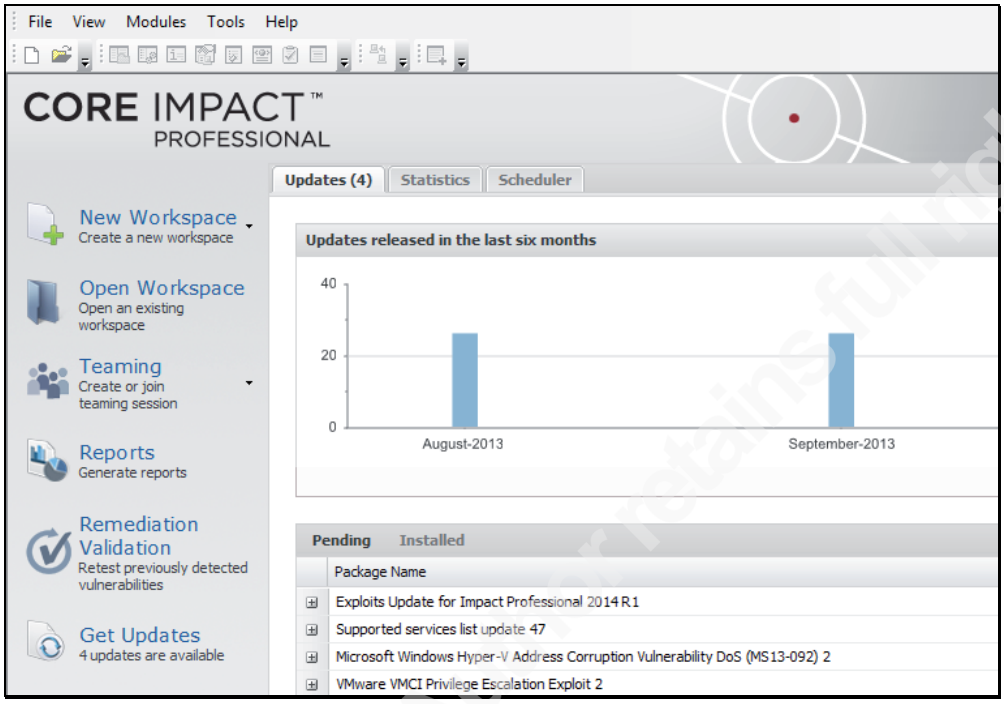
As previously mentioned, the PCI Cardholder Data Environment (CDE) is people, processes and technologies that store, process or transmit cardholder data or sensitive authentication data. For many Retailers, the CDE includes a variety of technologies. PCI DSS Requirement 11.3 requires yearly penetration testing of these technologies and after any significant change. This does not mean that each component changed is to be tested in isolation. The penetration testing approach is expected to simulate a real attacker pivoting through the CDE. Therefore, the penetration tester requires the ability to target multiple technologies for vulnerabilities and vectors through many layers of security controls. This appendix describes the Core Impact Professional software product and presents a use case study for meeting PCI DSS Requirement 11.3.

Core Impact Professional is a commercial penetration testing software product available from Core Security. Over the last 15 years, the product has grown beyond just being a network penetration testing tool into a versatile solution for assessing the security of hosts, network devices, mobile devices, wireless networks, IPS and firewalls. However, the scope of this paper is web application penetration testing, not all forms of penetration testing. That said, a web application is actually a collection of components including HTTP Server, Application Server, Database Server, Operating System, Firewall and possibly an IPS. A chain of security vulnerabilities or design weaknesses might result in unauthorized data leakage from the web application. Therefore, the tester's toolbox must be filled with options to pivot through all of these touch points. Core Impact Professional offers this versatility for testing.

Web Application penetration testing support has been offered by Core Impact Professional since in 2007. The product uses the Microsoft Windows platform and provides an intuitive user interface. Its home page presents a new project.

Author: Michael Hoehl, mmhoehl@gmail.com

Figure 4: Core Impact Professional initial home page.



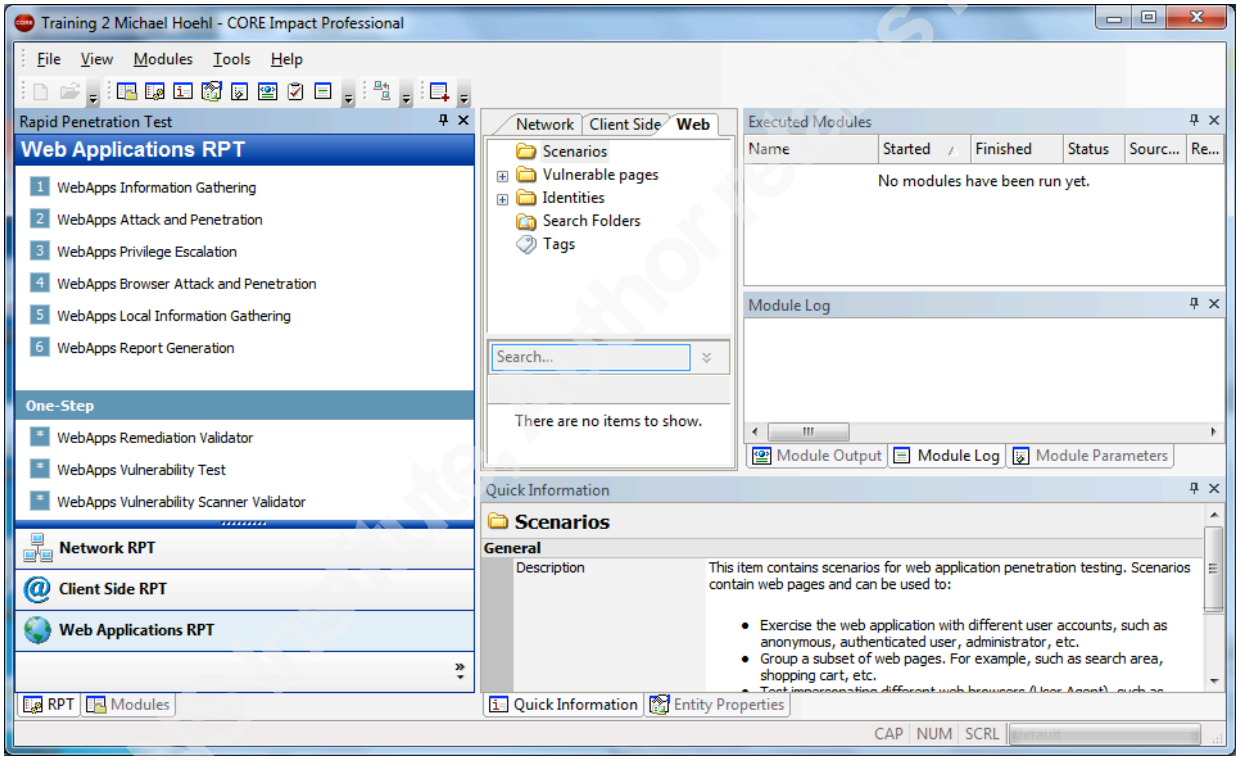
A workspace is essentially the work product of each penetration testing project. Core Security uses their own team of developers to create the exploits provided by Core Impact Professional. Core advertises that this approach guarantees their exploits are more effective, comprehensible and current. A listing of Core Impact Professional exploits is posted at http://coresecurity.com/core-impact-pro#sthash.UqCmjWTb.dpuf. Core Security typically releases 20-30 updates a month. Obtaining the updates is convenient and can be readily automated. If its exploits are inadequate for a penetration tester, Core Impact Professional offers the possibility of integration with MetaSploit from which to conveniently select and launch exploits.

The current version of Core Impact Professional provides a collaboration feature called Teaming. It provides multiple security testers the capability of interacting in the same workplace against the same environment across multiple copies of CORE Impact Pro. Each workspace is safeguarded with a unique password. This is very important as the workspace is essentially a recipe box of prior penetration testing information.

Author: Michael Hoehl, mmhoehl@gmail.com

Once a workspace is created, the penetration tester has the option to follow a wizard or manually perform each step. Web application penetration testing generally begins with the Web Applications Rapid Penetration Test (RPT) tab, as shown in the figure below. In addition, the RPT has six parts to it.

Figure 5: Core Impact Professional Web Applications RPT Workspace user interface.



The initial Information Gathering step provides a web crawling (aka spider) wizard that offers automated or interactive web crawling. The web crawler can impersonate almost all known web browsers. The wizard provides assistance configuring many options including:

- Depth of the web crawl;
- Forms response management;
- User authentication;
- Framework detection;
- Custom parsing of web page links;

Author: Michael Hoehl, mmhoehl@gmail.com

- Identification of SOAP web services.

As the wizard executes the necessary Python scripts, the penetration tester can observe logs and parameter output.

Once the website reconnaissance, mapping and discovery are complete, the penetration tester launches the Core Impact Professional WebApps Attack and Penetration wizard. Risk types based on the OWASP Top 10 are presented for testing. All or some of the identified potential vulnerabilities can be selected for testing. Testers can choose defaults or customize parameters for each test (e.g., SQL Injection, Cross-Site Scripting, etc.). A credit card primary account number (PAN) scanner is also included. Social Security Numbers and custom regular expressions can also be included with a scan.

Responsive Web Design approaches are now making websites available to a broader range of browsing devices (from the traditional PC browser to the smart phone browser). Web Services are easily modified to support mobile devices and Responsive Web Design. This is because of the web services architecture that separates user-interface and backend application logic; many mobile applications use this model. When these applications accept credit card payment, they become part of the CDE. Therefore, PCI DSS Requirement 11.3 applies. Core Impact Professional offers a proxy feature for mobile apps that crawls the backend of an application and performs the necessary penetration testing.

Figure 6: Testing of Web Services Used by Mobile applications from *What's New: CORE Impact Pro 2014 R1*

Author: Michael Hoehl, mmhoehl@gmail.com

Once initial Server-side attacks are complete, the penetration tester may choose to advance to Web Applications RPT privilege escalation attacks or switch between Web and Network RPT to pivot and build a chain of attacks. Core Impact provides access to each Python exploit script (called modules) allowing the penetration tester to selectively launch or customize provided Python scripts.

Results from prior vulnerability scans can be used as input to Core Impact Professional. This is very helpful for fast tracking and validating the vulnerabilities discovered by the ASV. Output from Acunetix Web Vulnerability Scanner, Cenzic, HP WebInspect, IBM AppScan, NTOSpider and Qualys Web Application Scanning can be used. In addition to penetration testing of ASV vulnerability findings, developer remediation efforts can be tested and validated. Assessors can automatically retest previously detected vulnerabilities to confirm developer work was completed as planned.

Lastly, a Report Generator is included that offers automatically generated Executive, Remediation Validation and Full Vulnerability reports. These reports map back to the OWASP Top 10 vulnerability list (PCI DSS Requirement 6.5).