ACI®’S DEVELOPMENT SECURITY MANAGEMENT PRACTICES

October 2018
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>STATIC CODE ANALYSIS</td>
<td>4</td>
</tr>
<tr>
<td>APPLICATION PENETRATION TESTING</td>
<td>5</td>
</tr>
<tr>
<td>THIRD-PARTY COMPONENT MANAGEMENT</td>
<td>6</td>
</tr>
<tr>
<td>PROCESSING SCAN RESULTS</td>
<td>7</td>
</tr>
<tr>
<td>GOVERNANCE AND EXCEPTIONS</td>
<td>8</td>
</tr>
<tr>
<td>CONTINUOUS IMPROVEMENT</td>
<td>8</td>
</tr>
<tr>
<td>CONFIDENTIAL AND PROPRIETARY INFORMATION</td>
<td>8</td>
</tr>
</tbody>
</table>
INTRODUCTION

ACI® is committed to industry best practices for managing security within its products and solutions and thus follows a structured secure software development lifecycle (SSDLC) approach that requires security-related artifacts at every stage (concept generation, project definition, design, development, validation, and distribution and maintenance). SSDLC principles are embodied within ACI’s Product Development Methodology (PDM), which accommodates waterfall, agile and iterative development methodologies. Regardless of development methodology, robust security-related development steps and artifacts are required, including: (1) static code analysis (also known as static scanning), (2) dynamic code analysis (also known as dynamic scanning or penetration testing) and (3) third-party component scanning and vulnerability management. In addition, ACI engineers undergo mandatory secure coding training. This process is designed to address sensitive personally identifiable information (PII) and payments data used within ACI’s solutions while satisfying industry standard best-practice behaviors as called out by PCI and PA-DSS expectations to mitigate common threats like the OWASP top 10 for sensitive cardholder data.

This document provides an overview of the capabilities of these tools within the context of ACI’s SSDLC and PDM applied to our Active products. Mature products follow similar practices, but at a slower cadence than the Active products.

NEW MAJOR/MINOR RELEASE DEVELOPMENT-RELATED SECURITY ACTIVITIES*

*Note: Non-major releases may incorporate different scanning strategies, including limited or no scans, based on release type or recent scan history.
STATIC CODE ANALYSIS

Security-related static analysis (aka static scanning) inspects code for patterns that may adversely affect the ability of the application to maintain confidentiality, integrity and security of all or part of the application. Static code analysis may also find defects that are not security-related, but are quality-related. A sampling of security- and quality-related code defects that static code analysis can discover are available upon request. Not all security or quality flaws can be discovered through static security testing.

For all Major or Enhancement releases, the PDM requires that the engineering teams must ensure that static code analysis has been performed in a timely manner within the project lifecycle, and that mandated remediations be completed before development is complete. Additionally, developers are encouraged to scan code changes in near-real time from their development environments to identify potential security defects early in the development lifecycle.

Static code analysis can be accomplished through manual or automated code inspection.

AUTOMATED STATIC CODE ANALYSIS

Many automated static scanning tools are available that detect and classify security and/or quality-related anti-patterns. The PDM specifies that only tools that are pre-approved by security engineering may be utilized for this purpose. Presently, approved tools include: Checkmarx and Klocwork Insight. Checkmarx is ACI’s preferred tool because it supports nearly all of ACI’s diverse development environments and IDEs, has a strong repertoire of rules (also called “checkers”) and allows for customization of rules embodied within those “checkers”, as follows:

Checkmarx includes strong IDE integration that facilitates quicker detection and resolution of vulnerabilities. Lines of code are highlighted with issues found via the Checkmarx scans. Developers can quickly navigate through the IDE to all files related to the defect, and thus are able to fully understand the issue. The tool helps identify areas where a change in behavior may result in the resolution of multiple issues, thus improving the quality of the changes and efficiency of the developer.

Checkmarx supports rule development through a simple query language and interactive audit workbench. It is easy both to understand the default rules provided by Checkmarx and customize them as needed on a corporate level, team level or project level. New queries may be created using the same workbench and query interface.

Checkmarx supports many languages and frameworks, including Java, C#/NET, PHP, Visual Basic 6.0, VB.NET, APEX, Ruby, Javascript, ASP, Perl, Android, Objective C, PL/SQL, HTML5, Python and Groovy. For C and C++ applications, Klocwork is our tool of choice.

When using automated tools, each automated static code analysis tool must be executed on the entire code base using the list of queries or rules prescribed by security engineering. The results are uploaded to a central server to facilitate review and tracking across multiple builds and projects. Each defect is assigned a unique defect ID and a severity level. The engineer works with developers to triage the results and resolve issues within the project. Analysis of false positive issues requires review by security engineering.

MANUAL CODE ANALYSIS

Manual code analysis (aka code reviews) may be performed instead of or in addition to automated code analysis. The PDM site includes a set of best practices, including documented coding standards, code review checklists and code review templates. Manual code reviews must follow PDM standards for documenting the reviews, the participants and any issues identified.

When conducting code reviews, many development teams use workflow tools that support the effort, such as Atlassian’s Crucible, to enforce peer code reviews prior to checking code into the source code repository.
APPLICATION PENETRATION TESTING

Application penetration testing (aka dynamic scanning or pen testing) involves detecting security flaws in actual run-time application behavior (rather than in code). The method employed is to simulate normal and malicious user behavior against a deployed version of the product under test, in order to expose symptoms of underlying security flaws in the application code. Dynamic code analysis is typically accomplished through a combination of tools and manual modeling of user-application interaction.

The PDM requires dynamic scanning to be completed for all Major and Enhancement releases. The goal of dynamic scanning is to identify remaining security vulnerabilities after static code analysis and associated remediation have been performed. Dynamic scanning may be performed using both automated testing tools and manual testing techniques.

AUTOMATED DYNAMIC SCANNING

ACI development teams may use satellite security members to perform dynamic scanning as part of their internal milestones using Burp Suite Pro, IBM App Scan or other developer-centric tools to support this level of scanning. The goal of this activity is to identify and resolve common web application vulnerabilities early in the development cycle. The scope of these scans may be limited to testing only the portion of the application affected by the changes implemented within the development period.

Dynamic scanning tools generally search for three types of vulnerability: application, infrastructure (frameworks) and third-party web components. ACI dynamic scanning focuses primarily on application-level vulnerability, but also includes scanning for the other types of vulnerabilities.

ACI’s security engineering team may also be called upon to perform dynamic scanning for Major and Enhancement releases. The scope of the scanning may vary from scanning only changes since the last scan to performing a full scan of the entire product. Security engineering uses IBM Security AppScan Standard as the primary automated testing tool and Burp Suite Pro as a secondary tool.

Some ACI products follow an agile development methodology with frequent and rapid deployment of changes to the production environment. For such products, security testing by security satellite engineers is required at least annually.

MANUAL DYNAMIC SCANNING

Automated tools performing application penetration testing test many variations of common web application vulnerabilities, but they typically are unable to detect a class of vulnerabilities involving data entered in one part of the application but consumed in a different part. The goal of manual security testing is to probe the application for vulnerabilities not detected by the automated testing tools and to test workflows that are not easily tested using automated tools.

Manual testing usually requires the use of a client-side proxy tool such as Burp Suite Pro plus knowledge of proper security testing. Security engineering will perform manual security testing or dynamic scanning as part of any testing that it performs on an application. Development teams may also perform this level of testing.
THIRD-PARTY COMPONENT MANAGEMENT

Many ACI products leverage the use of third-party components, such as open source software. These components may carry specific licensing requirements as well as have the potential for having vulnerabilities that could impact the security of the product. To address the potential issues that could arise, ACI policy requires that all products be scanned to identify all third-party components in use and that ongoing monitoring for vulnerabilities is used on supported release threads.

The ACI security policy requires that third-party component scanning be completed for all projects that are Major, Minor or Enhancement releases. All third-party components that are identified as the result of scanning are required to be submitted for review for legal (licensing), security and export control purposes.

THIRD-PARTY COMPONENT VULNERABILITY MANAGEMENT

ACI utilizes the Black Duck suite of products for third-party component scanning. These tools provide the ability to first scan the ACI product to identify all third-party components that are in use. Once components have been identified, the tool provides the detailed information on the component(s) including licensing, security and operational issues.

The Black Duck tool provides access to the National Vulnerability Database (NVD) for Common Vulnerability and Exposures (CVE) data. Additionally, Black Duck provides a proprietary vulnerability database to identify issues that may not be reported in the NVD.

In addition to identifying complete components that are in use by a product, the Black Duck tool provides the ability to scan ACI source code to identify sections of code that may have been copied from a third-party source. This functionality, known as snippet matching, allows ACI to identify embedded third-party source code that may carry additional licensing requirements.

TECHNICAL REVIEW OF INTELLECTUAL PROPERTY (TRIP)

When third-party components are identified as a result of the scanning process, the PDM and ACI policy require that all such components be submitted to the Technical Review of Intellectual Property (TRIP) process. This process provides for the review of the third-party components from a legal (licensing), security (vulnerability) and ECCN (export control) perspective.

The TRIP process helps to ensure:

- Use of third-party components complies with the licensing requirements specified by the component supplier
- That security issues are identified and remediated. Any component with outstanding vulnerabilities or that is no longer supported is not approved for use in ACI products
- That third-party components that contain cryptographic functionality that could affect the export classification of an ACI product are identified

THIRD-PARTY COMPONENT VULNERABILITY MANAGEMENT

When an ACI product that contains third-party components is released, the Black Duck tool provides the ability to perform ongoing monitoring and management of newly discovered vulnerabilities. The tool periodically (on a daily basis) retrieves new and updated information from the NVD to identify any new vulnerabilities that may apply to third-party components in use within an ACI product.

ACI has developed additional automated processes to leverage the vulnerability identification within the Black Duck tool to generate internal cases for management of the vulnerabilities. Product engineers then review the vulnerabilities to determine their applicability to the product’s usage of the component and to identify any required remediation.
Vulnerabilities are classified as consistently as possible according to an industry-accepted common vulnerability scoring system. Security engineering reviews the vendor-assigned severities and tightens them up where appropriate to reflect the stringent security requirements of individual payment applications. The tenets of ACI's scoring system are as follows:

- **High** – High probability of vulnerability being exploited which could result in the loss of confidentiality, integrity or availability of sensitive data, or the loss of integrity and availability of the application.

- **Medium** – Medium probability of vulnerability being exploited which could result in the loss of confidentiality, integrity or availability of sensitive data, or the loss of integrity and availability of the application. This classification is also used to identify possible coding errors which may affect business logic.

- **Low** – Low probability of vulnerability being exploited which could result in the loss of confidentiality, integrity or availability of sensitive data, or the loss of integrity and availability of the application.

- **Informational** – A variant of low probability vulnerabilities related to the leakage of information or identification of sub-optimal coding practices.

A summary of outstanding security issues is created during the validation phase of a project, identifying the number, severity and type of security issues detected through both static code analysis and dynamic scanning.

Following severity classification, ACI policy dictates that product architects in conjunction with security architects agree on a plan of action to fix defects classified as high severity before the product ships. Policy suggests that medium-severity issues may be added to the product roadmap backlog and resolved in a future release; the product architect and product manager prioritize the deferred issues to identify when they will be resolved.

Any exceptions to the above policies require executive approval on a defect-by-defect basis.
CONTINUOUS IMPROVEMENT

ACI prides itself on following a continuous improvement philosophy in order to bring best-in-class products and solutions to our customers. This philosophy extends to tool selection. We proactively assess static and dynamic scanning tools we feel will: expand the suite of detectable vulnerabilities and coding issues, support many languages and technological frameworks, provide the ability to detect and resolve issues earlier in the secure software development lifecycle with fewer false positives, and use flexible architectures that permit the development of custom rules and extensions. We look for tools that integrate seamlessly with developer IDEs, version control systems, bug tracking systems and other tools in our toolkit.

Tool assessment is an ongoing activity. We access new tools and technologies as they become available and reassess existing technologies whenever there appears to be sufficient improvements to warrant our re-evaluation. When change is warranted, we manage it quickly and efficiently through our tools strategy and AD tools teams dedicated to providing world-class service to our development community.

GOVERNANCE AND EXCEPTIONS

The PDM requires static scans and dynamic and third-party component scans be performed for all Major release projects and for all Enhancement type projects. Other types of projects, including patches, service packs and mandate projects, may be exempt from the scanning requirements, but scanning is required if those projects include product enhancements. Projects that do not alter the web application interface may be exempt from the dynamic scan analysis requirement.

The PDM requires that stage exit checklists be completed before a project can progress to the next stage. The development Risk Action Plan (RAP) includes entries where the product architect and lead engineer affirm they have completed the static and dynamic scanning activities as required by the PDM. These checklists are reviewed by business process analysts to ensure projects have complied with the PDM.

Projects that have outstanding high-severity security issues at the end of the development phase cannot be released for general availability unless a remediation plan has been created and the decision to defer remediation of those issues has received the approval of the product manager and the senior leadership team (VP/SVP). Only in rare instances are such exceptions permitted.

CONFIDENTIAL AND PROPRIETARY INFORMATION

The static and dynamic scan reports that ACI produces internally are considered ACI confidential and proprietary information which we do not share externally. Although ACI works to resolve the indicated types of issues prior to a release becoming generally available, earlier releases may exhibit unresolved vulnerabilities due to progressively improving defect detection processes within ACI. Security scan results are kept confidential to mitigate and minimize security risks for all ACI customers.
ACI Worldwide®, the Universal Payments® (UP®) company, powers electronic payments for more than 5,100 organizations around the world. More than 1,000 of the largest financial institutions and intermediaries, as well as thousands of global merchants, rely on ACI® to execute $14 trillion each day in payments and securities. In addition, myriad organizations utilize our electronic bill presentment and payment services. Through our comprehensive suite of software solutions delivered on customers’ premises or through ACI’s private cloud, we provide real-time, immediate payments capabilities and enable the industry’s most complete omni-channel payments experience.

LEARN MORE

WWW.ACIWORLDWIDE.COM
@ACI_WORLDWIDE
CONTACT@ACIWORLDWIDE.COM

Americas +1 402 390 7600
Asia Pacific +65 6334 4843
Europe, Middle East, Africa +44 (0) 1923 816393
© Copyright ACI Worldwide, Inc. 2018
ACI, ACI Worldwide, ACI Payment Systems, the ACI logo, ACI Universal Payments, UP, the UP logo, ReD, PAY.ON and all ACI product names are trademarks or registered trademarks of ACI Worldwide, Inc., or one of its subsidiaries, in the United States, other countries or both. Other parties’ trademarks referenced are the property of their respective owners.

ABR6735 10-18