Integrating Application Security into the Mobile Software Development Lifecycle

Keeping pace with the growth of mobile

According to the November 2015 edition of the Ericsson Mobility Report projections, there will be 6.1 billion smartphone users globally by 2020. The exponential growth in smartphone usage has led to a dramatic shift in the way consumers and employees interact with businesses. As a result, businesses must adopt new processes and strategies to ensure that they continue to offer a competitive and comprehensive solution in the market.

As with any revolutionary shift, adopting a mobile focused approach is complex and requires buy-in and participation from the company leadership. The widespread proliferation of smartphones has also brought a new set of security challenges that both development and security teams have to be cognizant of. The threat landscape has expanded to mobile applications as they are now used to access sensitive data and perform business critical activities.

Traditionally, development teams have been focused on rapid growth, and security teams only came to the table once all development work was complete and the applications were in production. However, in today's agile environments, the increased flexibility of the software development life cycle (SDLC) allows more features to be developed more quickly. This requires security to be embedded into the SDLC to allow for constant assessment of the application code for vulnerabilities and issues as the code is being developed.

Mobile applications are yet another avenue for attacks on corporate intellectual property. As organizations continue to invest a substantial amount of time and resources developing an application security program, securing mobile applications has become an integral part of a complete application security program in order to improve the organization's overall security posture.

Mobile software development lifecycle

The mobile software development life cycle consists of the following steps, which are performed as a part of a repeatable process called the phases of software development:

- Requirements
- Design
- Implementation
- User Acceptance Testing (UAT) / Quality Assurance
- Production

Historically, security testing has been typically performed during the UAT or QA phase near the end of development cycle. As a result, testing in only one phase of the SDLC provides a limited view of security. Unfortunately, a development team following this paradigm would likely not have identified security defects in the code until late in the development life cycle and after the code was compiled into a functional application. Finding security defects at this late stage of development is highly inefficient and very expensive.

Best practices in mobile software development lifecycle

Typical security activities in each phase of the SDLC are as follows:

Requirements

As software requirements are developed, the corresponding security requirements should also be defined. F or example, if sensitive customer data will be collected and stored, requirements on how the data should be encrypted, both in transit and at rest, should also be established.

Everyone involved in web application development should be provided basic security training. Scalability and repeatability are critical aspects of effective security training programs. Traditionally in the past, security training was done live - but as development teams grow, e-learning and computer based training (CBT) are a great way to ensure consistency, reproducibility, and regularity of training across your entire IT workforce. Best practices include:

- Developer training in secure coding best practices, OWASP Top 10 (at a minimum) delivered via CBT, SME or professional delivery.
- Recognizing early on that the application will need access, to process and store critical/sensitive data including PII, PCI, HIPPA and SOC, thus including security requirements in addition to function or usability requirements.

Design

Once the mobile application requirements are captured, an architecture is created that should correspond to these requirements. At this stage of development, necessary security controls should also be identified and allotted as part of the application. Best practices include:

- Designing appropriate security controls for a given type of critical data identified during the "requirements" phase.
- Incorporating Threat Modeling to identify and address the security risks associated with an application.

Implementation

After requirements have been determined and an architectural design is in place, constructing the mobile software begins. Ideally, developers should receive security feedback while they are coding. This feedback should be provided as early and often as possible. Because this phase is often the most labor-intensive, continuously running automated security assessments allows a developer to address issues in near-real time. Otherwise, organizations may develop applications built on faulty code, with security as an afterthought, rather than being designed in at the start. Best practices include:

- Running static source code security assessments (SAST) at least nightly to enable near real time feedback to developers regarding the security of their code.
- Using static source code analysis as an early indicator of code quality

UAT / Quality Assurance

New code needs to be tested before it goes into production to ensure that the code behaves as expected. While most organizations currently test for functional requirements, most are also beginning to test for security requirements in this phase as well. Best practices include:

- Testing pre-production code via dynamic analysis (DAST) to identify vulnerabilities

Production

In the deployment phase, continuous testing is vital to maintain security assurance and to protect the application where most attacks occur. Also, updates to production applications can introduce new flaws, therefore, any code updates should also be subjected to static, QA and production testing. Best practices include:

- Performing continuous dynamic analysis (DAST) and manual security testing as the new code is introduced or otherwise updated with major revisions or simple bug fixes.

WhiteHat Sentinel Mobile Dynamic Analysis

- Enables continuous and concurrent assessments
- Provides verified, prioritized vulnerability findings
- Offers flexible assessment configuration
- Provides detailed and custom reporting

WhiteHat Sentinel Mobile Manual Assessments

- Line-by-line assessment of the mobile source code
- One-time white-box testing
- Analysis of client, network, and server

Static Analysis and Dynamic Analysis

SAST and DAST are both critical for mobile application security

With the proliferation of mobile applications, it is vital to incorporate security into the mobile SDLC as a strategic initiative. The threat landscape is constantly changing, and with hundreds of millions of mobile apps already launched, it is critical for an organization to utilize static analysis as well as dynamic analysis to get complete coverage of their mobile SDLC. Each type of testing has its own advantages.

Performing nightly static application security testing on the mobile source code will greatly assist developers to ensure that they code securely throughout the development process. There are certain vulnerabilities that are best identified through SAST and may be eliminated early in the process. However, certain other types of vulnerabilities can only be seen once the mobile app is in production.

Therefore dynamic scanning of production applications is equally critical. Issues such as cross-site request forgery, business logic flaws, and some forms of cross-site scripting are much more easily identified by a DAST solution.

Using SAST and DAST technologies at the appropriate stage, along with manual testing, is necessary for a comprehensive mobile application security program.

Embedding security right at the beginning is key

A comprehensive mobile strategy involves not just strategic planning, but also identifying and mitigating roadblocks on the path to mobile maturity, establishing strategic objectives and KPIs, and choosing the right tools and technology. WhiteHat Security's Sentinel Mobile uses a combination of static analysis, dynamic analysis and manual testing to assess mobile applications for potential vulnerabilities early in the SDLC - where the cost to fix vulnerabilities and impact to resources is minimal.

WhiteHat Security Sentinel Mobile

Sentinel Mobile provides complete coverage across all aspects of mobile applications. Static analysis of the source code is performed to identify security vulnerabilities before the code goes to production. Sentinel Dynamic emulates mobile browsers and analyzes mobile optimized websites continuously to detect the full array of OWASP Top 10 vulnerabilities. Manual assessment by the security experts of our Threat Research Center (TRC) includes a line-by-line assessment of the mobile source code as well as an analysis of client, network and server.

Manual Mobile Assessment

This includes white-box testing and deep-dive penetration test into the mobile application. Source code analysis, including data flow analysis, and dynamic testing between the client and the server is also performed.

Production Mobile Website Assessment

Dynamic Analysis

Sentinel Dynamic emulates mobile browsers and analyzes mobile optimized websites continuously for the full array of OWASP Top 10 vulnerabilities and more.

Automated Mobile Assessment

Static Analysis

Static code analysis of mobile source code identifies vulnerabilities and provides validated, actionable results. It integrates seamlessly with agile processes and tools including IDEs, ALM tools and Bug tracking systems.

Integrated Assessment

	CLIENT SIDE APPLICATION	NETWORK	SERVER SIDE
STATIC	Static analysis with Sentinel Source		Static analysis with Sentinel Source
DYNAMIC		Dynamic analysis with Sentinel Dynamic	Dynamic analysis with Sentinel Dynamic
MANUAL	Manual assessment by Threat Research Center	Manual assessment by Threat Research Center	Manual assessment by Threat Research Center

Supported Vulnerabilities and Concerns

We scrutinize for a wide variety of vulnerabilities and privacy concerns including but not limited to the following:

- Configuration Setting
- Binary Analysis
- Anti-Analysis
- Jailbreak/Root Detection
- Authentication/Authorization
- Session Management
- Cryptography
- Data Handling
- Data Storage
- Handling of Personal Information
- Secure Coding
- Server Side Controls
- Informational Findings

WhiteHat SECURITY

WhiteHat Security, Inc. | 3970 Freedom Circle | Santa Clara, CA 95054 | 1.408.343.8300 | www.whitehatsec.com