Effective Software Security Management

choosing the right drivers for applying application security

Author: Dharmesh M Mehta

dharmeshmm@mastek.com / dharmeshmm@owasp.org

Table of Contents

Abstract

Effective Software Security Management has been emphasized mainly to introduce methodologies which are Practical, Flexible and Understandable. This white paper describes the need and methodology of improving the current posture of Application Development by integrating Software Security. It attempts to provide an effective platform for organizations to understand how they can align software security in their SDLC.

Security is an important property of any software. Many applications are outsourced too where the application development lacks strong integration of software security. The growing need to address software security measures across development life cycle has been discussed here. Application Security can be seamlessly integrated in the SDLC by introducing specific steps or process within the development phases. These practices which endeavor to secure the application developed will be what future customers will look for.

Introduction

If you wonder "What makes secure software different?" you would realize that security is an innate property of the software which was expected to be built in. Unfortunately, most of applications lack security today. The traditional practices used to develop outsourced applications are no more effective. Even the Indian IT services companies lag in improvising their SDLC at the same pace with the global industry. One of the weakest areas where these companies fall is Software Security. Current business environment is fraught with risks. The applications demand tight software security embedded inside to prevent hackers getting in. To incorporate software security measures, enterprises need to change their existing application development lifecycle.

The current scenario is such that many companies to an extent have started addressing security earlier in the lifecycle to mitigate the risks of application security attacks. But, there is still room for improvement. The application security landscape is changing rapidly.

Customers outsourcing applications need to ensure the application development lifecycle of the IT services provider embark software security inline. The IT services companies on the other hand need to develop confidence in the customer for software security levels in their SDLC.

Maintaining a high level of security is no simple proposition. One of the key issues with outsourced applications is that unlike functional concerns, non-functional concerns of application like security and performance are always given lower priority. If the services companies fail to understand the importance of these non-functional factors, the customer is at loss. At the end, if these security defects are injected due to lack of measures taken during SDLC, it may destroy customer value and trust.

Applying Security in Software Development Lifecycle (SDLC)

Growing Demand of Moving Security Higher in SDLC

Application Security has emerged as a key component in overall enterprise defense strategy. Companies that build a strong line of defense usually learn to think like an attacker. Often is a developer is asked to wear two hats: one as developer that works in complex distributed environments, and the other as a security expert who builds software security. Organizations that understand application security practices and priorities are using resources far more effectively than in years past, while avoiding costly and potentially crippling problems.

In the years past, anti-virus software, firewalls, intrusion detection and intrusion prevention systems have been successful enough to protect network and hosts. While still the bulk of attacks happen at network layer, attackers have been successful compromising the application with lower ratio of making applications as targets. The industry reports of organization suffering application attacks with significant downtime in the application or loss of customer data. Financial institutions, Healthcare providers, Retailers, Telecom Industry or even IT Companies have not been able to get escaped from becoming a victim of application attacks. The impact of these attacks have been damage to their brand name, loss in revenue, loss of customer data, system or network downtime and even legal issues with compliance to PCI (Payment Card Industry) or SOX (Sarbanes-Oxley) standards.

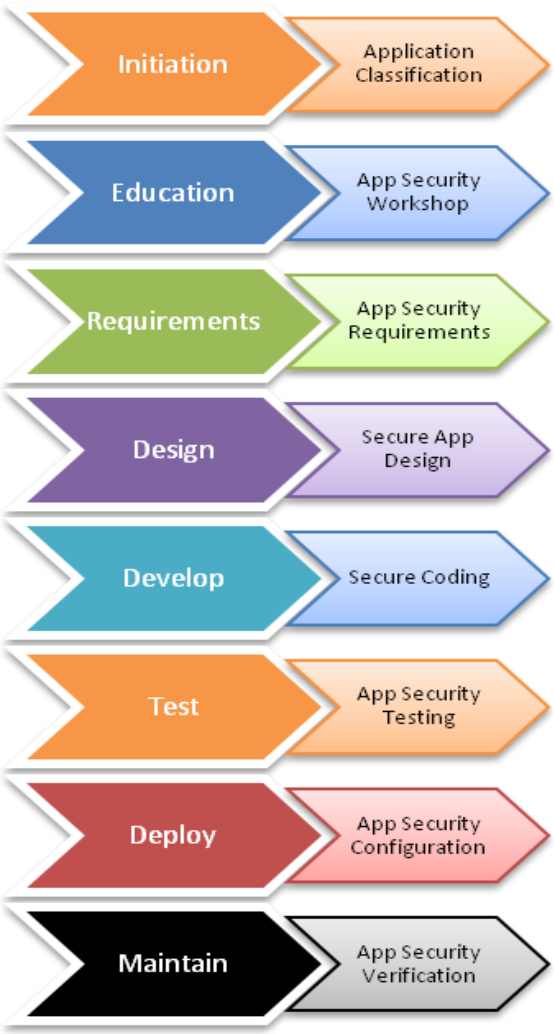
In the current world, software security assurance needs to be addressed holistically and systematically in the same way as quality and safety. Most of the assurance can be driven by improved software development practices. It is also important to realize that the security cost factor increases as you move down the SDLC.

In the next section of this white paper, we shall examine an approach to align application security in the current SDLC to produce secure software.

Effective Application Security Model

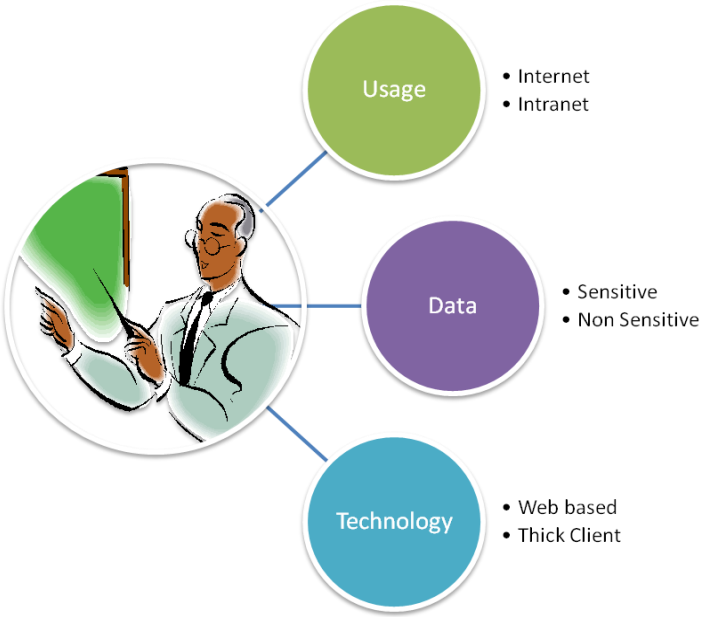
Let us discuss the following model to align application security in the SDLC. The organizations going forward must look to align in a similar fashion to take care of attackers. Every phase has been explained in the sections forward.

Effective Application Security Model for SDLC



I. Initiation – Application Classification

At the application initiation, try and classify the level of security expected of the software to be built in. Of the critical parameters that one can question to categorize the application include application usage (internet or intranet), data sensitivity (sensitive, personally identifiable or non-sensitive) and technology used (web based or non-web based application). These parameters help you categorize the application security level as High, Medium or Low.



This categorization is important to mandate and recommend security processes across SDLC. I am sharing a sample matrix showing the tasks which can be mandates, recommended or set optional for the application team to be followed in the SDLC.

Defining Tasks based on Application Classification

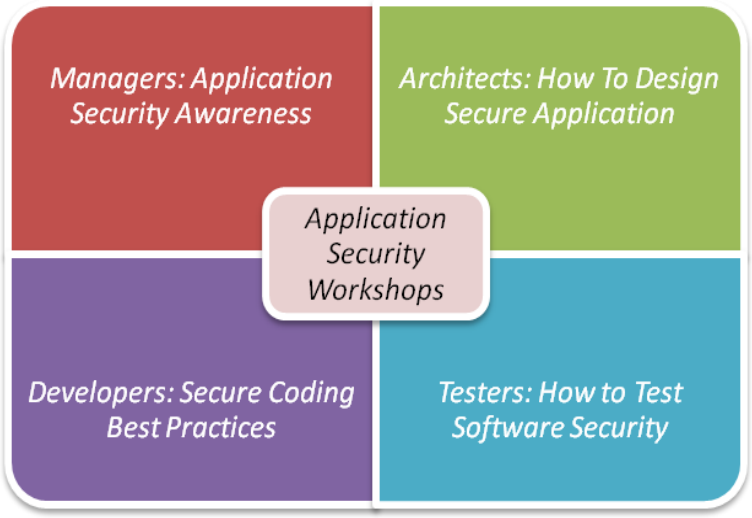
TASK ID	TASK	HIGH	MEDIUM	LOW
1	Application Security Training	M	M	R
2	Getting Security Req.	R	R	O
3	Secure Design Review	M	R	O
4	Secure Code Review	M	R	O
5	Security Testing	M	M	R
6	Regression Testing	M	M	R

M – Mandatory; R – Recommended; O – Optional

II. Education – Application Security Workshops

Awareness amongst the team members regarding application security issues is a must. The professionals must be able to foresee and address application security. You can think of having different training courses to train your team to inculcate culture of application security throughout the lifecycle. You can also think of getting technology specific training courses to help developers understand and tackle issues easily. Educating the team on application security to help build resilience in them to withstand the adverse attacks and also think about them prior to designing applications.

One of the better ways of training developers is demonstrating the vulnerabilities or exploits in their application or dummy application to help them realize how attackers work on. This perspective is very necessary to bring security in development phase. It helps them evaluate any module they work on with a mindset of how it can be hacked and how it can be mitigated.



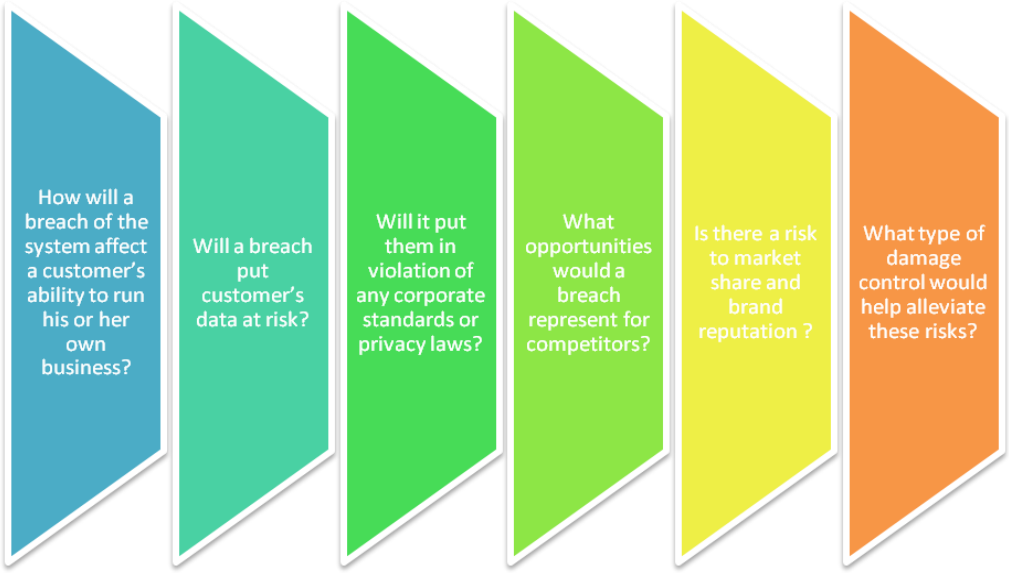
III. Requirements – Application Security Requirements

Separate Security Requirements from Functional Requirements to facilitate explicit review and testing. Gather all the possible application security requirements from the customer. The customer may not be aware of the security risks, risks to the business objective and vulnerabilities that could creep in while building an application. In these cases, ask the customer for security requirements.

Map specific security objectives in order to derive a secure design. The security objectives are based on understanding of what risks the end user might be exposed to and the risk to customer brand if security is compromised. For e.g. the objective that – only authorized users can access the application is the objective. This objective should be mapped to series of requirements. In the above example, a password protection mechanism could be implemented that enforces strong passwords (at least eight characters in length with a mandatory mix of numeric, upper case, lower case and special characters). The user account can be locked on five consecutive failed logon attempts.

For every use case, write a misuse case and assurance case. Since the use cases describe legitimate users and how they interact with the application to get real work done. Misuse cases (or abuse cases) describe attackers (malicious users) and how they can intentionally misuse the system. Analyze the requirements from customer by asking yourselves questions like below. Thinking proactively about your customer's security concerns will help you mitigate, or at least prepare contingencies, for such a worst-case scenario.

Analyzing Security Requirements



IV. Design – Secure Application Design

Most of the CIO's are concerned about the software security and the potential vulnerabilities that might creep in if the application is not designed securely. Loss in customer's trust can lead to disastrous effect on relationship.

The most effective practices in improving security architecture typically center around minimising the publicly available application features. This principle is often called as reducing the surface area of attack.

Document the assumptions and identify possible known attack scenarios against the system. Produce combined formal application specifications and security requirements specification. A very known activity adopted during this phase is called Threat Modeling. This is a technique used to understand and analyze threats against the system before it is built. It is primararily intended to identify architectural flaws rather than code level flaws. The basic steps in a Threat Modeling activity include Decomposing of Application, Defining Application Components & External Dependencies and Modeling the System to resolve threats.

Review the design of the application from security standpoint by enumerating all possible avenues of attack. This activity is genuinely helpful and does not have to be difficult. There are several threat, attack and vulnerability modeling tools and techniques. Microsoft, in particular has emphasized Threat Modeling and had also provided a tool to enable modeling.

The aim is to identify threats against each of the use case scenarios, system processes, data, transactions and functions. The common threats uncovered include possible loss in confidential data, unauthorized access, possible denial of service attacks, etc. By identifying all inappropiate actions that could be taken, we would capture all actions of malicious system use. This would help in mitigating the risks associated with the malicious system use. The risk response could be one which either *removes* the risk, *reduces* the risk, *transfers* the risk or *accept* the risk.

V. Develop – Secure Coding

It is true that most of the security bugs sneak in the application during development phase. Security Issues during coding phase include the language choice, development environment, coding conventions, coding guidelines, baselines for security, documents for sensitive data handling, implementation of security features and integration with external applications or systems.

Establish Secure Coding Guidelines for your applications. The process must mandate it for the developers to follow the guidelines for secure coding. There is a lot of information available for specific techniques for writing secure code on all different technologies. This information must be utilized to avoid coding errors. Ideally these practices are believed to be taught to the developers during the education phase. Hence, the important task here would be to ensure that these practices are followed across all the code modules and the defects are found at the early stage rather than the code being ready for deployment.

Many organizations take up premium tasks of using specialized tools for secure code review. Security Code Review tools available commercially are utilized to identify issues earlier in the cycle. However, automated code reviews are not very effective in analyzing all the security defects in the application. Hence critical parts of the applications must be provisioned for manual code review from peers or experts. Start rating the security defects to understand the impact and complexity of the issue. These metrics many a times helps you analyze the trade-off of performance because of intense security. Strike a balance between security and performance by verifying the business impact, complexity of attack and efforts required to mitigate the attack.

The commonly accepted secure coding practices include data validation (input and output validation), segregation of trust, no hard coded secrets, proper error and exception handling, safe coding constructs to prevent known injection attacks and proper auditing and logging of application.

VI. Test – Application Security Testing

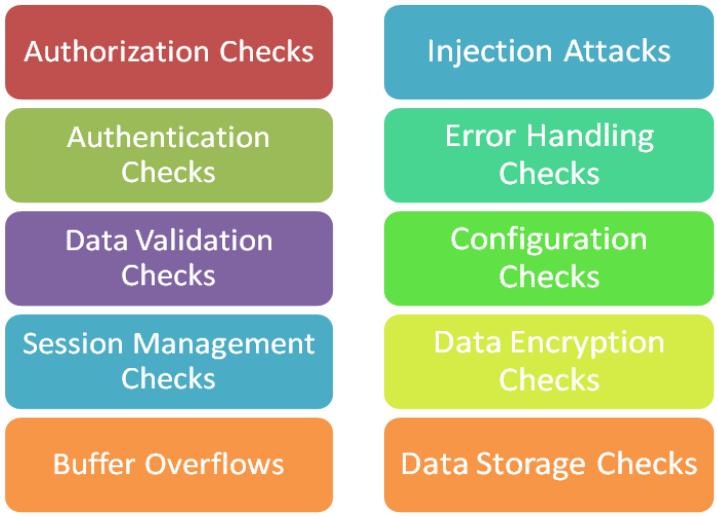
It is important to recognize that the cost of security increases as you move down the SDLC. Strategize to test early and testing often to prevent defects being leaked at a later stage and minimize the impact. Organizations usually take up software security at this phase by adopting black box testing of applications to identify the potential risks. However, the security bugs identified have a small window of closure and most of times end up in the production software by not being patched or having a temporary fix which may not be full proof.

Black box testing has matured in very well in the industry. The number of software code analysis or application scanners have increased exponentially in the market. Automated scanners provide vulnerability assessment or penetration testing. Security testing services are becoming a standard offering of many firms. Professinal training for application security testers is widely available.

One of the effective ways of handling security testing is to have a small team of security experts inhouse to take care of the testing or invest in an automated scanner to identify vulnerabilities. The option of hiring a third party to take penetration testing service may cost you premium.

Follow the security testing with a list of checks for the applications for comprehensive tests. The list must include the following checks at a minimum.

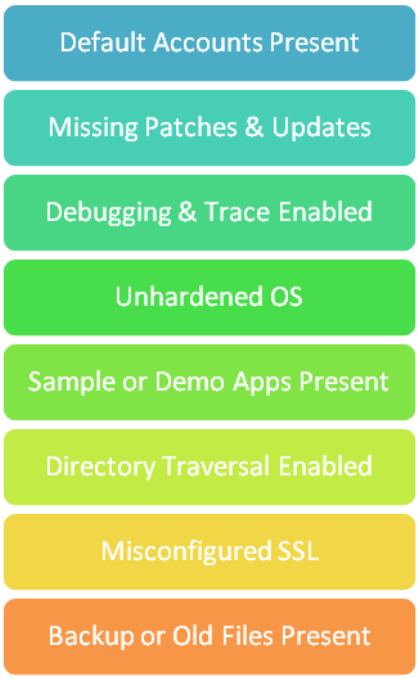
Common Security Tests



VII. Deploy – Application Security Configuration

It is observed many times that the development environment is lot more different than the production environment. The application in development environment is by default insecure because of many configurations in place like debugging enabled, trace enabled, accounts with known usernames and passwords, backup files present in the directories, etc. These are potential vulnerabilities found in the production boxes often categorized as Deployment Issues.

Few of the configuration checks include:



VIII. Maintain – Application Security Verification

Validate security of the application at regular periods. This is to verify if new security vulnerabilities have been injected over application maintenance or enhancements. The high rate of change in the normal code rapidly decays the accuracy of software security.

Regression Testing needs to be performed for the security vulnerabilities found during testing phase. The regression testing needs to take into account the technique by which the bug has been fixed or the severity level reduced. This metrics helps the application team to analyze security issues better in future designs and mitigate the same higher in the SDLC. The final report can be shared and benchmarked with the security countermeasures that were decided during Security Design phase. This will help make an internal compliance check with the application development.

Conclusion

Companies that understand application security as a multi dimensional issue will need to recognize the fact that the Software Development Lifecycle requires transformation at various phases to support application security.

Although each company might discover its own methodology to cater application security, the approach needs to be effective i.e. practical, flexible and understandable. In this paper we tried to discuss one of the effective approaches to address application security. I believe that in future, the enterprises will need to adopt one of the effective strategies and embrace the way of building a stronger and better enterprise.

About the Author

This white paper has been written by Dharmesh Mehta who works as Technical Analyst with Mastek Ltd. (www.mastek.com) in Mumbai, India. He is a Certified Ethical Hacker and is part of Application Security Assurance Team at Mastek. He is mainly involved in application security consulting for various projects at Mastek. He and his team colleagues are responsible for establishing application security across application development life cycle, performing security reviews and assessments and conducting application security workshops.

Dharmesh is also the Chapter Leader for OWASP Mumbai. He writes on application security at http://blogs.owasp.org/dharmesh and http://smartsecurity.blogpost.com. He can be contacted at dharmeshmm@mastek.com or dharmeshmm@owasp.org.

Other OWASP contributions from the author can be read at links below:

- Security Concerns in Web 2.0: http://www.owasp.org/index.php/OWASP_Papers/Jeopardy_in_Web_2_0
- Application Threat Modeling: http://www.owasp.org/images/6/6c/Dharmesh_Threat_Modeling.ppt