SERVICE DESCRIPTION

SpiderLabs Secure Development Training (SDT) – Instructor Led

Service Scope

Trustwave SpiderLabs provides customized training to developers to ensure that future applications are designed and implemented in a secure fashion. An instructor with a background in both application security and development will lead the training.

Key Benefits:

- Original research from experienced security advisors endorses technically relevant and current training content
- Tailored to fit the development environment that the development staff utilizes; this includes the programming language and application frameworks in use
- Customization of class modules based on the developers' own applications for real world examples
- Hands on lab work to understand and prevent vulnerabilities

This training service allows for effective and credible communications between the instructors and developers. Trustwave utilizes original security research and results from its many application penetration tests, code reviews, network penetration tests, and incident response investigations to ensure the training content is technically relevant and current.

In addition to basing the class on industry best practice, the training will be tailored to fit the development environment that the development staff utilizes; this includes the programming language and application frameworks in use. Many programming languages have a unique set of vulnerabilities that would only add overhead to another class; for example, discussing stack overflow vulnerabilities in a Java class would not add significant value for the students.

The most valuable aspect of this training is the customization that Trustwave SpiderLabs can perform is the use of examples taken from the developers' own applications. In the case where Trustwave SpiderLabs has performed a source code review service, or if real examples of known vulnerabilities can be provided, the instructor will base class modules around actual vulnerabilities from the students' projects. This approach is often far more effective because developers are already familiar with the context of their own applications.

The class also features hands-on lab work to teach developers how vulnerabilities are exploited in the real world. Following worked examples of exploiting application layer issues; developers have a better understanding of the

technical root of the problem, allowing them to be more effective at fixing it. Additionally, once developers realize how damaging vulnerability can be when it is exploited, they are more aggressive about preventing the vulnerability and will often become security advocates themselves.

Core Software Development Languages Supported:

- C/C++
- Java
- C#
- Visual Basic
- Objective C
- PHP
- ASP.Net
- Python
- Ruby
- Perl
- JavaScript
- Assembly
- Action Script
- COBOL Pascal
- TCL
- Perl
- VBScript
-

Scope and Project Phases

Classes will be held at on-site for up to a maximum of 40 students per class. The training and best practices guidance will consist of two full days of covering the following topics:

Table 1: **Course Agenda: Day 1 of 2**

Task	Description
Introduction to Security	The importance of security and the role of developers, along with the vocabulary used to describe important security concepts.
Policy Framework	Securing code is as much operational as it is technical. Organizational concepts such as risk management and change management are covered.
Secure Development Life Cycle (SDLC)	Building on the policies module, a cradle to grave development life cycle is outlined. Threat modeling is introduced, as a means of ensuring that software design can meet policy needs.
Lab: Threat Modeling	Different application scenarios will be described and the class will work through modeling threats to them. The primary goal is to get developers to think like an attacker, allowing them to anticipate threats to their own applications.
Principles of Secure Code	Focusing on code quality concepts, the practices that help to quantify secure coding are explored. Practical goals and approaches are reviewed, so that a consistent understanding of "secure" can be encouraged and measured appropriately.
Authentication & Authorization	The different aspects of authentication and authorization are covered. Pitfalls and common attacks against identity management are explored. Mistakes covered include insecure

Task	Description
	direct object references, failure to restrict URL access, and various types of other authentication and authorization bypass.
Session Management	Due to the stateless nature of the web, the security implications of session generation and management are discussed. This includes both Client-side token tracking and server-side session handling.
Input Validation	The heart of securing software is dealing with user-controlled data to ensure that it doesn't violate the integrity of a computer system. Improper input validation can allow for vulnerabilities like Cross-Site Scripting and SQL Injection, which are covered extensively. Where relevant, buffer overflow attacks will be covered. Less common input validation vulnerabilities such as XML Injection, XML Entity Expansion, XPATH Injection, and LDAP Injection are also discussed. The advantages of whitelisting over blacklisting are explained, and examples are provided of when more flexible validation schemes are required.
Lab: SQL Injection	A webpage vulnerable to SQL Injection will be hosted on a virtual machine. Students will be guided through practical attacks against a test website to learn how sensitive data can be extracted out of an SQL Injection vulnerability.
Lab: Cross-Site Scripting	Multiple webpages vulnerable to Cross-Site Scripting will be hosted on a virtual machine. Students can attack the webpage to inject various payloads. The instructor will demonstrate how a user's browser can be controlled using a Cross-Site Scripting vulnerability.

Table 2: **Course Agenda: Day 2 of 2**

Task	Description
Proper Encryption	Initialization vectors, key generation and storage, cipher selection, and decryption oracles will all be discussed. Hashing and secure password storage will also be explained.
Lab: Defense Exercise	In the first part of the lab, students will be given access to source code on the instructor's virtual machine that contains vulnerabilities to identify and fix. In the second part, the instructor will review the patched code with the class and why it did or did not work.
Logic Flaws	Application logic flaws can be devastating, but may take no special technical skills to exploit. Preventing them during the design and implementation phases will be discussed, as will techniques for finding logic flaws in existing applications.
Other Attacks	This module explores additional vectors of attack such as Cross-Site Request Forgery, insecure redirects, HTTP response splitting, browser specific issues, and rich media security. Compound and other advanced attacks are also covered in this module.
Error Handling and Logging	Handling exceptional circumstances poorly can leak information about a system useful to an attacker, and in some cases be a

Task	Description
	source of compromise themselves. This module outlines a variety of concerns and security best practices in the logging and communication of errors.
Application-Specific Lab	Attacks based on previous penetration tests or source code reviews will be demonstrated and students will be able to exploit the vulnerabilities themselves. Note: This will require previous coordination to provide student access to a development/test environment for attacks.
Exam	A brief exam will be administered based on the content of the training. The exam is scored by Trustwave SpiderLabs and exam scores are provided. All elements will be covered to measure effectiveness of the training.