CyberRes

Brochure

Role-Based Secure Development Training

Secure Development Training for everyone involved in the software development lifecycle is a cornerstone of any application security program and helps reduce the organization's exposure to application security risk.

Most organizations are aware that secure development training is a key security control that helps reduce application security risk. However, it is all too often only available on an ad-hoc basis. An effective secure development training program should feature:

- Mandatory training for all development personnel before they participate in the application development process.
- Appropriate training based on individual needs—no more, no less. For example, a regular Java developer should receive specific training on how to develop secure Java code. A Java developer for an e-commerce application will need more advanced training.
- On-demand capability that enables easy scheduling and minimizes the impact on developers' productivity.
- Scalability that suits the needs of everyone involved in development, including third-parties if appropriate.
- Up-to-date content to ensure that new threats and technologies are understood and addressed.

This eLearning offering is specifically designed to address these requirements. It provides over 100 hours of application security training material, divided into 13 role-based curricula. It is managed through Fortify on Demand, our cloud-based application security platform.

Curricula Overview

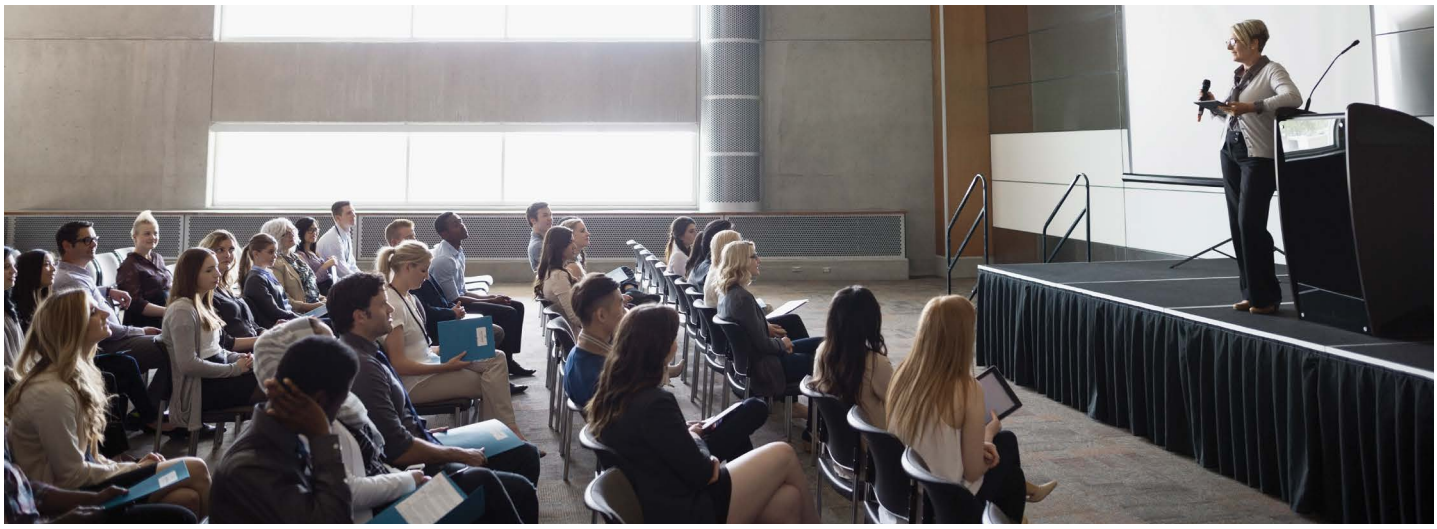
The Developer curriculum provides a thorough grounding in application security concepts. It includes programming language-specific constructs and implementation best practices. Upon completion of the course, developers will have a better appreciation of the importance of secure coding and the knowledge to develop secure applications in their chosen programming language and platform.

Role-Based Training Delivered by Fortify on Demand

- Developer curricula for Java.NET, C/C++, and PHP
- Other curricula specific to Software Architect, Project Managers, and Test/QA personnel
- Two training levels for maximum flexibility
- Topics include mobile, attack surface analysis, PCI, OWASP Top10, and application risk reduction
- Technologies covered include .NET, Java, iOS, Android, C/C++, C#, PHP, and HTML5
- Fully integrated with Fortify on Demand
- On-demand training minimizes the impact on developer productivity
- Available for training third-party developers
- Course completion is tracked to ensure compliance

The Software Architect curriculum covers how to design secure applications and includes architectural risk analysis and threat modeling. Upon completion of the course, software architects will know how to address application security risk at the application design stage. Other curricula are provided for Project Managers and Test/QA personnel.

Two levels of courses are offered. The Standard level is appropriate for developers of low- to medium-risk applications and takes approximately 5 hours to complete. The Premium level is designed for developers of high-risk applications or security lead developers and takes approximately 10 hours to complete.



Curricula

The table below shows the courses included in the Standard and Premium curricula for each role:

Code	Course	Java Developer		.NET Developer		C/C++ Developer		PHP Developer		Mobile Developer		Software Architect		Project Manager		Test/QA	
		Std	Pre	Std	Pre	Std	Pre	Std		Std	Pre	Std	Pre	Std	Pre	Std	
AWA101	Fundamentals of Application Security	●	●	●	●	●	●	●		●	●	●	●	●		●	
COD102	The Role of Software Security	●	●	●	●	●	●	●						●			
COD103	Creating Software Security Requirements	●	●	●	●	●	●	●						●			
COD104	Designing Secure Software	●	●	●	●	●	●	●						●			
COD105	Secure Software Development	●	●	●	●	●	●	●						●			
COD106	The Importance of Integration and Testing	●	●	●	●	●	●	●						●			
COD107	Secure Software Deployment	●	●	●	●	●	●	●						●			
COD108	Software Operations and Maintenance	●	●	●	●	●	●	●						●			
COD110	Fundamentals of Secure Mobile Development									●	●						
COD153	Fundamentals of Secure AJAX Code		●		●		●	●									
COD281	Java Security Model	●	●														
COD282	Java Authentication and Authorization	●	●														
COD283	Java Cryptography	●	●														
COD284	Secure Java Coding	●	●														
COD201	Secure C Encrypted Network Communications					●	●										
COD202	Secure C Run-Time Protection					●	●										
COD206	Creating Secure Creating Secure C++ Code					●	●										
COD207	Communication Security in C++					●	●										
COD307	Protecting Data in C++					●	●										

Continued on next page

Code	Course	Java Developer		.NET Developer		C/C++ Developer		PHP Developer		Mobile Developer		Software Architect		Project Manager		Test/QA	
		Std	Pre	Std	Pre	Std	Pre	Std		Std	Pre	Std	Pre	Std	Pre	Std	
COD216	Leveraging .NET Framework Code Access Security (CAS)			●	●												
COD217	Mitigating .NET Security Threats			●	●												
COD246	PCI DSS 3: Protecting Stored Cardholder Data		●		●												
COD247	PCI DSS 34: Encrypting Transmission of Cardholder Data		●		●												
COD248	PCI DSS 6: Develop & Maintain Secure Systems and Applications		●		●												
COD249	PCI DSS 11: Regularly Test Security Systems and Processes		●		●												
COD251	Creating Secure AJAX Code—ASP.NET Foundations				●												
COD252	Creating Secure AJAX Code—Java Foundations	●	●														
COD255	Creating Secure Code—Web API Foundations				●												
COD311	Creating Secure Code ASP. NET MVC Applications																
COD301	Secure C Buffer Overflow Mitigations						●										
COD302	Secure C Memory Management						●										
COD303	Common C Vulnerabilities and Attacks						●										
COD380	Protecting Java Code: SQLi and Integer Overflows		●														
COD381	Protecting Java Code: Canonicalization, Information Disclosure and TOCTOU		●														
COD382	Protecting Data in Java		●														

Continued on next page

Code	Course	Java Developer		.NET Developer		C/C++ Developer		PHP Developer		Mobile Developer		Software Architect		Project Manager		Test/QA	
		Std	Pre	Std	Pre	Std	Pre	Std		Std	Pre	Std	Pre	Std	Pre	Std	
COD321	Protecting C# from Integer Overflows and Canonicalization Issues				●												
COD322	Protecting C# from SQL and XML Injection				●												
COD323	Protecting Data in C#				●												
COD315	Creating Secure PHP Code							●									
COD316	Creating Secure iOS Code in Objective C									●	●						
COD317	Creating Secure iOS Code									●	●						
COD318	Creating Secure Android Code in Java									●	●						
COD361	HTML5 Secure Threats										●						
COD362	HTML5 Built-In Security Features										●						
COD363	Securing HTML5 Data										●						
COD364	Security HTML5 Connectivity										●						
COD352	Creating Secure jQuery Code		●														
COD411 COD412	Content can be found in Creating Secure C Code series, Creating Secure C++ Code Series, Protecting C Code Series																
DES101	Fundamentals of Security Architecture											●	●	●			
DES212	Architecture and Risk Analysis											●	●				
DES214	Securing Infrastructure Architecture											●	●				
DES215	Defending Infrastructure											●	●				
DES216	Securing Cloud Instances											●	●				
DES222	Applying OWASP 2017: Mitigating Injection	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES223	Applying OWASP 2017: Mitigating Broken Authentication	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES224	Applying OWASP 2017: Mitigating Sensitive Data Exposure	●	●	●	●	●	●	●	●	●	●	●	●	●		●	

Continued on next page

Code	Course	Java Developer		.NET Developer		C/C++ Developer		PHP Developer		Mobile Developer		Software Architect		Project Manager		Test/QA	
		Std	Pre	Std	Pre	Std	Pre	Std		Std	Pre	Std	Pre	Std	Pre	Std	
DES225	Applying OWASP 2017: Mitigating XML External Entities	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES226	Applying OWASP 2017: Mitigating Broken Access Control	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES227	Applying OWASP 2017: Mitigating Security Misconfiguration	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES228	Applying OWASP 2017: Mitigating Cross Site Scripting	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES229	Applying OWASP 2017: Mitigating Insecure Deserialization	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES230	Applying OWASP 2017: Mitigating Use of Components with Known Vulnerabilities	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES231	Applying OWASP 2017: Mitigating Insufficient Logging & Monitoring Vulnerabilities	●	●	●	●	●	●	●	●	●	●	●	●	●		●	
DES311	Creating Secure Application Architecture												●				
ENG211	How to Create Application Security Design Requirements												●				
ENG311	Attack Surface Analysis & Reduction												●				
TST101	Fundamentals of Security Testing													●		●	
TST251	Testing for SQL Injection															●	
TST252	Testing for OS Command Injection															●	
TST253	Testing for Classic Buffer Overflow															●	
TST254	Testing for Cross-site Scripting															●	
TST255	Testing for Missing Authentication for Critical Function															●	
TST256	Testing for Missing Authorization															●	

Continued on next page

Code	Course	Java Developer		.NET Developer		C/C++ Developer		PHP Developer		Mobile Developer		Software Architect		Project Manager		Test/QA	
		Std	Pre	Std	Pre	Std	Pre	Std		Std	Pre	Std	Pre	Std	Pre	Std	
TST257	Testing for Use of Hard-Coded Credentials															●	
TST258	Testing for Missing Encryption of Sensitive Data															●	
TST259	Testing for Unrestricted Upload of File with Dangerous Type															●	
TST260	Testing for Reliance on Untrusted Inputs in a Security Decision															●	
TST261	Testing for Execution with Unnecessary Privileges															●	
TST262	Testing for Cross Site Request Forgery															●	
TST263	Testing for Path Traversal															●	
TST264	Testing for Download of Code without integrity Check															●	
TST265	Testing for Incorrect Authorization															●	
TST266	Testing for Inclusion of Functionality from Untrusted Control Sphere															●	
TST267	Testing for Incorrect Permission Assignment for Critical Resource															●	
TST268	Testing for Use of a Potentially Dangerous Function															●	
TST269	Testing for Use of a Broken or Risky Cryptographic Algorithm															●	
TST270	Testing for Incorrect Calculation of Buffer Size															●	
TST271	Testing for Improper Restriction of Excessive Authentication Attempts															●	
TST272	Testing for Open Redirect															●	
TST273	Testing for Uncontrolled Format String															●	
TST274	Testing for Integer Overflow or Wraparound															●	
TST275	Testing for Use of a One-Way Hash without a Salt															●	

Continued on next page

Contact us at **CyberRes.com**
Like what you read? Share it.

Code	Course	Java Developer		.NET Developer		C/C++ Developer		PHP Developer		Mobile Developer		Software Architect		Project Manager		Test/QA	
		Std	Pre	Std	Pre	Std	Pre	Std		Std	Pre	Std	Pre	Std	Pre	Std	
TST222	Testing for OWASP 2017: Injection															●	
TST223	Testing for OWASP 2017: Broken Authentication															●	
TST224	Testing for OWASP 2017: Sensitive Data Exposure															●	
TST225	Testing for OWASP 2017: XML External Entities															●	
TST226	Testing for OWASP 2017: Broken Access Control															●	
TST227	Testing for OWASP 2017: Security Misconfiguration															●	
TST228	Testing for OWASP 2017: Cross Site Scripting															●	
TST229	Testing for OWASP 2017: Insecure Deserialization															●	
TST230	Testing for OWASP 2017: Use of Components with Known Vulnerabilities															●	
TST231	Testing for OWASP 2017: Insufficient Logging and Monitoring															●	

Curricula are pursued per named user per year. Volume discounts apply. For more information, email us at **fodsales@microfocus.com**.

Course content is provided through our partnership with Security Innovation. For a detailed description of each course, click here.

CyberRes
A Micro Focus line of business