



Tough Target: Awareness for Application Developers

- Teaching security to application developers

```
End Sub  
End Sub  
Private Sub tbToolBar_ButtonClick  
On Error Resume Next  
timTimer.Enabled = True  
Select Case Button.Key  
Case "Back"  
brwWebBrowser.GoBack  
Case "Forward"  
brwWebBrowser.GoForward  
Case "Refresh"  
brwWebBrowser.Refresh  
Case "Home"  
brwWebBrowser.Home
```



Speakers

- Nish Bhalla: Founder, Security Compass
– nish@securitycompass.com
- Rohit Sethi: Manager, Security Compass
– rohit@securitycompass.com





Agenda

Traditional Security Awareness

Developer Awareness Requirements

Training In-House

Training External

Strategies for Staying In-Budget



Traditional Awareness

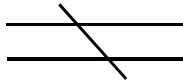
- Aims at avoiding common pitfalls (e.g. unsafe browsing, password selection, etc.)
- Analogy – teaching a driver how to drive safely





Traditional Awareness

- Traditional security awareness doesn't address developer needs
- More like teaching a mechanical engineer how to create safe cars, rather than how to drive safely



Question to Audience

- What's been the toughest part of teaching developers in your experience?





Agenda

Traditional Security Awareness

Developer Awareness Requirements

Training In-House

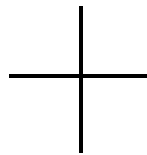
Training External

Strategies for Staying In-Budget



Developer Awareness Challenges

- Teacher needs to Understand:
 - Software engineering
 - Application security



CSISX
SECURITY EXCHANGE

Developer Awareness Challenges

- Like all training, teachers need to:
 - Project voice
 - Speak clearly
 - Understand body language



Security Compass

CSI
COMPUTER
SECURITY
INSTITUTE

CSISX
SECURITY EXCHANGE

Developer Awareness Challenges

- Content needs to be interactive
 - Time for a demo!

Security Compass

CSI
COMPUTER
SECURITY
INSTITUTE



Developer Awareness Challenges

- Content needs to address:
 - How to attack
 - How to spot poor code
 - How to write securely



Developer Awareness Challenges

- Needs to address security throughout SDLC





Agenda

Traditional Security Awareness

Developer Awareness Requirements

Training In-House

Training External

Strategies for Staying In-Budget



In-House Advantages

- Investment rather than cost per developer
- Expertise in house, so able to impart knowledge and take part in projects between training





In-House Disadvantages

- Tough to find correct skill-set
- Highly marketable individual
- Doesn't scale for multiple programming languages



Building an In-House Program

- **Find the correct training resources:** Application security training resources need to be strong:

Constructors
(application)



Detectives
(security)



Presenters
(training)





Building an In-House Program

- Reality: very few people have all skills
- Need to hire somebody and train them on missing skills



Building an In-House Program

- Reality: software development – very hard to teach; security much easier to teach
- Find developer with strong communication skills
- Ensure they get real world app sec experience to bring to classroom





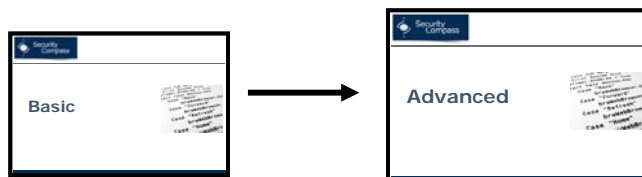
Develop a Strong Curriculum

- At least use Open Web Application Security Project (OWASP) Top 10 (<http://www.owasp.org>)
- Integrate real examples
- Challenging labs



Develop a Strong Curriculum

- Varying degrees of difficulty



- Include necessary languages/frameworks
 - E.g. J2EE, .Net, PHP, Struts, Ruby on Rails, etc.
- Include labs to break into apps





Develop a Strong Curriculum

- Use “anti-patterns”
- Include useful tools to help build securely



Develop a Strong Curriculum

- Goodie-bag of takeaways:
 - Methods to review current SDLC and how to build a strong one
 - Guide to help build language standards
 - Guide to help build architecture standards
 - Questionnaires/checklists as starting point for security





Agenda

Traditional Security Awareness

Developer Awareness Requirements

Training In-House

Training External

Strategies for Staying In-Budget



External Advantages

- No upfront investment in training the trainers
- Lots of varied experience from prior clients
- Multiple areas of expertise
- Dedicated to YOUR happiness – no politics!





External Disadvantages

- Can be cost prohibitive, especially to small organizations
- May be difficult to differentiate between effective and ineffective training programs



Determining a Budget

- Who's paying? Development, Security, LOBs?
- Sometimes Compliance department can pay
- Typically, training can range anywhere from \$500 to \$5,000 per student
 - How do you know how much to spend?





Vendor Selection

- What credentials does your vendor organization have?
- How many developers have they taught before?
- Do they understand both software engineering AND security? What credentials do they have in both?
- Are they strong presenters?



Vendor Selection

- Good hackers are NOT necessarily good at creating secure software
 - Should a good car thief teach you how to build secure cars?





Vendor Selection

- Cheapest is not necessarily the best
- Most expensive is not necessarily the best
- Judge based on credentials, experience, and ability to adapt to your technical environment
- Remember: “hacking & finding bugs” isn’t “hacking, finding bugs, & creating secure code”
 - Need all three!



Agenda

Traditional Security Awareness

Developer Awareness Requirements

Training In-House

Training External

Strategies for Staying In-Budget



CSISX
SECURITY EXCHANGE

Prioritize

- Most proactive security move is to train ALL developers, but few orgs have budget for this

High Risk Apps

Medium Risk Apps

Low Risk Apps

Testers Developers Lead Developers Architects

Importance

Security Compass

CSI
COMPUTER
SECURITY
INSTITUTE

CSISX
SECURITY EXCHANGE

Supplement

- Consider using books or computer-based training to supplement for low budgets
- Once benefits are proven, get more funding for next round of training

Security Compass

CSI
COMPUTER
SECURITY
INSTITUTE



Looking Forward

- Consider hiring developers who are already trained



Analyzing Costs

- Suppose 1000 records stolen, non-technical costs:
\$166,272 [1]
 - Cybercrime consulting, attorney fees, customer notification, call center support, crisis management consulting, media management, credit monitoring for affected customers, regulatory investigations, government fines or fees
- Technical costs (cost of fixing software defect in production): **\$8,500**
 - \$3,500 [2] (manpower, computer time, testing time, configuration management, fault-report database updates, generation of release notifications internally),
 - \$5,000 estimated for system downtime (lost sales, inability to generate new sales, time wasted for marketing & sales staff)
- Total cost to fix in production: **\$174,772**

[1] Insurance calculator – costs for stolen records: <http://www.tech-404.com/calculator.html>

[2] <https://www.thedacs.com/techs/roispi2/> - the Data Analysis Centre for Software: Return On Investment from Process Software Improvement (ROI-PSI)



CSISX SECURITY EXCHANGE Fix in SDLC - Test

- What if the bug was caught with application penetration testing?

The image shows a collage of application penetration testing tools. On the left, there's a WebScarab interface with a 'Summary' tab and a 'Tree Selection filters conversation list'. In the center, a Nessus scan results window is visible, showing a 'Summary' section with a red 'Error' icon and a message: 'CVS3 scans sensitive information in the directory. This information could be exposed and give an attacker knowledge which could later be used to...'. On the right, a login page for 'glaffodil CRM' is shown with a 'Wrong Username or Password' error message. Below the login page, the text 'Application Penetration Testing Tools' is written. At the bottom left is the 'Security Compass' logo, and at the bottom right is the 'CSI COMPUTER SECURITY INSTITUTE' logo.

CSISX SECURITY EXCHANGE Fix in SDLC - Test

The diagram shows a linear SDLC process flow: Design → Code → Test → Production. The 'Test' stage is highlighted in green, indicating the focus of the slide.

- What if it was caught with pen testing?
- Cost of defense per vulnerability: **\$1,500**
 - Estimated cost for testing \$30,000, estimated # of bugs prevented: 20 [1]
- Technical costs (cost of fixing software defect in test): **\$2,125**
 - Based on ¼ of cost of being caught in production [2] (manpower, computer time, testing time, configuration management)
- Total cost to fix in testing: **\$3,625**

At the bottom left is the 'Security Compass' logo, and at the bottom right is the 'CSI COMPUTER SECURITY INSTITUTE' logo.

CSISX SECURITY EXCHANGE Fix in SDLC - Code

- What if the bug was caught with source code review?

The image shows a side-by-side comparison. On the left is a screenshot of a Java source code file named 'NewLeadBean.java'. The code defines a login method that takes a username and password. It constructs a SQL query string by concatenating the user input directly into the query, which is a classic SQL injection vulnerability. The code then executes the query and checks for an 'admin' user. On the right is a screenshot of a web browser showing a login form for 'daffodil CRM'. The form has fields for 'Username' and 'Password', a 'Login' button, and a 'Only leads' checkbox. A message above the form reads 'Wrong Username or Password', indicating a failed login attempt.

Source Code Review



CSISX SECURITY EXCHANGE Fix in SDLC - Code



- What if it was caught with code review?
- Cost of defense per vulnerability: **\$600**
 - Estimated cost for training \$30,000, estimated # of bugs prevented in development: 50
- Technical costs (cost of fixing software defect in test): **\$920**
 - Based on 6.5/60 of cost of being caught in production [2] (developer's time for redevelopment of code and unit testing)
- Total cost to fix in development: **\$1,520**



CSISX Fix in SDLC - Design

SECURITYEXCHANGE

- What if the bug was caught during design?

Threat Model & Design Review

Security Compass

CSI
COMPUTER
SECURITY
INSTITUTE

CSISX Fix in SDLC - Design

SECURITYEXCHANGE

- What if the bug was caught during design?
- Cost of defense per vulnerability: **\$150**
 - Estimated cost for training \$15,000, estimated # of bugs prevented in design: 100
- Technical costs (cost of fixing software defect in test): **\$142**
 - Based on 1/60 of cost of being caught in production [2] (developer's time for change in design)
- Total cost to fix in design: **\$292**

Security Compass

CSI
COMPUTER
SECURITY
INSTITUTE



Fix in SDLC - Summary

- Like QA, earlier bug catching is quicker
- Unlike QA, cost of production vulnerability can be 1000x more than design

