



# **Web Attacks and How to Stop Them**

**John Rogers**

**Senior Application Security  
Engineer**

**Lincoln Financial Group**

# Objectives

## ➤ Overview

- Threats
- Vulnerabilities
- Remediation
- Education & Awareness

## ➤ Invite Discussion

- Please ask questions
  - Short ones during
  - Long ones after



# Warning Get out of jail card

## Always Get Written Permission!

\*\*\*\*\* Warning \*\*\*\*\*

Information, programs, and techniques contained in this presentation should not be used on corporate equipment or systems unless you have explicit written permission of the system owner, your management team and your Information Security department.

There are cases of well intentioned individuals having their employment terminated because they utilized penetration tools or techniques on company hardware or systems without written permission.

\*\*\*\*\* Warning \*\*\*\*\*

# Introduction

## **John Rogers (JR)**

- ▶ Senior Application Security Engineer
- ▶ Certified Information Systems Security Professional (CISSP)
- ▶ InfraGard Nebraska Board Member
- ▶ Application and User Interface developer
- ▶ Email: [John.Rogers@lfg.com](mailto:John.Rogers@lfg.com)

# Outline

Web Application  
Threats

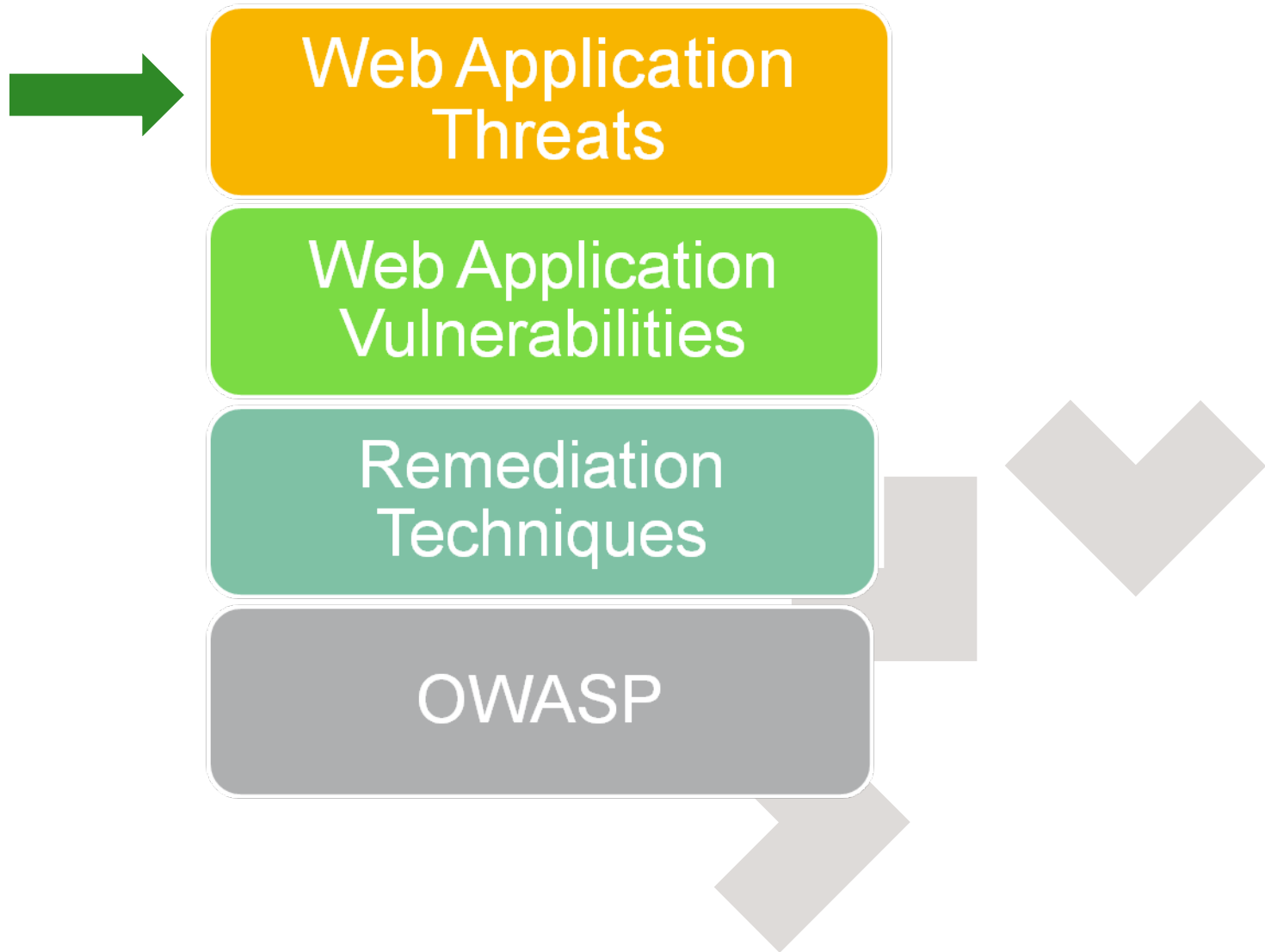
Web Application  
Vulnerabilities

Remediation  
Techniques

OWASP



# Checkpoint



# Web Application Threats

- The Top Cyber Security Risks
  - SANS Report – September 2009
  - <http://www.sans.org/top-cyber-security-risks/>
- >60% of all Internet attacks
- >80% of the vulnerabilities being discovered
  - Cross-Site Scripting
  - SQL Injection



# Web Application Threats

- Immature Security Awareness
- In-House Development
- Deceptive Simplicity
- Rapidly Evolving Threat Profile
- Resource and Time Constraints
- Overextended Technologies



# Web Application Threats

- **Users can submit Arbitrary Input!**
  - Never, ever, trust the client, no control.
  - Users can interfere with any piece of data transmitted.
  - Client side validation is considered a vulnerability.
  - Don't Trust Client Demo.



# Web Application Threats

## ➤ Don't Trust Client Demo

- Firefox

- Burp Suite

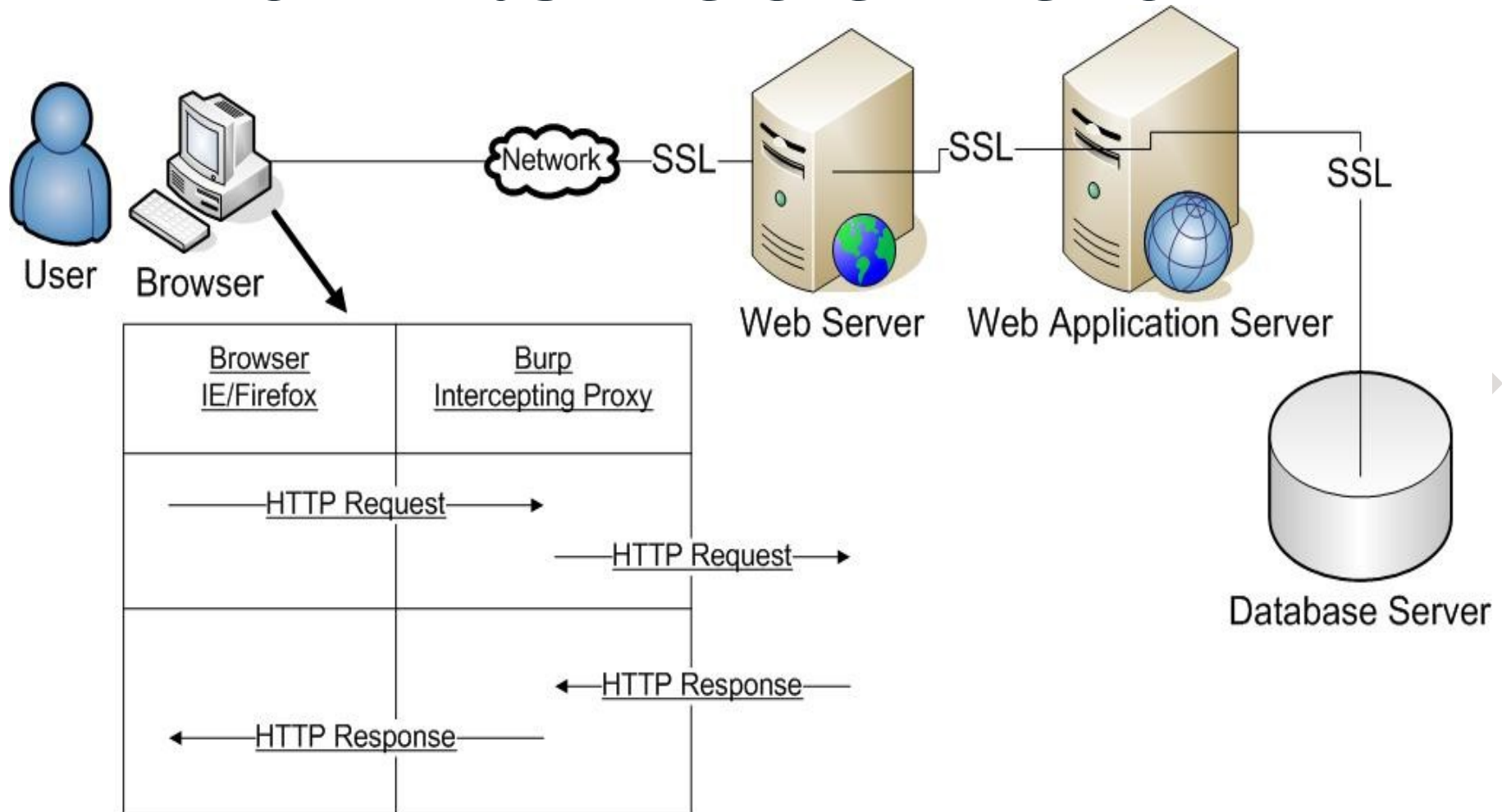
- Integrated platform for analyzing web applications.

- Intercepting Proxy, Spider, Decoder, etc.

- Combine manual and automated techniques to enumerate, analyze, attack and exploit web applications.

# Web Application Threats

## ➤ Don't Trust the Client Demo



# Web Application Threats

## ➤ Don't Trust the Client Demo

➤ URL: <http://demo.testfire.net>

➤ Sign In

➤ Username: jsmith

➤ Password: Demo1234

➤ Modify Account Details

➤ Add Select element

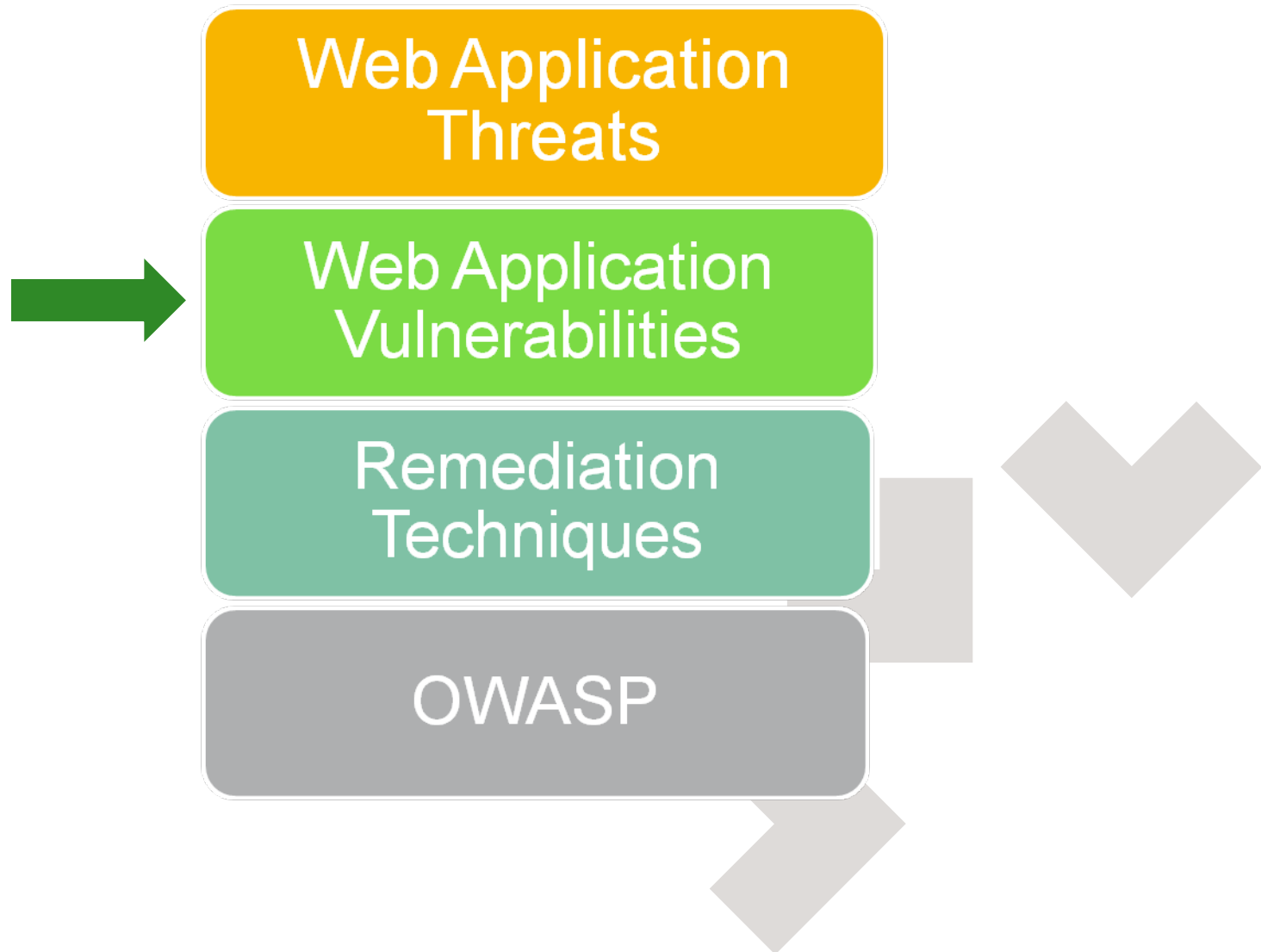


# Web Application Threats

## ➤ Don't Trust the Client Demo

<u>PERSONAL</u>	<u>SMALL BUSINESS</u>
<h2>Hello John Smith</h2> <p>Welcome to Altoro Mutual Online.</p> <p>View Account Details:</p> <div><div><div>1001160140 Checking</div><div>1001160140 Checking</div><div>1001160141 Savings</div><div>1001160141 Savings JR Hacked This</div></div><div>▼</div><div>GO</div></div> <p><b>Congratulations!</b></p> <p>You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!</p> <p>Click <a href="#">Here</a> to apply.</p>	

# Outline



# Web Application Vulnerabilities

- Firewall
- Cross-Site Scripting (XSS)
- SQL Injection
- Malicious File Execution
- Top Web Application Attacks 2008/2009

# Web Application Vulnerabilities

## ➤ Firewall

- False sense of security
- Firewall = No Application Security
- Port 80/443 are wide open
- Firewall only = “Crunchy Outside, Soft Chewy Inside”



# Web Application Vulnerabilities

## ➤ **Cross-Site Scripting (XSS)**

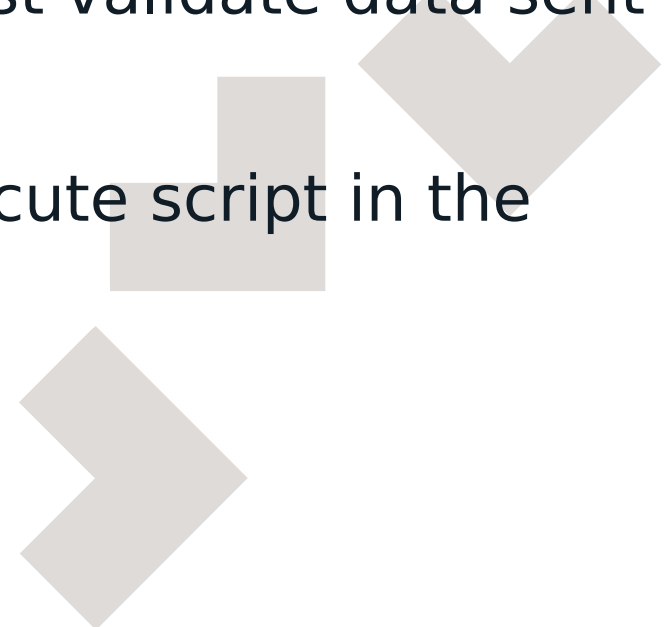
- OWASP Rank/Description

- #1 on OWASP Top Ten

- Attack against the client (initially)

- Application does not first validate data sent from client.

- Allows attackers to execute script in the victim's browser.

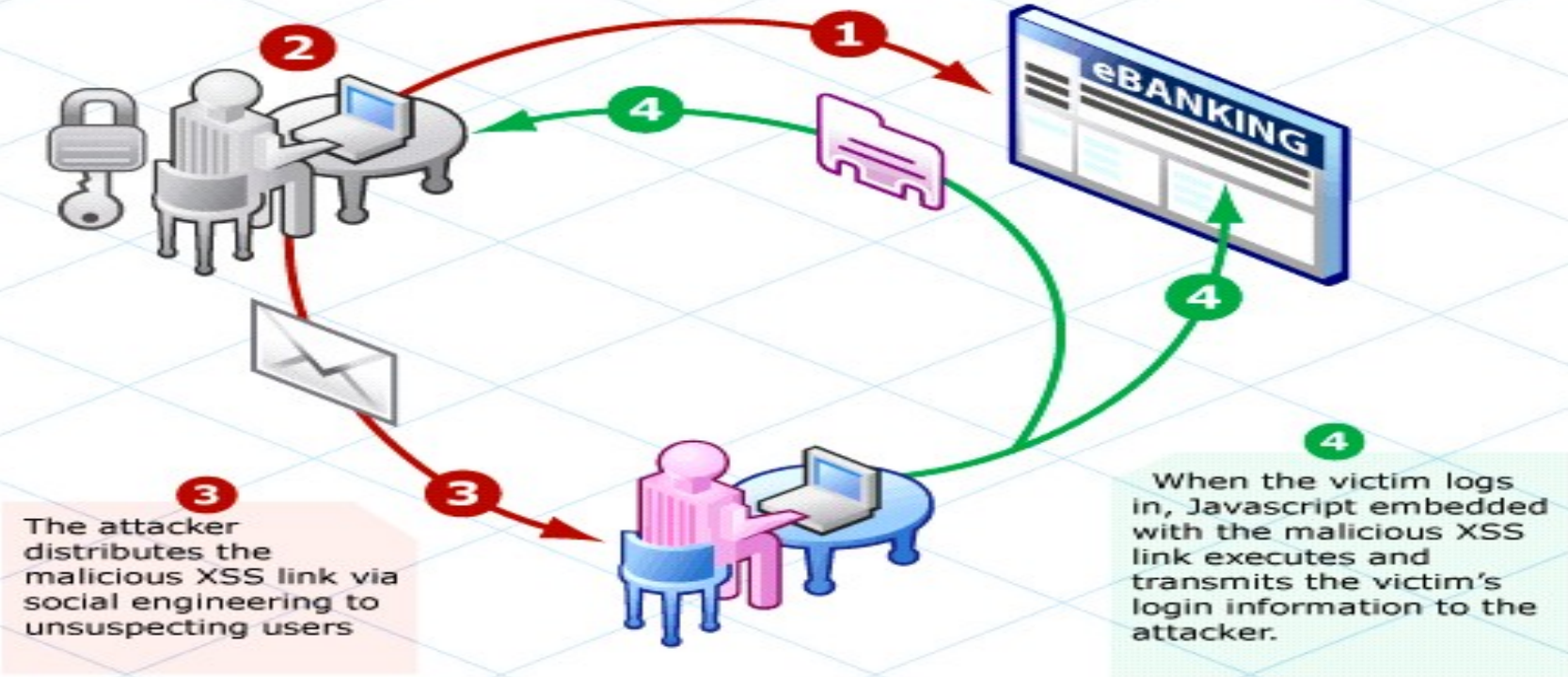


# Web Application Vulnerabilities

## ➤ Cross-Site Scripting (XSS)

**1** An attacker finds an XSS hole in a web application

**2** The attacker creates an attack URL for stealing sensitive information and disguises it so that it appears legitimate



# Web Application Vulnerabilities

## ➤ Cross-Site Scripting (XSS)

### ➤ Types

- Reflective
- Stored
- DOM Injection

### ➤ Demo

- AltoroMutual Search – Reflective XSS
- WebGoat XSS Lesson – Stored XSS

# Web Application Vulnerabilities

## ➤ Cross-Site Scripting (XSS)

➤ Demo – AltoroMutual Search – Reflective XSS

➤ <http://demo.testfire.net>

➤ Unauthenticated vulnerability

➤ Test<script>alert('hi mom');</script>

➤ <iframe  
src=http://ha.ckers.org/scriptlet.html <

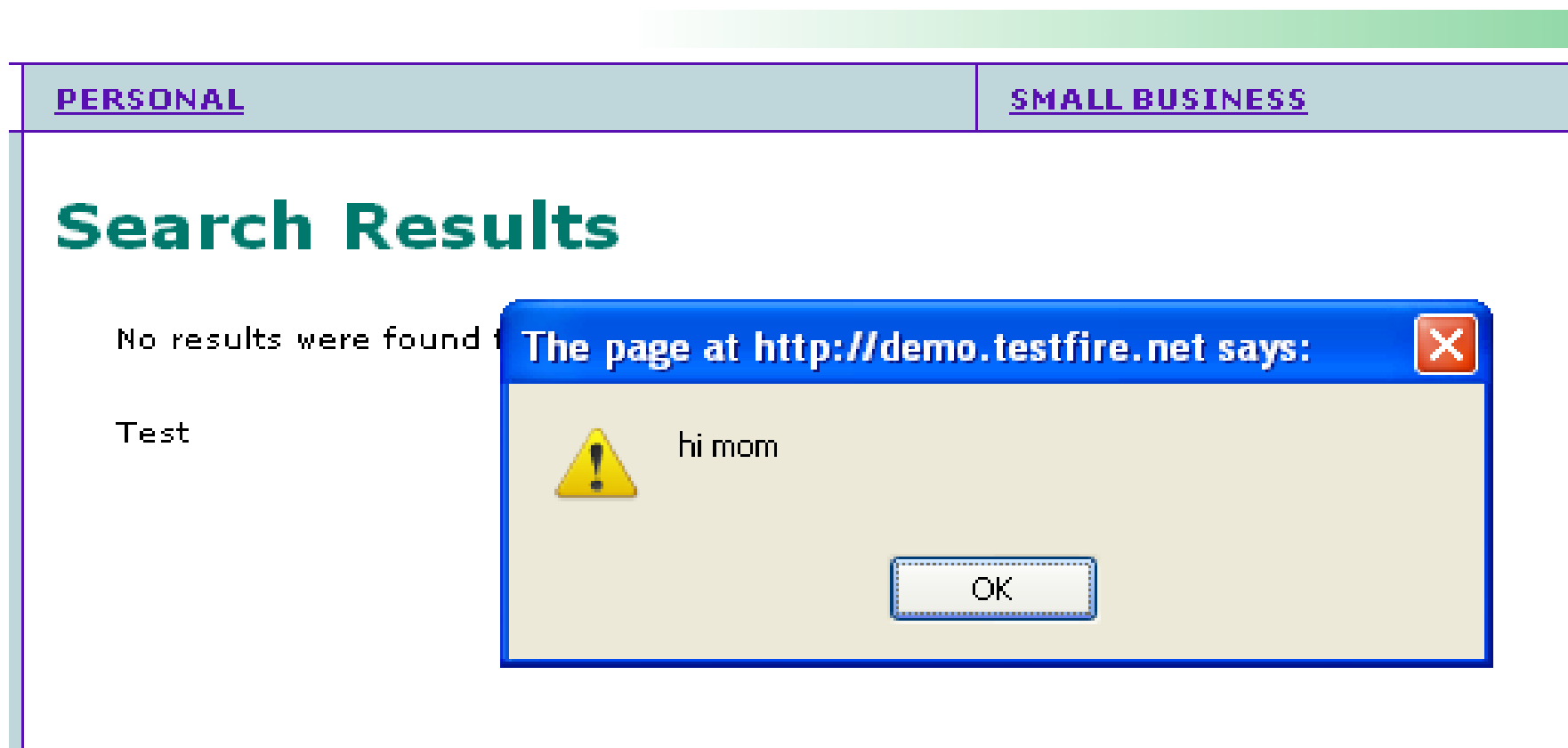
➤ RSnake XSS Cheatsheet

➤ <http://ha.ckers.org/xss.html>

# Web Application Vulnerabilities

## ➤ Cross-Site Scripting (XSS)

➤ Demo – AltoroMutual Search – Reflective XSS



# Web Application Vulnerabilities

## ➤ **Cross-Site Scripting (XSS)**

- Demo – WebGoat XSS Lesson – Stored XSS
  - Start WebGoat – WebGoat 5.3
    - <http://127.0.0.1:8080/webgoat/attack>
    - user = guest, password = guest
    - Restart XSS Lesson



# Web Application Vulnerabilities

## ► Cross-Site Scripting (XSS)

### ► Demo – WebGoat XSS Lesson – Stored XSS

Logout ?

### LAB: Cross Site Scripting

OWASP WebGoat V5.2

◀ Hints ▶ Show Params Show Cookies Lesson Plan Show Java Solution

Introduction

General

Access Control Flaws

AJAX Security

Authentication Flaws

Buffer Overflows

Code Quality

Concurrency

Cross-Site Scripting (XSS)

Denial of Service

**Solution Videos**

Restart this Lesson

Stage 2: Block Stored XSS using Input Validation.

**THIS LESSON ONLY WORKS WITH THE DEVELOPER VERSION OF WEBGOAT**

Implement a fix to block the stored XSS before it can be written to the database. For example, if the user enters 'Eric' as the manager, the system should verify that 'David' is not the manager.

**\* You have completed Stored XSS.**  
**\* Welcome to Block Stored XSS using Input Validation**

**Goat Hills Financial Human Resources**

Welcome Back **Jerry**

Windows Internet Explorer

hi mom

OK

# Web Application Vulnerabilities

## ➤ **Cross-Site Scripting (XSS)**

### ➤ Attack Vectors

- Steal Cookies
- Key Logging
- Port Scanning internal network
- Steal Clipboard contents
- Browser Exploits
- Redirect to other malicious content
- Age of JavaScript attacks
  - JINX – JavaScript Malware Framework
  - Browser Exploitation Framework (BeEF)



# Web Application Vulnerabilities

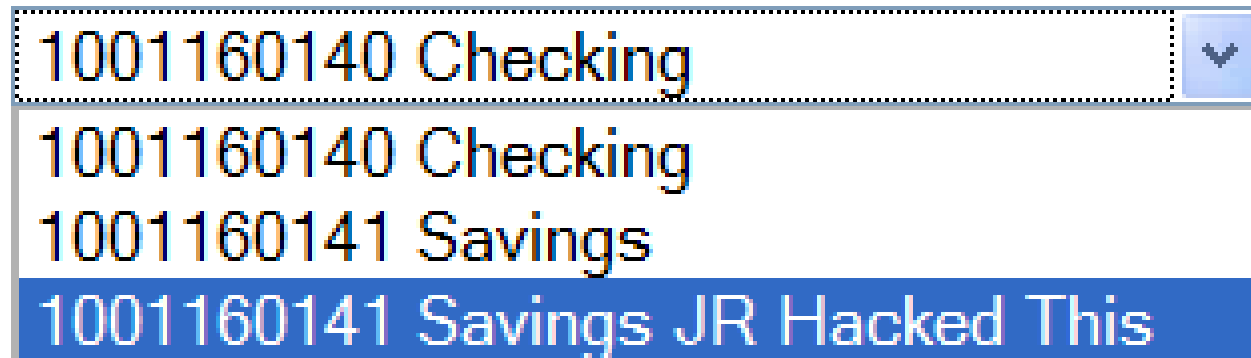
## ➤ Cross-Site Scripting (XSS)

### ➤ Defenses – Input Validation

➤ Use Standard input validation mechanism.

➤ Validate all input data for length, type, syntax, and business rules before trusting data.

➤ Warn about user entered input data.



1001160140 Checking

1001160140 Checking

1001160141 Savings

1001160141 Savings JR Hacked This

# Web Application Vulnerabilities

## ➤ **Cross-Site Scripting (XSS)**

- Defenses – Input Validation.

- OWASP Input Validation Strategies

- Accept Known Good

- Reject Known Bad

- Sanitize

- No Action – Best way to make the news.

- Maybe a combination of first three.

# Web Application Vulnerabilities

## ➤ Cross-Site Scripting (XSS)

### ➤ Defenses – Output Encoding

- Ensure that all user-supplied data is appropriately entity encoded (either HTML or XML depending on the output mechanism) before rendering to client.

- “<” and “>” = “&lt;” and “&gt;”

- Specify the output encoding (such as ISO 8859-1 or UTF 8). Do not allow the attacker to choose this for your users.

# Web Application Vulnerabilities

## ➤ **Cross-Site Scripting (XSS)**

### ➤ Defenses – Issues

- Use Input Validation and Output Encoding together.
  - Don't allow hostile content to live in DB.
- Perform Input Validation Close to “Front Door”
  - Need to handle error processing.
- Blend Input Validation Strategies
  - Accept Known Good for everything but free form text fields. Use Reject Known Bad these fields.

# Web Application Vulnerabilities

## ➤ SQL Injection

- OWASP Rank/Description

  - #2 OWASP Top Ten

  - Attack against server

  - Common vulnerability

  - Occurs when user-supplied data is sent to an command/query interpreter.

  - Hostile data tricks the interpreter into executing unintended commands.

# Web Application Vulnerabilities

## ➤ SQL Injection

- Demo - AltoroMutual Authentication Bypass
  - Bypass Authentication
    - Single Quote Test
      - Talk directly to Interpreter
      - Lots of Information Leakage
      - Yum!



# Web Application Vulnerabilities

## ➤ SQL Injection

- Demo – AltoroMutual Authentication Bypass
- Information Leakage

**Altoro Mutual: Server Error**

SQL Injection - OWASP | SQL injection - Wikipedia, the free ency... | escaping data hate irish - Google Search

[Sign In](#) | [Contact Us](#) | [Feedback](#) | Search

**AltoroMutual**

**DEMO SITE ONLY**

### An Error Has Occurred

**Summary:**

Syntax error in string in query expression 'username = 'admin' AND password = '''.

**Error Message:**

System.Data.OleDb.OleDbException: Syntax error in string in query expression 'username = 'admin' AND password = '''. at System.Data.OleDb.OleDbCommand.ExecuteNonQueryErrorHandling(OleDbHResult hr) at System.Data.OleDb.OleDbCommand.ExecuteNonQueryForSingleResult(tagDBPARAMS dbParams, Object& executeResult) at System.Data.OleDb.OleDbCommand.ExecuteNonQuery(Object& executeResult) at System.Data.OleDb.OleDbCommand.ExecuteReaderInternal(CommandBehavior behavior, Object& executeResult) at System.Data.OleDb.OleDbCommand.ExecuteReader(CommandBehavior behavior, String method) at System.Data.OleDb.OleDbCommand.ExecuteReader(CommandBehavior behavior) at System.Data.Common.DbDataAdapter.FillInternal(DataSet dataset, DataTable[] datatables, Int32 startRecord, Int32 maxRecords, String srcTable, IDbCommand command, CommandBehavior behavior) at System.Data.Common.DbDataAdapter.Fill(DataSet dataSet, Int32 startRecord, Int32 maxRecords, String srcTable, IDbCommand command, CommandBehavior behavior) at System.Data.Common.DbDataAdapter.Fill(DataSet dataSet, String srcTable) at Altoro.Authentication.ValidateUser(String uName, String pWord) in d:\downloads\AltoroMutual\_v6\website\bank\login.aspx.cs:line 68 at Altoro.Authentication.Page\_Load(Object sender, EventArgs e) in d:\downloads\AltoroMutual\_v6\website\bank\login.aspx.cs:line 33 at System.Web.Util.CalliHelper.EventArgFunctionCaller(IntPtr fp, Object o, Object t, EventArgs e) at System.Web.Util.CalliEventHandlerDelegateProxy.Callback(Object sender, EventArgs e) at System.Web.UI.Control.OnLoad(EventArgs e) at System.Web.UI.Control.LoadRecursive() at System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint)

# Web Application Vulnerabilities

## ➤ SQL Injection

- Demo - AltoroMutual Authentication Bypass

- Bypass Authentication

- Exploit Steps

- Username: admin

- Password: abc' or 1=1;--

- Result: Hello Admin User



# Web Application Vulnerabilities

## ➤ SQL Injection

### ➤ Demo – AltoroMutual Authentication Bypass

The screenshot shows the AltoroMutual Online Banking Home page. The header includes the AltoroMutual logo and navigation links: Sign Off, Contact Us, Feedback, and a search bar. The main content area is divided into four tabs: MY ACCOUNT, PERSONAL, SMALL BUSINESS, and INSIDE ALTORO MUTUAL. The PERSONAL tab is selected, displaying a welcome message for 'Admin User' and a 'View Account Details' button. A red circle highlights the 'Admin User' text and the 'View Account Details' button. The footer contains links for Privacy Policy, Security Statement, and copyright information for Watchfire, Inc.

Altoro Mutual: Online Banking Home

Sign Off | Contact Us | Feedback | Search  Go

**AltoroMutual**

**MY ACCOUNT** | **PERSONAL** | **SMALL BUSINESS** | **INSIDE ALTORO MUTUAL**

**I WANT TO ...**

- [View Account Summary](#)
- [View Recent Transactions](#)
- [Transfer Funds](#)
- [Search News Articles](#)
- [Customize Site Language](#)

**ADMINISTRATION**

- [View Application Values](#)
- [Edit Users](#)

**Hello Admin User**

Welcome to Altoro Mutual Online.

View Account Details:

[Privacy Policy](#) | [Security Statement](#) | © 2010 Altoro Mutual, Inc.

The Altoro Mutual website is published by Watchfire, Inc. for the sole purpose of demonstrating the effectiveness of Watchfire products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. Watchfire does not assume any risk in relation to your use of this website. For additional Terms of Use, please go to <http://www.watchfire.com/statements/terms.aspx>.

Copyright © 2010, Watchfire Corporation, All rights reserved.

# Web Application Vulnerabilities

## ➤ SQL Injection

### ➤ Little Bobby Tables



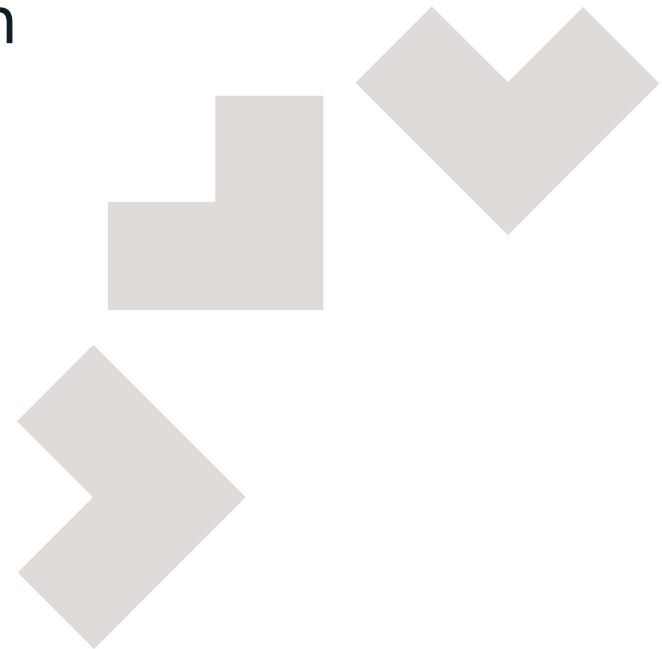
Courtesy of XKCD -  
<http://xkcd.com/327/>

# Web Application Vulnerabilities

## ➤ **SQL Injection**

### ➤ Attack Vectors

- Unauthorized Data Access
- Unauthorized Data Modification
- Database Destruction
- OS Command Execution
  - Yee Haw!



# Web Application Vulnerabilities

## ➤ SQL Injection

### ➤ Defenses

#### ➤ Best

#### ➤ SQL Prepared Statements

➤ Placeholder Substitution Markers

➤ User Input is Strongly Typed.



# Web Application Vulnerabilities

## ➤ SQL Injection

### ➤ Defenses

#### ➤ Next Best

➤ Escape Data. If possible, escape data before it is sent to the database query engine.

➤ Escaping is bad for the Irish.

➤ O'Malley -> OMalley

➤ Input Validation

➤ Same as Cross-Site Scripting.

# Web Application Vulnerabilities

## ➤ SQL Injection

### ➤ Defenses

- Enforce least privilege when connecting to databases and other backend systems.
- Avoid detailed error messages that are useful to an attacker.
- Stored procedures can also be injected.

# Web Application Vulnerabilities

## ➤ Vulnerabilities – Malicious File Execution

### ➤ GIFAR – Code Smuggling

- Uploaded User Generated Content
- Billy Rios/Nathan McFeters at BlackHat
- GIF Image file combined with Java Archive file (JAR) = GIFAR
- GIF stores ident/control at front of file
- JAR (ZIP) stores ident/control at end of file

# Web Application Vulnerabilities

## ➤ **Top Web Application Attacks**

- Top Web Application Attacks 2008/2009

- 2008

- <http://jeremiahgrossman.blogspot.com/2008>

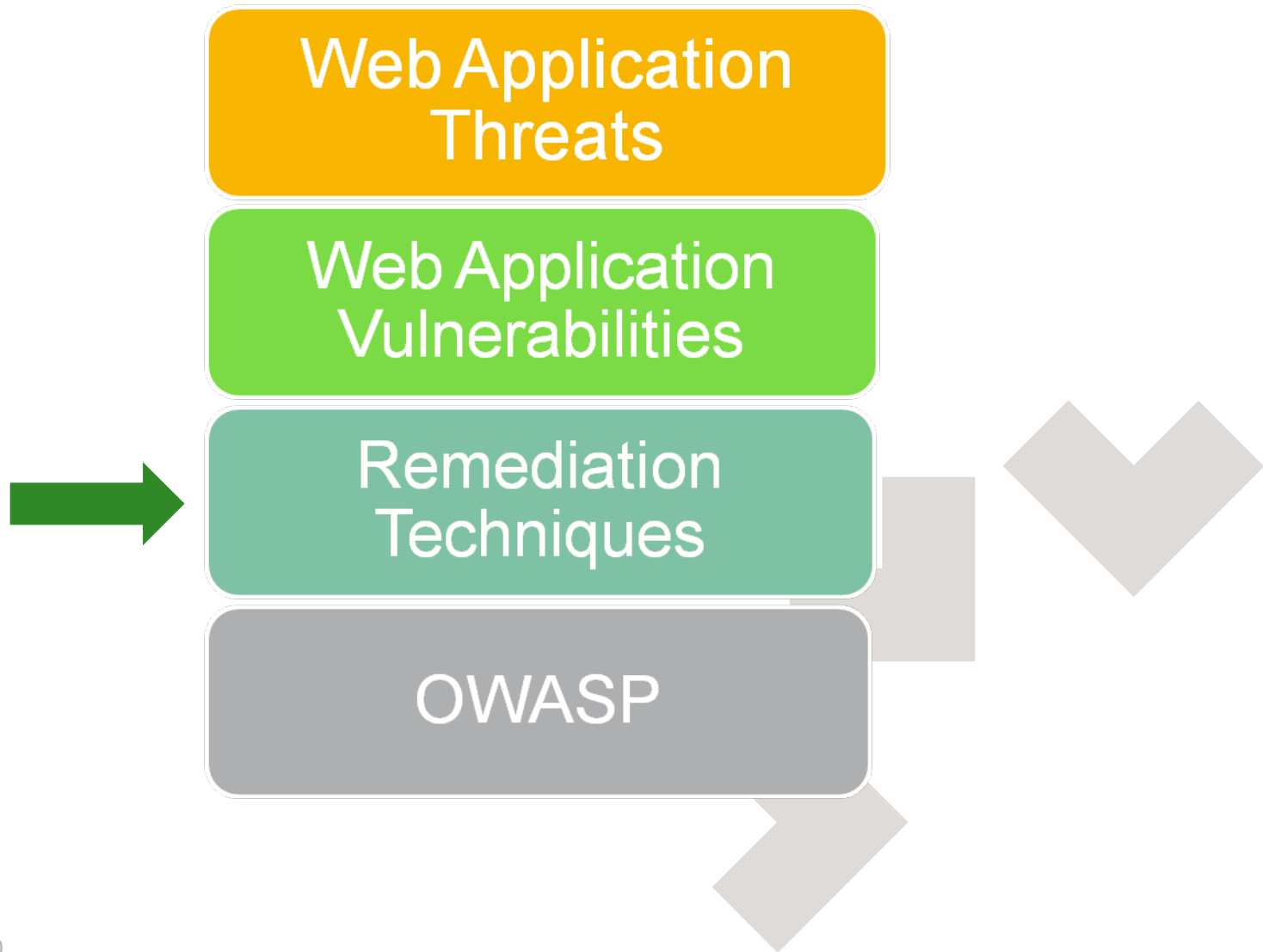
- 2009

- <http://jeremiahgrossman.blogspot.com/2009>





# Checkpoint



# Remediation Techniques

- No Silver Bullet
- Security & Software Development Lifecycle (SDLC)
- Static Source Scanning
- Dynamic Scanning
- Vulnerability Tracking
- Developer Training & Awareness
- Infrastructure

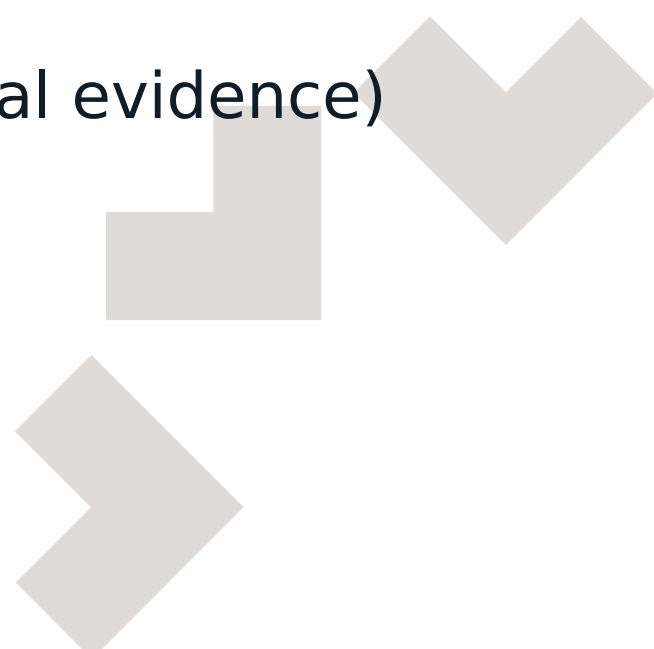
# Remediation Techniques

- **No Silver Bullet**
  - Plan
  - Resources
    - Hardware
    - Software
    - Human
  - Hard Work
    - You will see results



# Remediation Techniques

- **Security & Software Development Lifecycle (SDLC)**
  - Security Tasks in the SDLC
  - Developer Training for Security Tasks in the SDLC
    - Less than 10% (empirical evidence)



# Remediation Techniques

## ➤ **SDLC - Security Tasks**

### ➤ Planning

- Security Task Durations
- Plan to address outstanding security issues

### ➤ Requirements

- Reference Security Policy
- Authentication
- Cryptography
- Protect Sensitive Data in transit
- Protect Sensitive Data at rest

# Remediation Techniques

- ▶ SDLC – Security Tasks
  - ▶ Design
    - ▶ Security Reviews Design Documents
  - ▶ Development
    - ▶ Static Scanning – Developer, Build, Ad-Hoc
  - ▶ QA/UAT
    - ▶ Dynamic Scanning – Manual and Automated
  - ▶ Implementation/Deployment/Operational
    - ▶ System Object Security

# Remediation Techniques

## ➤ **Static Source Scanning**

- Overview
- Developer vs Build vs Ad Hoc Scans
- Issue Management Strategy
- Hard part is Enterprise Implementation



# Remediation Techniques

## ➤ **Static Source Scanning**

### ➤ Issue Management Strategy

- Turn on all checkers

- Initial Scan = 50K+ Issues

- Fix top Critical and High

- Defer all others until next version

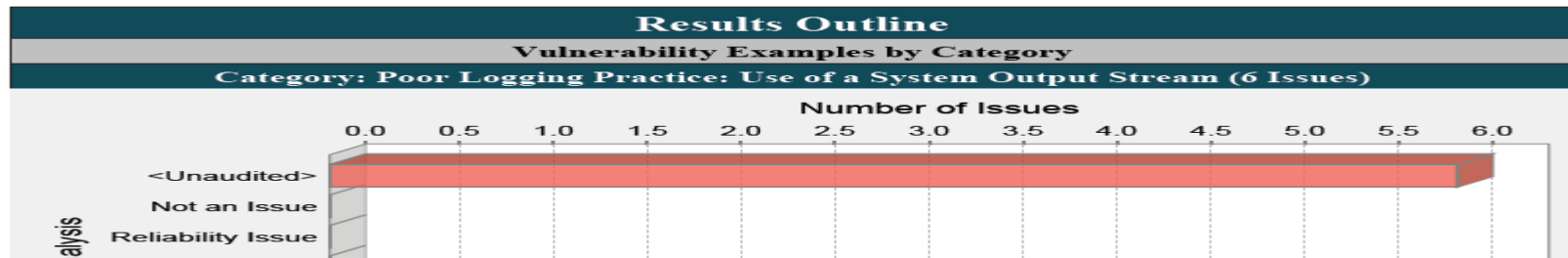
- No new issues during development



# Remediation Techniques

## ► Static Source Scanning

### ► Fortify 360 Demo



HelloWorldJava1.java, line 21 (Poor Logging Practice: Use of a System Output Stream)			
Fortify Priority:	Low	Folder	Low
Kingdom:	Encapsulation		
Abstract:	Using System.out or System.err rather than a dedicated logging facility makes it difficult to monitor the behavior of the program.		
Sink:	HelloWorldJava1.java:21 FunctionCall: println()		
19	int intVar;		
20			
21	System.out.println("Hello World Java 1");		
22	System.out.println();		
23	System.out.println("Argument 1 = " + args[0]);		

# Remediation Techniques

## ➤ Dynamic Scanning

### ➤ Manual

#### ➤ Intercepting Proxy - Burp

### ➤ Automated

#### ➤ IBM Rational AppScan

#### ➤ Skipfish – New, Google

#### ➤ 404 Generator

➤ "Give a man an audit and he will be secure for a day. Teach a man to audit and he will be secure for the rest of his life." David Rhoades (**SANS**)

# Remediation Techniques

## ➤ Dynamic Scanning

### ➤ Automated - IBM Rational AppScan

The screenshot shows the IBM Rational AppScan interface. The top menu bar includes File, Edit, View, Scan, Tools, and Help. The main window displays a list of 94 security issues for 'My Application'. The issues are arranged by severity (Descending) and include categories like Authentication Bypass Using SQL Injection, Blind SQL Injection, Cross-Site Scripting, and Database Error Pattern Found.

The left sidebar shows a tree view of the application structure, including files like comment.aspx, default.aspx, disclaimer.htm, feedback.aspx, search.aspx, servererror.aspx, subscribe.aspx, subscribe.swf, survey\_questions.aspx, admin, bank, images, and pr.

The bottom left corner features a 'Dashboard' with an 'Issue Severity Gauge' showing the total number of issues: 94. The gauge is divided into four categories: 38 (High), 22 (Medium), 20 (Low), and 14 (Info).

The main content area displays a detailed view of a specific issue (ID: 6348). The issue is titled 'GET /search.aspx?txtSearch=1234<script>alert(17133)</script> HTTP/1.0'. The description states: 'Set parameter value to: <script>alert(17133)</script>'. The difference shows the injected payload: 'Injected "1234<script>alert(17133)</script>" into parameter "txtSearch" value'. The reasoning states: 'The test successfully embedded a script in the response, and it will be executed once the page is loaded'.

The bottom status bar shows 'Visited URLs 98/98', 'Completed Tests 18274/18274', and '94 Security Issues'.

# Remediation Techniques

- Vulnerability Tracking
  - Open Source Vulnerability Database
    - <http://osvdb.org/>
  - NIST National Vulnerability Data
    - <http://nvd.nist.gov/>
  - Secunia
    - <http://secunia.com/advisories/search/>
  - IBM Internet Security Systems (ISS) X-Force
    - <http://xforce.iss.net/>
  - Symantec Deep Sight Threat Management
    - <https://tms.symantec.com/Default.aspx>

# Remediation Techniques

## ➤ Vulnerability Tracking

### ➤ NIST National Vulnerability Database (NVD)

#### National Cyber-Alert System

##### Vulnerability Summary for CVE-2009-2902

**Original release date:** 01/28/2010

**Last revised:** 04/01/2010

**Source:** US-CERT/NIST

#### Overview

Directory traversal vulnerability in Apache Tomcat 5.5.0 through 5.5.28 and 6.0.0 through 6.0.20 allows remote attackers to delete work-directory files via directory traversal sequences in a WAR filename, as demonstrated by the ...war filename.

#### Impact

CVSS Severity (version 2.0):

**CVSS v2 Base Score:** 4.3 (MEDIUM) (AV:N/AC:M/Au:N/C:N/I:P/A:N) (legend)

**Impact Subscore:** 2.9

**Exploitability Subscore:** 8.6

CVSS Version 2 Metrics:

**Access Vector:** Network exploitable; Victim must voluntarily interact with attack mechanism

**Access Complexity:** Medium

**Authentication:** Not required to exploit

**Impact Type:** Allows unauthorized modification

# Remediation Techniques

## ‣ Training & Awareness

- OWASP

- SANS/Mitre top 25 Security Errors report

  - <http://www.sans.org/top25-programming-errors>

- CERT Secure Coding

  - <https://www.securecoding.cert.org/confluence/>

- Google Browser Security Handbook - Part 1

  - <http://code.google.com/p/browsersec/wiki/Part1>

- Google Browser Security Handbook - Part 2

  - <http://code.google.com/p/browsersec/wiki/Part2>

# Remediation Techniques

## ➤ Training & Awareness

- Create Training & Awareness Courses
  - Present to Developers and Architects
  - In person
  - Content
    - What Security Engineering/Assurance does
    - Regulatory
    - Web Application
    - Defensive Coding
    - Etc.



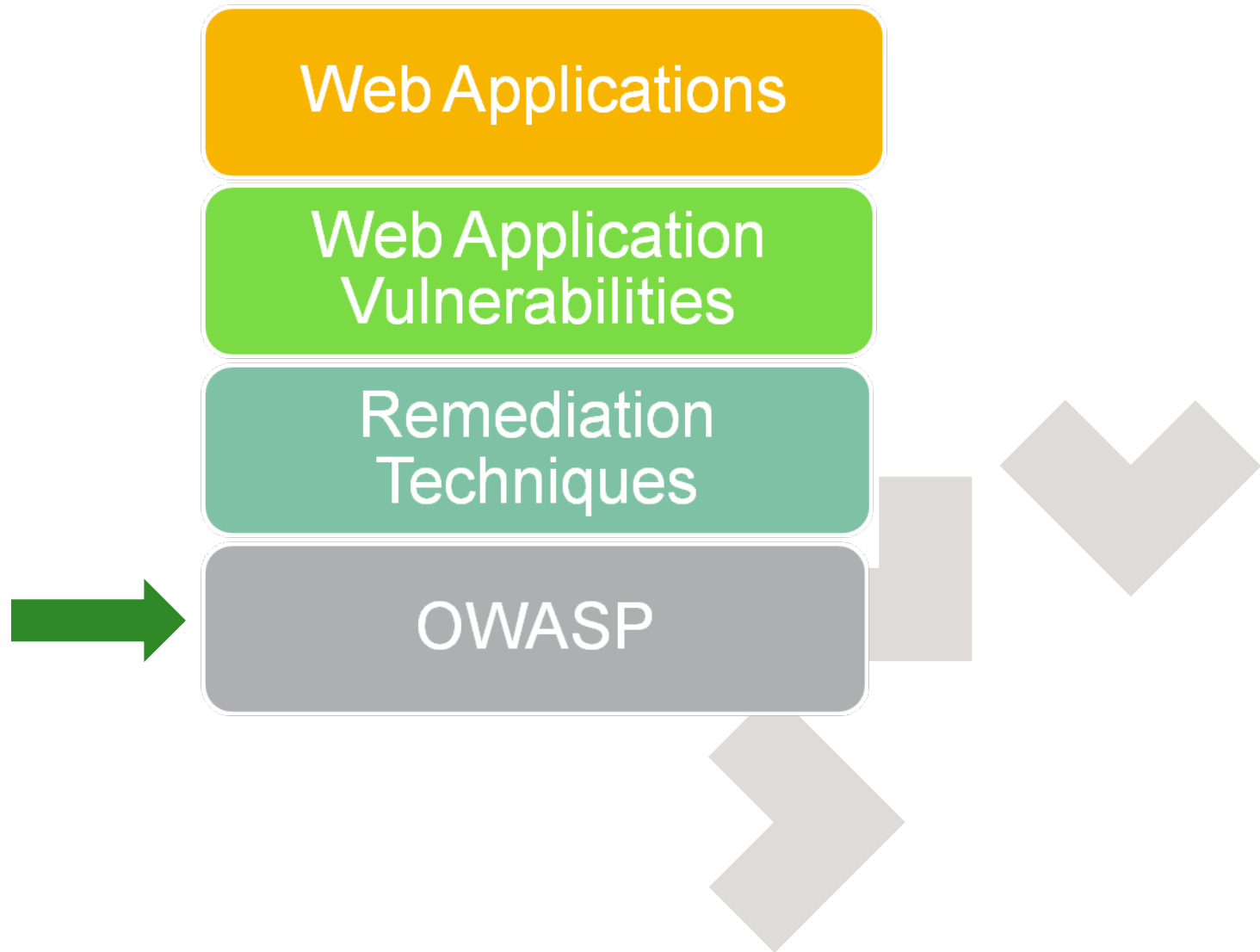
# Remediation Techniques

## ➤ Infrastructure

- System Object Security
  - File Access
  - J2EE Container Security
- Web Application Firewalls
  - Quick and Easy = 1 week purchase to deployment
  - Need to tune for each Web Application
  - Nice defense in depth posture
  - PCI Approved



# Checkpoint



# OWASP

## ➤ **Open Web Application Security Project (OWASP)** - <http://www.owasp.org>

- Description

- Projects

- Guides – Development/Testing

- Tools – WebScarab

- Education – WebGoat

- Awareness – Top Ten

## ➤ Description

- The Open Web Application Security Project (OWASP) is a worldwide free and open community focused on improving the security of application software.
- Make application security "visible"
- All materials and software are open source.

## ➤ Projects

### ➤ Guides - Development

➤ Architects, developers, consultants and auditors

➤ Comprehensive manual for designing, developing and deploying secure web applications.



# OWASP

## ➤ Projects

### ➤ Guides - Testing

#### ➤ "Best Practices" penetration testing framework

#### ➤ URL:

<http://www.owasp.org/index.php/Category:OWASP>

# OWASP

## ➤ Projects

### ➤ Tools - WebScarab

➤ Framework for analyzing applications that communicate using HTTP and HTTPS protocols.

➤ Includes intercepting proxy, spider, fuzzer, etc. Written in Java, portable to many platforms.

➤ URL:

<http://www.owasp.org/index.php/Category:OWASP>

# OWASP

## ➤ Projects

### ➤ Education - WebGoat

- Deliberately insecure J2EE web application maintained by OWASP

### ➤ Web Application security lessons

### ➤ URL:

<http://www.owasp.org/index.php/Category:OWASP>

# OWASP

## ➤ Projects

- Awareness - Top Ten (2007 & 2010)

  - Powerful awareness document

  - Broad consensus about most critical web application security flaws

  - Project members include a variety of security experts from around the world

  - h

    - [http://www.owasp.org/index.php/Top\\_10\\_2007](http://www.owasp.org/index.php/Top_10_2007)



# Presentation Message

- **Don't trust the client**
- **Scary and Dangerous Out There**
- **No Silver Bullet, Hard Work**



# Question and Answers

*Questions?*



# References

## ➤ OWASP

### ➤ Main

➤ <http://www.owasp.org>

### ➤ Top Ten

➤ h

[http://www.owasp.org/index.php/Top\\_10\\_2007](http://www.owasp.org/index.php/Top_10_2007)

### ➤ Cross-Site Scripting

➤ <http://>

[www.owasp.org/index.php/Data\\_Validation](http://www.owasp.org/index.php/Data_Validation)

➤ [http://www.owasp.org/index.php/XSS\\_%28Cro](http://www.owasp.org/index.php/XSS_%28Cro)

# References

## ➤ OWASP

### ➤ Output Encoding

➤ [http://www.owasp.org/index.php/How\\_to\\_perform](http://www.owasp.org/index.php/How_to_perform)

### ➤ SQL Injection

➤ [http://www.owasp.org/index.php/SQL\\_Injection](http://www.owasp.org/index.php/SQL_Injection)

### ➤ WebGoat

➤ URL:

[http://www.owasp.org/index.php/Category:OWASP\\_WebGoat](http://www.owasp.org/index.php/Category:OWASP_WebGoat)

### ➤ Phoenix Tools

➤ <http://www.owasp.org/index.php/Phoenix/Tools>

# References

## ➤ OWASP

### ➤ Development Guide

➤ [http://www.owasp.org/index.php/Category:OWASP\\_Development\\_Guide](http://www.owasp.org/index.php/Category:OWASP_Development_Guide)

### ➤ Testing Guide

➤ URL:

[http://www.owasp.org/index.php/Category:OWASP\\_Testing\\_Guide](http://www.owasp.org/index.php/Category:OWASP_Testing_Guide)

### ➤ WebScarab

➤ URL:

[http://www.owasp.org/index.php/Category:OWASP\\_WebScarab](http://www.owasp.org/index.php/Category:OWASP_WebScarab)

# References

- SANS

- Main

- <http://www.sans.org>

- SANS Report – September 2009

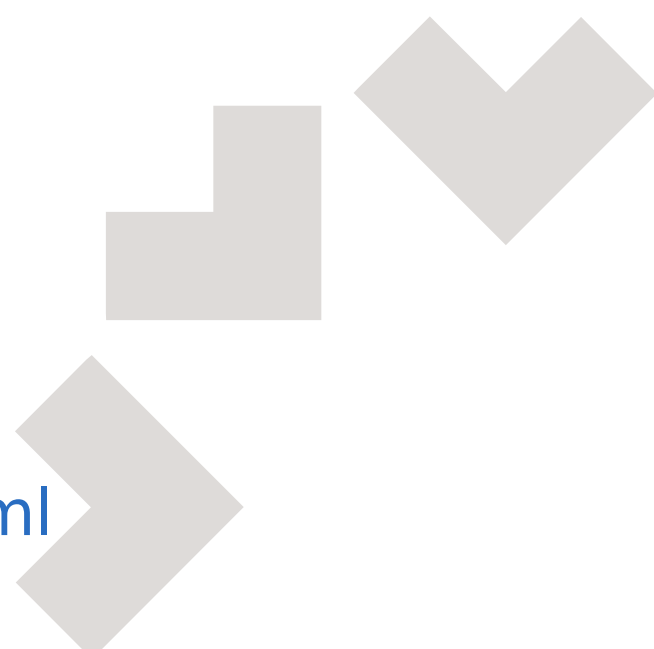
- <http://www.sans.org/top-cyber-security-risks/>

- AltoroMutual Test Site

- <http://demo.testfire.net>

- RSnake XSS Cheatsheet

- <http://ha.ckers.org/xss.html>



# References

## ➤ XKCD

- XKCD – <http://xkcd.com/327/>

## ➤ GIFAR

- Billy Rios

- <http://xs-sniper.com/blog/2008/12/17/sun-fixes-gifars/>

- How to create a GIFAR

- <http://riosec.com/how-to-create-a-gifar>

- More on GIFARs and Other Java Smuggling

- <http://riosec.com/more-on-gifars-and-other-java>

# References

- SQL Attacks By Example

- <http://www.unixwiz.net/techtips/sql-injection.html>

- Top Web Application Attacks 2008/2009

- 2008

- <http://jeremiahgrossman.blogspot.com>

- 2009

- <http://jeremiahgrossman.blogspot.com>



# References

- Vulnerability Tracking
  - Open Source Vulnerability Database
    - <http://osvdb.org/>
  - NIST National Vulnerability Data
    - <http://nvd.nist.gov/>
  - Secunia
    - <http://secunia.com/advisories/search/>
  - IBM Internet Security Systems (ISS) X-Force
    - <http://xforce.iss.net/>
  - Symantec Deep Sight Threat Management
    - <https://tms.symantec.com/Default.aspx>

A thick red horizontal bar at the top of the slide, with a white L-shaped cutout on the left side.

# Thank You!

