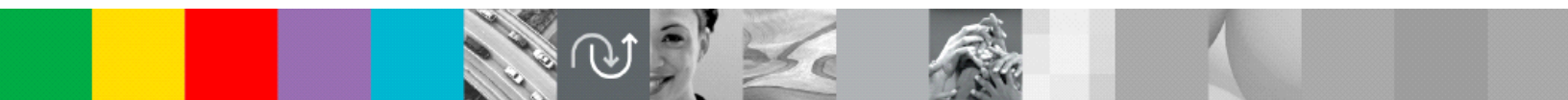IBM Software Group

An IBM Proof of Technology

Discovering the value of
Web Application Security Testing
with IBM Rational AppScan

Presentation

PoT.Rational.07.2.035.05

Discovering the Value of Verifying Web Application Security Using IBM Rational AppScan

An IBM Proof of Technology

© 2009 IBM Corporation

Agenda

- **Introductions & facilities**
- Security Landscape
- Vulnerability Analysis
  - ▸ Top Attacks Overview
  - ▸ Hands on Lab 1
- Vulnerability Analysis (continued)
  - ▸ Hands on Lab 2
- Automated Vulnerability Analysis
  - ▸ IBM® Rational® AppScan Overview
  - ▸ Hands on Lab 3

Welcome to the Technical Exploration Center

- Introductions
- Access restrictions
- Restrooms
- Emergency Exits
- Smoking Policy
- Breakfast/Lunch/Snacks – location and times
- Special meal requirements?

POT Objectives

By the end of this session you will:

- Understand the Web application environment
- Understand and differentiate between network and application level vulnerabilities
- Understand where the vulnerabilities exist
- Understand how to leverage AppScan to perform an automated scan for vulnerabilities

Introductions

- Please introduce yourself

- Name and organization

- Current integration technologies/tools in use
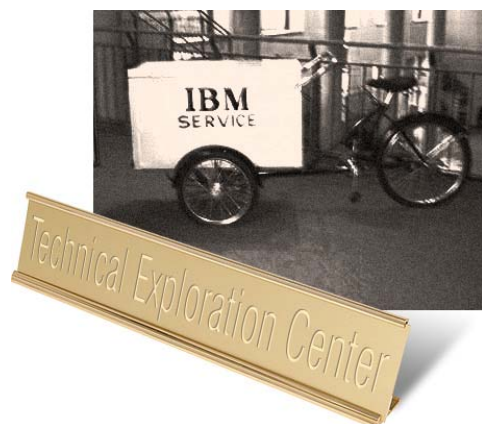
What do you want out of this Exploration session?

Agenda

- Introductions & facilities
- **Security Landscape**
- Vulnerability Analysis
  - ▸ Top Attacks Overview
  - ▸ Cross Site Scripting
  - ▸ Hands on Lab 1
- Vulnerability Analysis (continued)
  - ▸ Hands on Lab 2
- Automated Vulnerability Analysis
  - ▸ AppScan Overview
  - ▸ Hands on Lab 3

The Alarming Truth

"Approximately 100 million Americans have been informed that they have suffered a security breach so this problem has reached epidemic proportions."

Jon Oltsik – Enterprise Strategy Group

"Up to 21,000 loan clients may have had data exposed"

Marcella Bombardieri, Globe Staff/August 24, 2006

"Personal information stolen from 2.2 million active-duty members of the military, the government said…"
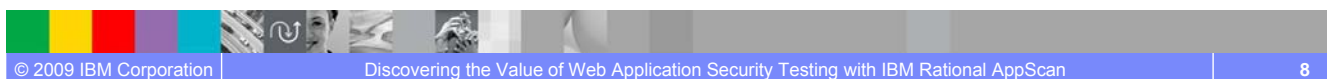
New York Times/June 7, 2006

"Hacker may have stolen personal identifiable information for 26,000 employees.."
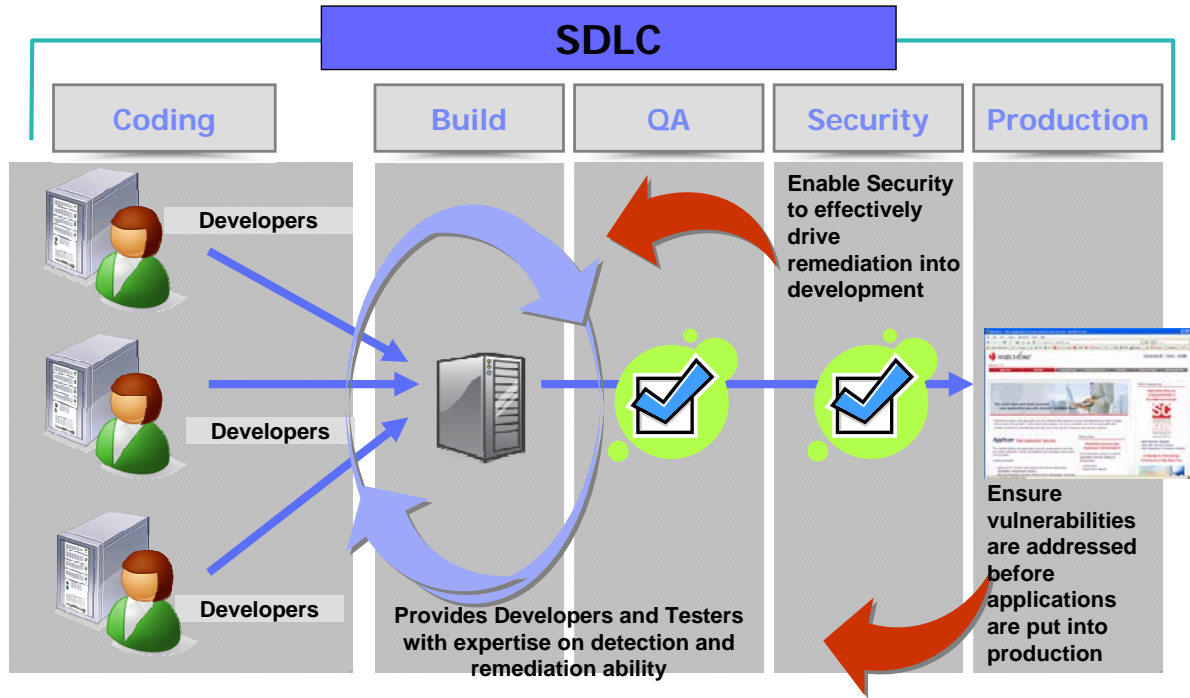
ComputerWorld, June 22, 2006
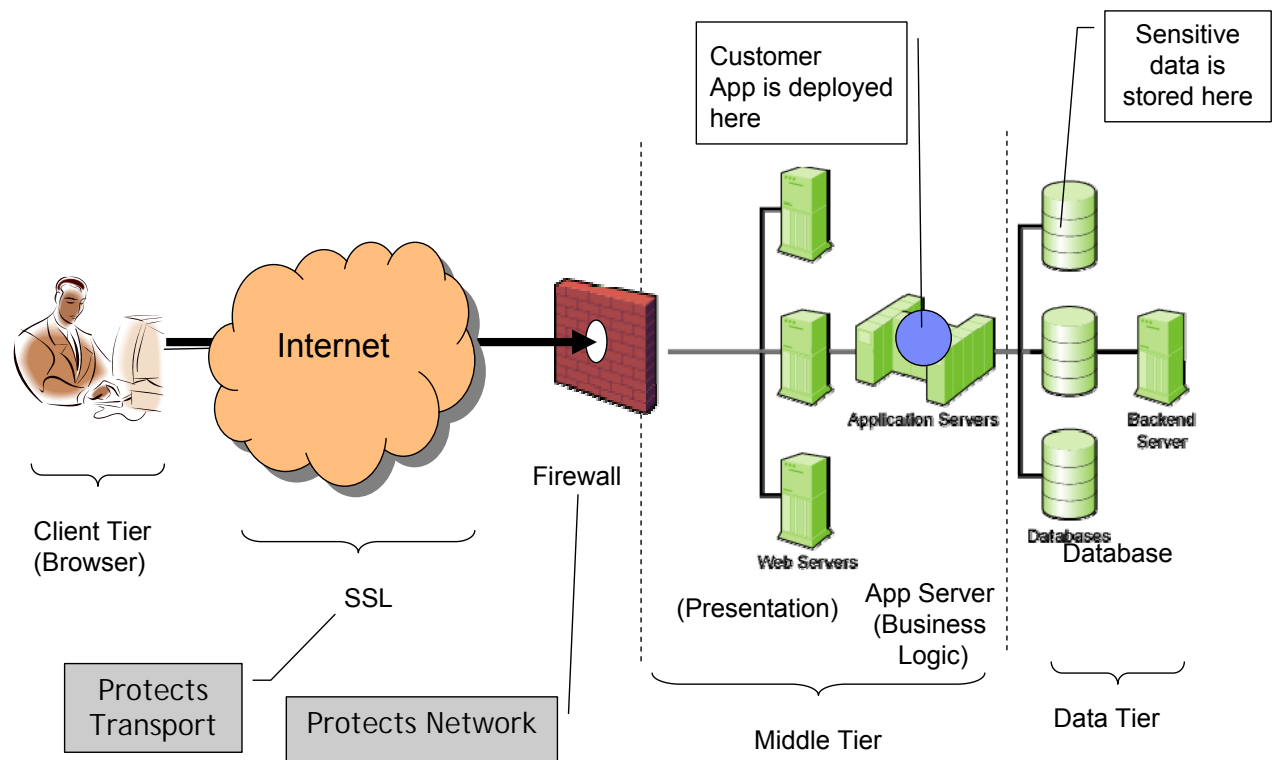

Why Application Security is a High Priority

- **Web applications are the #1 focus of hackers:**
  - 75% of attacks at Application layer (Gartner®)
  - XSS and SQL Injection are #1 and #2 reported vulnerabilities (Mitre®)

- **Most sites are vulnerable:**
  - 90% of sites are vulnerable to application attacks (Watchfire®)
  - 78% percent of easily exploitable vulnerabilities affected Web applications (Symantec™)
  - 80% of organizations will experience an application security incident by 2010 (Gartner)

- **Web applications are high value targets for hackers:**
  - Customer data, credit cards, ID theft, fraud, site defacement, etc

- **Compliance requirements:**
  - Payment Card Industry (PCI) Standards, GLBA, HIPPA, FISMA,

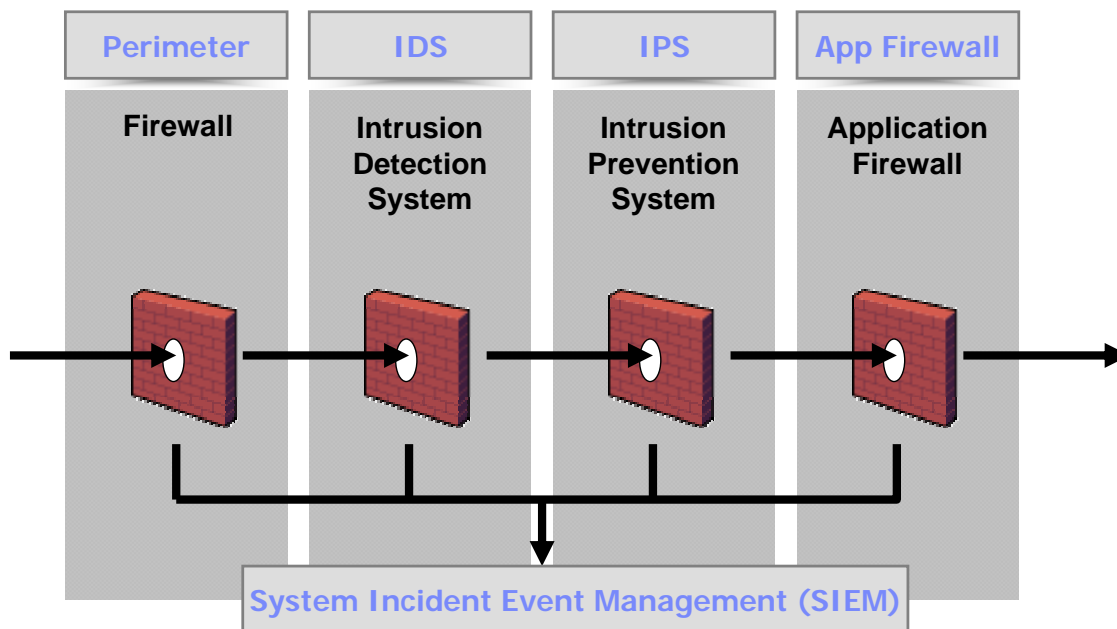# Building Security & Compliance into the Software Development Lifecycle (SDLC)

SDLC

Coding	Build	QA	Security	Production

Developers

Developers

Developers

Enable Security to effectively drive remediation into development

Ensure vulnerabilities are addressed before applications are put into production

Provides Developers and Testers with expertise on detection and remediation ability

---

High Level Web Application Architecture Review

Customer App is deployed here

Sensitive data is stored here

Internet

Firewall

Application Servers

Backend Server

Web Servers

Databases
Database

Client Tier (Browser)

SSL

Protects Transport

Protects Network

(Presentation)

App Server (Business Logic)

Middle Tier

Data Tier

Network Defenses for Web Applications

Perimeter	IDS	IPS	App Firewall
Firewall	Intrusion Detection System	Intrusion Prevention System	Application Firewall

System Incident Event Management (SIEM)

Agenda

- Introductions & facilities
- Security Landscape
- **Vulnerability Analysis**
  - ▶ Top Attacks Overview
  - ▶ Hands on Lab 1
- Vulnerability Analysis (continued)
  - ▶ Hands on Lab 2
- Automated Vulnerability Analysis
  - ▶ AppScan Overview
  - ▶ Hands on Lab 3

Where are the Vulnerabilities?

Security

Code Scanning Emerging Tech	Client-Side	Custom	Web Services
AppScan DE/BE Fortify Ounce Labs Klockwork Parasoft®		Web Applications	

Third-party Components

Web Server Configuration

Web Server

Database

Applications

Operating System

Network
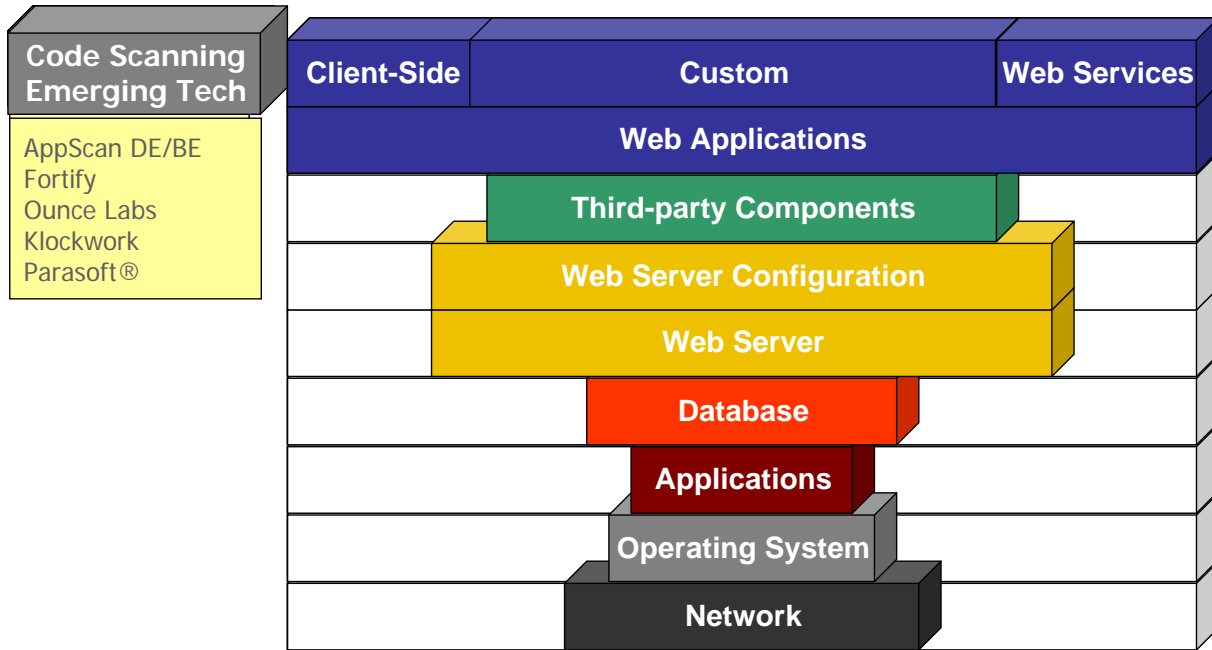
The Myth: "Our Site Is Safe"

Security

We Have Firewalls in Place

We Audit It Once a Quarter with Pen Testers

We Use Network Vulnerability Scanners

The Reality: Security and Spending Are Unbalanced

Security	Spending
% of Attacks	**% of Dollars**

Web Applications

	10%
75%	90%
25%	

Network Server

75% of All Attacks on Information Security Are Directed to the Web Application Layer

2/3 of All Web Applications Are Vulnerable
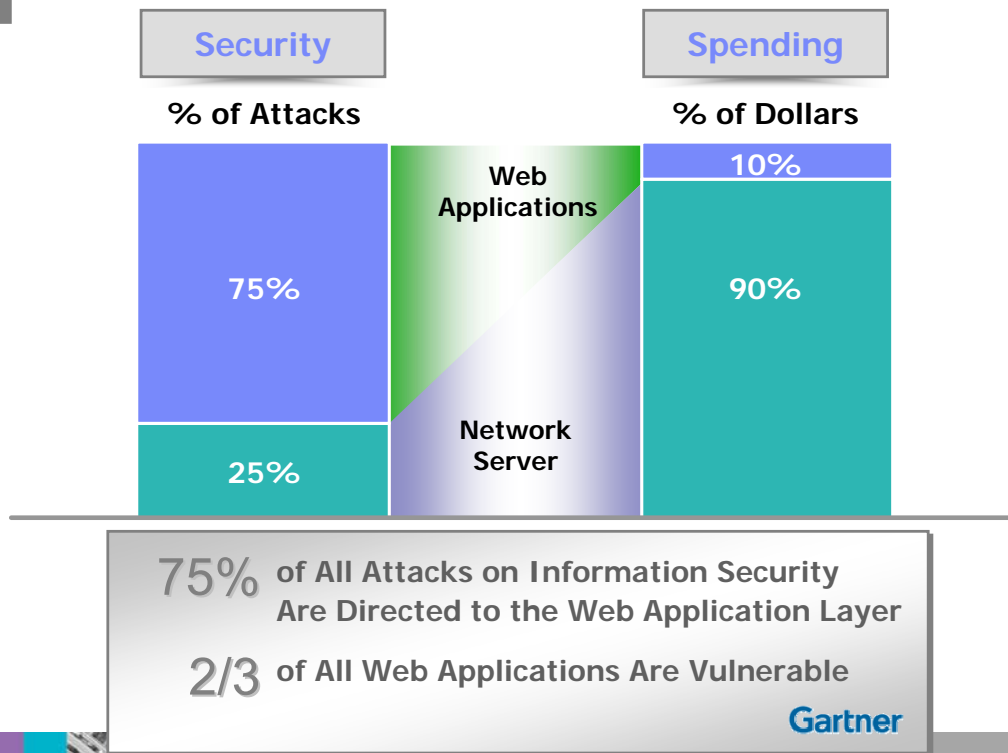
Gartner

Sources: Gartner, Watchfire

What is a Web Application?

Data

Database

Backend Application

Front end Application

User Interface Code

Web Server

User Input
HTML/HTTP

Browser

- **The business logic that enables:**
  - ▸ User's interaction with Web site
  - ▸ Transacting/interfacing with back-end data systems (databases, Change and Release Management(CRM), Packaged Applications etc)
- **In the form of:**
  - ▸ 3rd party packaged software; i.e. Web server, application server, software packages etc.
  - ▸ Code developed in-house / Web builder / system integrator

Input and Output flow through each layer of the application

A break in any layer breaks the whole application

Security Defects: Those I manage vs. Those I own

	Infrastructure Vulnerabilities or Common Web Vulnerabilities (CWVs)	Application Specific Vulnerabilities (ASVs)
Cause of Defect	Insecure application development by **3rd party SW**	Insecure application development **In-house**
Location within Application	3rd party **technical building blocks or infrastructure** (Web servers,)	**Business logic** - dynamic data consumed by an application
Type(s) of Exploits	Known vulnerabilities (patches issued), misconfiguration	SQL injection, path tampering, Cross site scripting, Suspect content & cookie poisoning
Detection	Match signatures & check for known misconfigurations.	Requires application specific knowledge
Business Risk	Patch latency primary issue	Requires automatic application lifecycle security
Cost Control	As secure as 3rd party software	Early detection saves $$$

Open Web Application Security Project (OWASP) and the OWASP Top 10 list

- Open Web Application Security Project (OWASP) – an open organization dedicated to fight insecure software
- "The OWASP Top Ten document represents a broad consensus about what the most critical Web application security flaws are"
- We will use the Top 10 list to cover some of the most common security issues in Web applications

The OWASP Top 10 list

Application Threat	Negative Impact	Example Impact
Cross-Site® scripting	Identity Theft, Sensitive Information Leakage, …	Hackers can impersonate legitimate users, and control their accounts.
Injection Flaws	Attacker can manipulate queries to the DB / LDAP / Other system	Hackers can access backend database information, alter it or steal it.
Malicious File Execution	Execute shell commands on server, up to full control	Site modified to transfer all interactions to the hacker.
Insecure Direct Object Reference	Attacker can access sensitive files and resources	Web application returns contents of sensitive file (instead of harmless one)
Cross-Site Request Forgery	Attacker can invoke "blind" actions on Web applications, impersonating as a trusted user	Blind requests to bank account transfer money to hacker
Information Leakage and Improper Error Handling	Attackers can gain detailed system information	Malicious system reconnaissance may assist in developing further attacks
Broken Authentication & Session Management	Session tokens not guarded or invalidated properly	Hacker can "force" session token on victim; session tokens can be stolen after logout
Insecure Cryptographic Storage	Weak encryption techniques may lead to broken encryption	Confidential information (SSN, Credit Cards) can be decrypted by malicious users
Insecure Communications	Sensitive info sent unencrypted over insecure channel	Unencrypted credentials "sniffed" and used by hacker to impersonate user
Failure to Restrict URL Access	Hacker can access unauthorized resources	Hacker can forcefully browse and access a page past the login page

---

1. Cross-Site Scripting (XSS)

- ● What is it?
  - ▶ Malicious script echoed back into HTML returned from a trusted site, and runs under trusted context

- ● What are the implications?
  - ▶ Session Tokens stolen (browser security circumvented)
  - ▶ Complete page content compromised
  - ▶ Future pages in browser compromised

Demonstration – Cross Site Scripting

- Main points covered in the demo:
  - ▶ Locating an a place where user input which is echoed back to the browser
  - ▶ Seeing if the user input is echoed back 'as-is' or if it is properly encoded
  - ▶ Exploiting the vulnerability

XSS Example I

HTML code:

```
<p>No results were found for the query:<br /><br />
<span id="_ct10__ct10_Content_Main_lblSearch">asdf</span>
```

XSS Example II

earch.aspx?txtSearch=<script>alert(document.cookie)</script>

AltoroMutual

Sign In | Contact Us | Feedback | Search [   ] Go

DEMO SITE ONLY

| ONLINE BANKING LOGIN | PERSONAL | SMALL BUSINESS | INSIDE ALTORO MUTUAL |

PERSONAL
- Deposit Product
- Checking
- Loan Products
- Cards
- Investments & Insurance
- Other Services

SMALL BUSINESS
- Deposit Products
- Lending Services
- Cards
- Insurance
- Retirement
- Other Services

INSIDE ALTORO MUTU
- About Us
- Contact Us
- Locations
- Investor Relations
- Press Room
- Careers

Search Results

The page at http://www.testfire.net says:

⚠ ASP.NET_SessionId=trohgq450cpi5r45rr2pl1fg; amSessionId=1824418181

[OK]
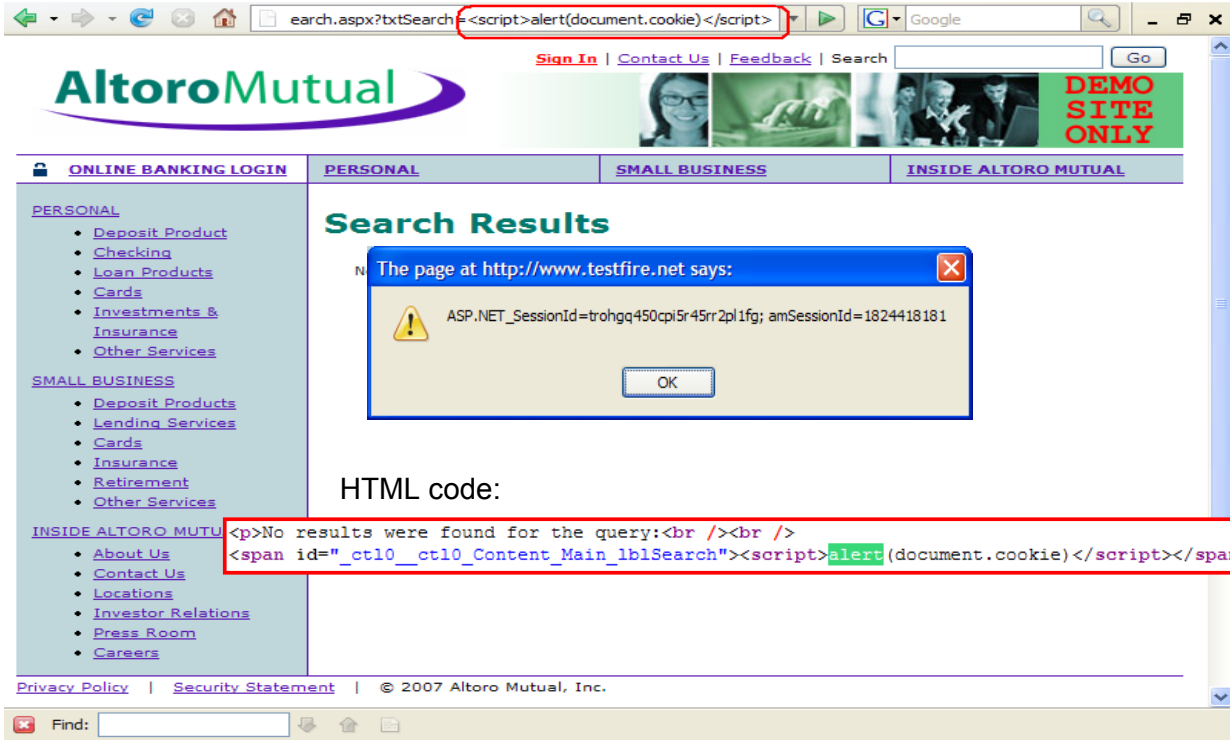
HTML code:

```
<p>No results were found for the query:<br /><br />
<span id="_ctl0__ctl0_Content_Main_lblSearch"><script>alert(document.cookie)</script></span>
```

Privacy Policy | Security Statement | © 2007 Altoro Mutual, Inc.

Find: [   ]

Cross-Site Scripting – The Exploit Process
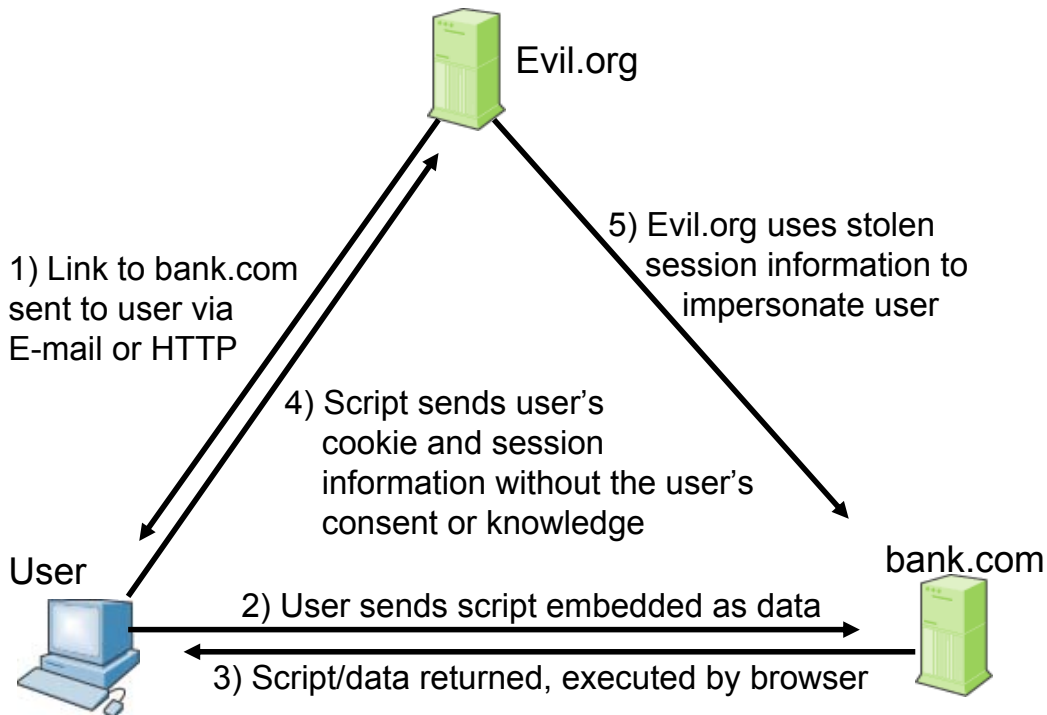
Evil.org

1) Link to bank.com sent to user via E-mail or HTTP

5) Evil.org uses stolen session information to impersonate user

4) Script sends user's cookie and session information without the user's consent or knowledge

User

2) User sends script embedded as data

3) Script/data returned, executed by browser

bank.com

Agenda

- Introductions & facilities
- Security Landscape
- Vulnerability Analysis
  - ▶ Top Attacks Overview
  - ▶ **Hands on Lab 1**
- Vulnerability Analysis (continued)
  - ▶ Hands on Lab 2
- Automated Vulnerability Analysis
  - ▶ AppScan Overview
  - ▶ Hands on Lab 3

Lab 1 Profile Web Application and XSS

- The Goal of this lab is to:
  - ▶ profile the demo.testfire.net application
  - ▶ utilize a Cross-Site Scripting vulnerability on the demo.testfire.net application in order to access cookies on a target user's browser

Agenda

- Introductions & facilities
- Security Landscape
- Vulnerability Analysis
  - ▸ Top Attacks Overview
  - ▸ Hands on Lab 1
- **Vulnerability Analysis (continued)**
  - ▸ Hands on Lab 2
- Automated Vulnerability Analysis
  - ▸ AppScan Overview
  - ▸ Hands on Lab 3

---

2 - Injection Flaws

- What is it?
  - ▸ User-supplied data is sent to an interpreter as part of a command, query or data.

- What are the implications?
  - ▸ SQL Injection – Access/modify data in DB
  - ▸ SSI Injection – Execute commands on server and access sensitive data
  - ▸ LDAP Injection – Bypass authentication

SQL Injection

- User input inserted into SQL Command:
  - Get product details by id:
    Select * from products where id='$REQUEST["id"]';
  - Hack: send param id with value ' or '1'='1
  - Resulting executed SQL:
    Select * from products where id='' or '1'='1'
  - All products returned

Demonstration – SQL Injection

- Main points covered in the demo or video:
  - How to find a SQL injection vulnerability
  - How to exploit a SQL injection vulnerability

SQL Injection Example I

SQL Injection Example II

SQL Injection Example - Exploit

SQL Injection Example - Outcome

Injection Flaws (SSI Injection Example)
Creating commands from input
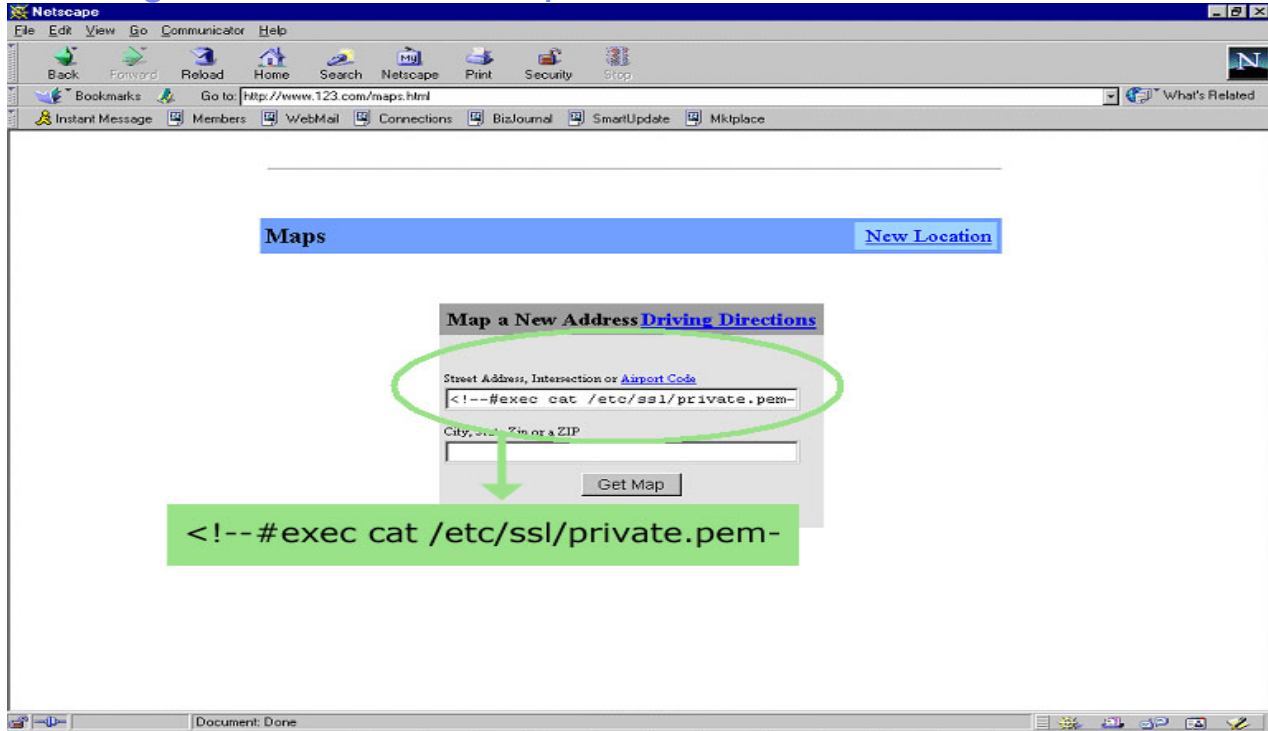
The return is the private SSL key of the server
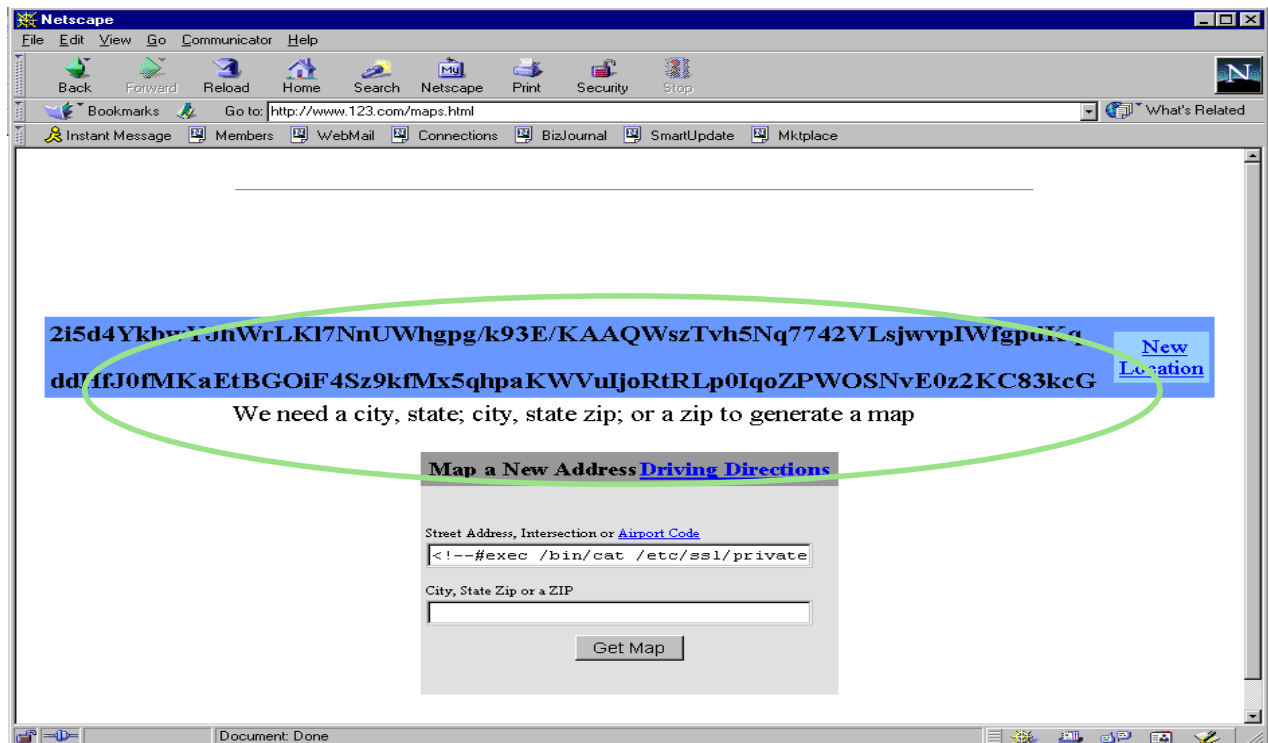
3 - Malicious File Execution

- **What is it?**
  - ▶ Application tricked into executing commands or creating files on server

- **What are the implications?**
  - ▶ Command execution on server – complete takeover
  - ▶ Site Defacement, including XSS option

Malicious File Execution – Example I

Malicious File Execution – Example cont.

Malicious File Execution – Example cont.

asdf, asdf, asdf, # Copyright (c) 1993-1999 Microsoft Corp. # # This is a sample HOSTS file used by Microsoft TCP/IP for Windows. # # This file contains the mappings of IP addresses to host names. Each # entry should be kept on an individual line. The IP address should # be placed in the first column followed by the corresponding host name. # The IP address and the host name should be separated by at least one # space. # # Additionally, comments (such as these) may be inserted on individual # lines or following the machine name denoted by a '#' symbol. # # For example: # # 102.54.94.97 rhino.acme.com # source server # 38.25.63.10 x.acme.com # x client host 127.0.0.1 localhost

4 - Insecure Direct Object Reference

- What is it?
  - ▸ Part or all of a resource (file, table, etc.) name controlled by user input.

- What are the implications?
  - ▸ Access to sensitive resources
  - ▸ Information Leakage, aids future hacks

Insecure Direct Object Reference - Example

Insecure Direct Object Reference – Example Cont.

http://www.testfire.net/default.aspx?content=../boot.ini

Error! File must be of type txt or htm

Insecure Direct Object Reference – Example Cont.

tp://www.testfire.net/default.aspx?content=../boot.ini%00.htm

[boot loader]timeout=30default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS[operating systems]multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP Professional" /noexecute=optin /fastdetect

5 - Information Leakage and Improper Error Handling

- **What is it?**
  - ▶ Unneeded information made available via errors or other means.

- **What are the implications?**
  - ▶ Sensitive data exposed
  - ▶ Web App internals and logic exposed (source code, SQL syntax, exception call stacks, etc.)
  - ▶ Information aids in further hacks

Information Leakage - Example


```
<h1>Online Banking Login</h1>

<!-- To get the latest admin login, please contact SiteOps at 415-555-6159 -->
<p><span id="_ctl0__ctl0_Content_Main_message"
```

Improper Error Handling - Example

Information Leakage – Different User/Pass Error

6 - Failure to Restrict URL Access

- What is it?
  - ▶ Resources that should only be available to authorized users can be accessed by forcefully browsing them

- What are the implications?
  - ▶ Sensitive information leaked/modified
  - ▶ Admin privileges made available to hacker

Failure to Restrict URL Access - Admin User login

Simple user logs in, forcefully browses to admin page

Failure to Restrict URL Access: Privilege Escalation Types

- Access given to completely restricted resources
  - ▸ Accessing files that shouldn't be served (*.bak, "Copy Of", *.inc, *.cs, ws_ftp.log, etc.)

- Vertical Privilege Escalation
  - ▸ Unknown user accessing pages past login page
  - ▸ Simple user accessing admin pages

- Horizontal Privilege Escalation
  - ▸ User accessing other user's pages
  - ▸ Example: Bank account user accessing another's

Agenda

- Introductions & facilities
- Security Landscape
- Vulnerability Analysis
  - ▶ Top Attacks Overview
  - ▶ Hands on Lab 1
- Vulnerability Analysis (continued)
  - ▶ **Hands on Lab 2**
- Automated Vulnerability Analysis
  - ▶ AppScan Overview
  - ▶ Hands on Lab 3

Lab 2

Lab 1 – Profile Web Application, Steal Cookies

Lab 2 – Login without Credentials, Steal Usernames and Passwords, Logging into the Administrative Portal
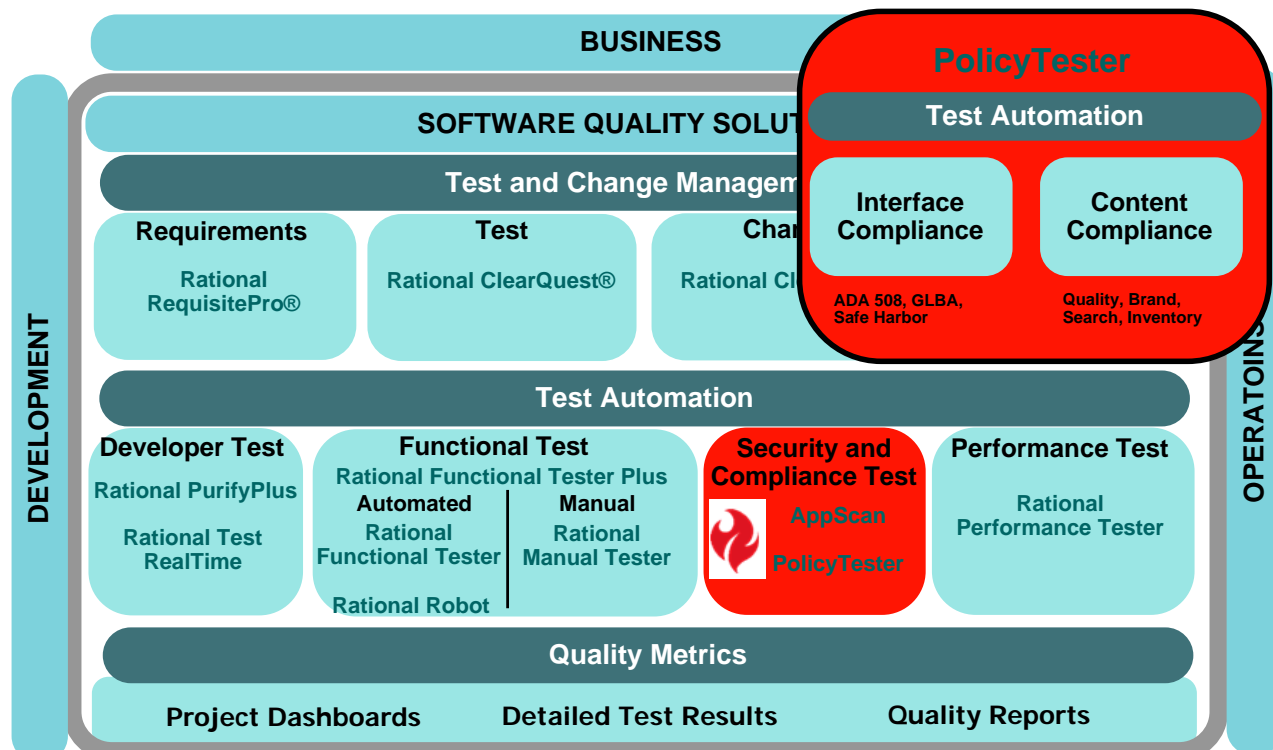
Lab 3 – Automated Scan of Website

Agenda

- Introductions & facilities
- Security Landscape
- Vulnerability Analysis
  - ▸ Top Attacks Overview
  - ▸ Hands on Lab 1
- Vulnerability Analysis (continued)
  - ▸ Hands on Lab 2
- **Automated Vulnerability Analysis**
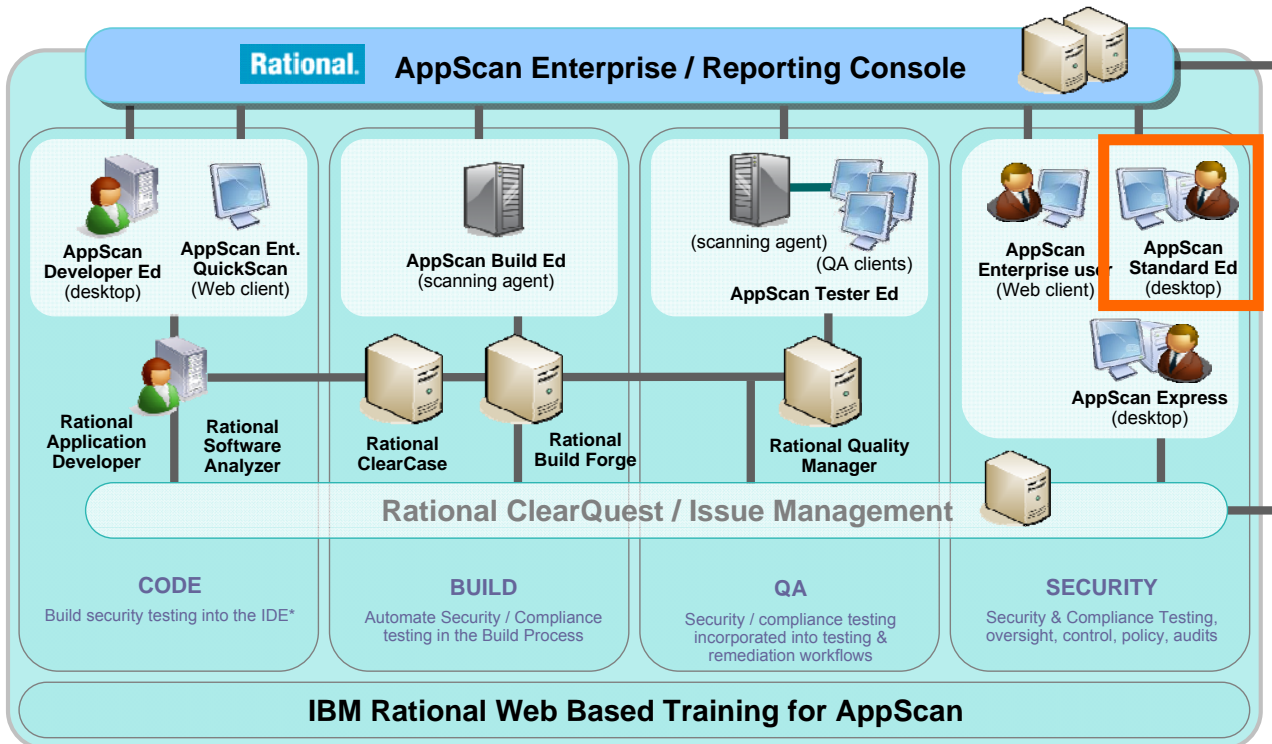 - ▸ AppScan Overview
  - ▸ Hands on Lab 3

Watchfire in the Rational Portfolio

BUSINESS

PolicyTester

SOFTWARE QUALITY SOLUT

Test Automation

Test and Change Managem

Requirements	Test	Char
Rational RequisitePro®	Rational ClearQuest®	Rational Cl

Interface Compliance

ADA 508, GLBA, Safe Harbor

Content Compliance

Quality, Brand, Search, Inventory

DEVELOPMENT

OPERATOINS

Test Automation

Developer Test

Rational PurifyPlus

Rational Test RealTime

Functional Test

Rational Functional Tester Plus

Automated	Manual
Rational Functional Tester	Rational Manual Tester

Rational Robot

Security and Compliance Test

AppScan

PolicyTester

Performance Test

Rational Performance Tester

Quality Metrics

Project Dashboards   **Detailed Test Results**   **Quality Reports**
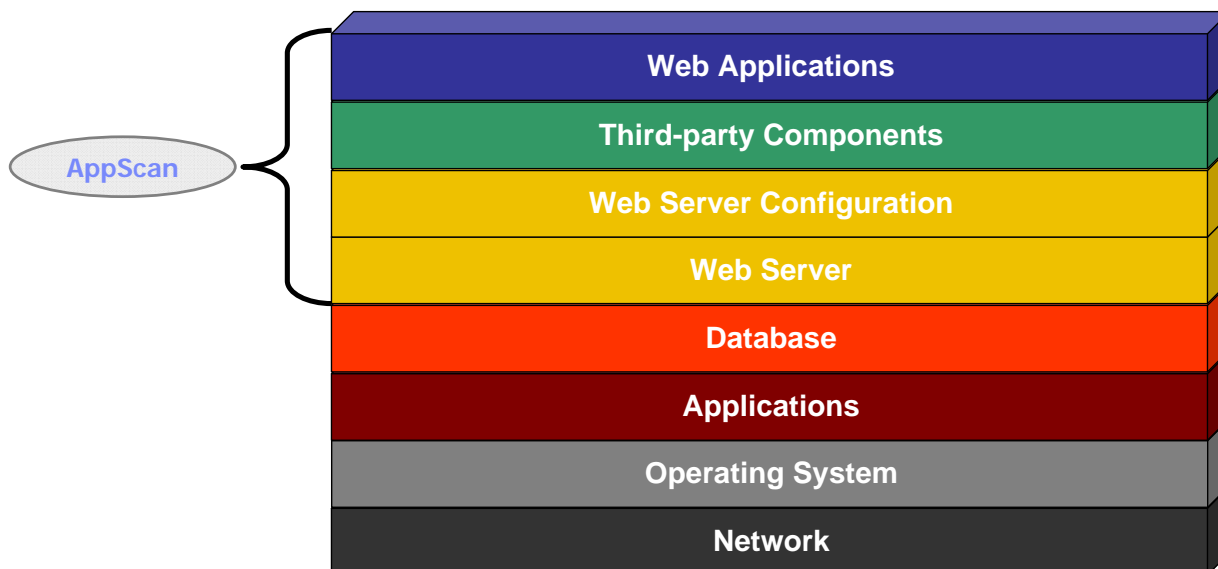

AppScan

- What is it?
  - ▸ AppScan is an automated tool used to perform vulnerability assessments on Web Applications

- Why do I need it?
  - ▸ To simplify finding and fixing Web application security problems

- What does it do?
  - ▸ Scans Web applications, finds security issues and reports on them in an actionable fashion

- Who uses it?
  - ▸ Security Auditors – main users today
  - ▸ QA engineers – when the auditors become the bottle neck
  - ▸ Developers – to find issues as early as possible (most efficient)

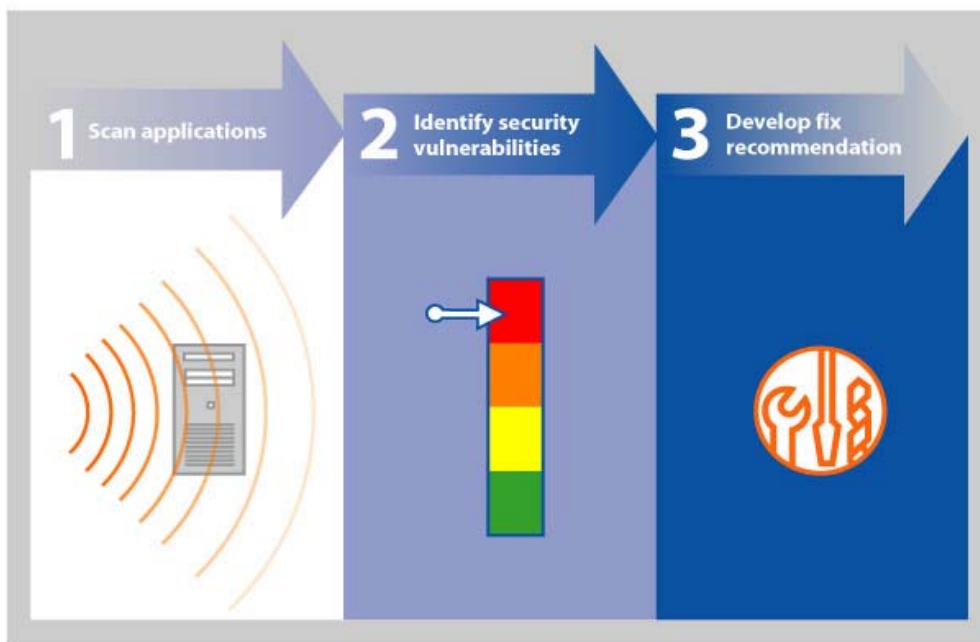# IBM Rational AppScan Ecosystem

Rational. **AppScan Enterprise / Reporting Console**

AppScan Developer Ed (desktop)   **AppScan Ent. QuickScan** (Web client)

AppScan Build Ed (scanning agent)

(scanning agent)   (QA clients)
AppScan Tester Ed

AppScan Enterprise user (Web client)   **AppScan Standard Ed** (desktop)

Rational Application Developer   **Rational Software Analyzer**   **Rational ClearCase**   **Rational Build Forge**   **Rational Quality Manager**   **AppScan Express** (desktop)

Rational ClearQuest / Issue Management

CODE
Build security testing into the IDE*

BUILD
Automate Security / Compliance testing in the Build Process

QA
Security / compliance testing incorporated into testing & remediation workflows

SECURITY
Security & Compliance Testing, oversight, control, policy, audits

IBM Rational Web Based Training for AppScan

What does AppScan test for?

AppScan

| Web Applications |
| Third-party Components |
| Web Server Configuration |
| Web Server |
| Database |
| Applications |
| Operating System |
| Network |

How does AppScan work?

- Approaches an application as a black-box
- Traverses a Web application and builds the site model
- Determines the attack vectors based on the selected Test policy
- Tests by sending modified HTTP requests to the application and examining the HTTP response according to validate rules
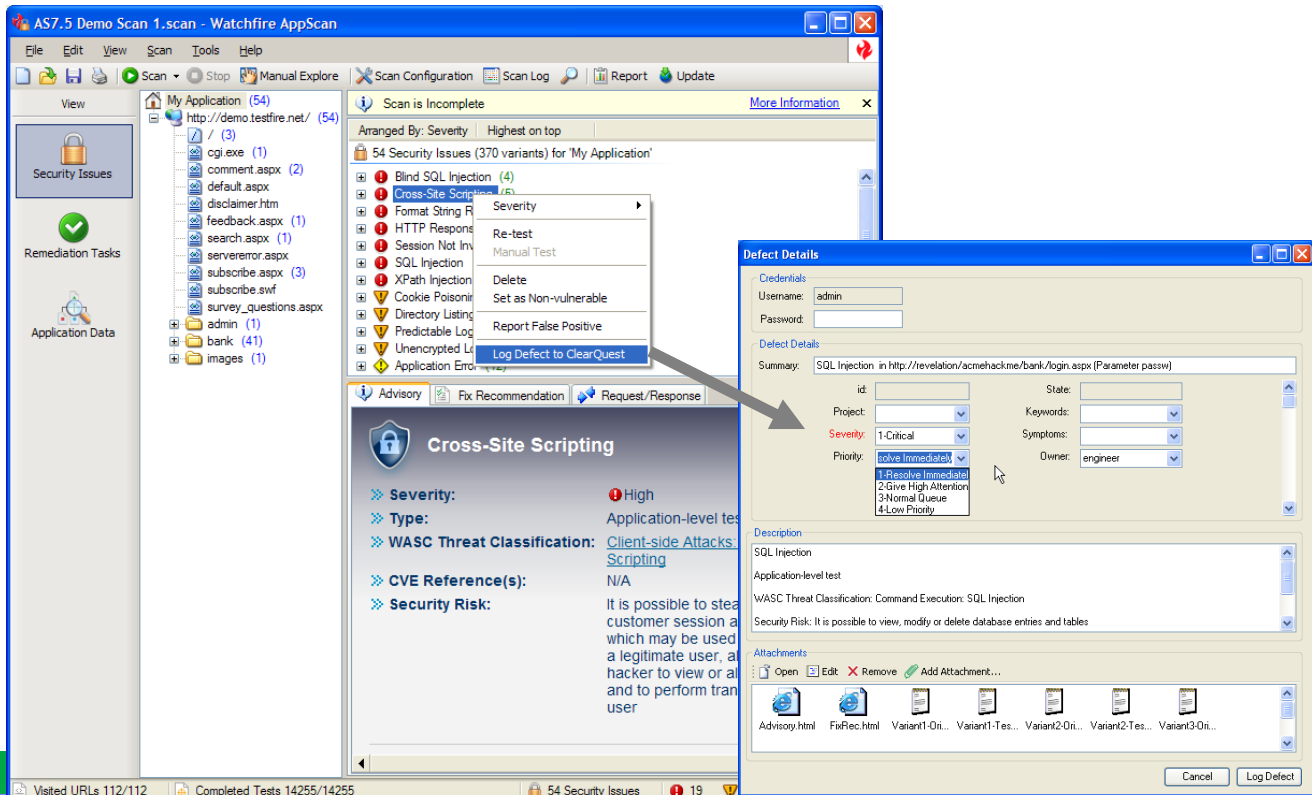
Web Application

HTTP Request

Application

Databases

Web Servers

HTTP Response
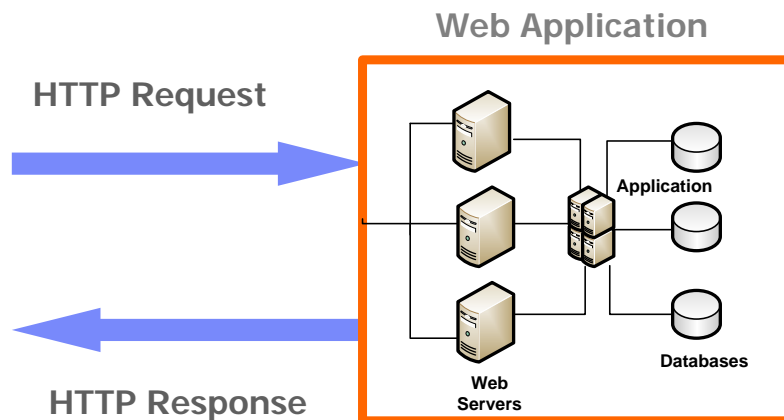
AppScan Goes Beyond Pointing out Problems

1 Scan applications

2 Identify security vulnerabilities

3 Develop fix recommendation

Actionable Fix Recommendations
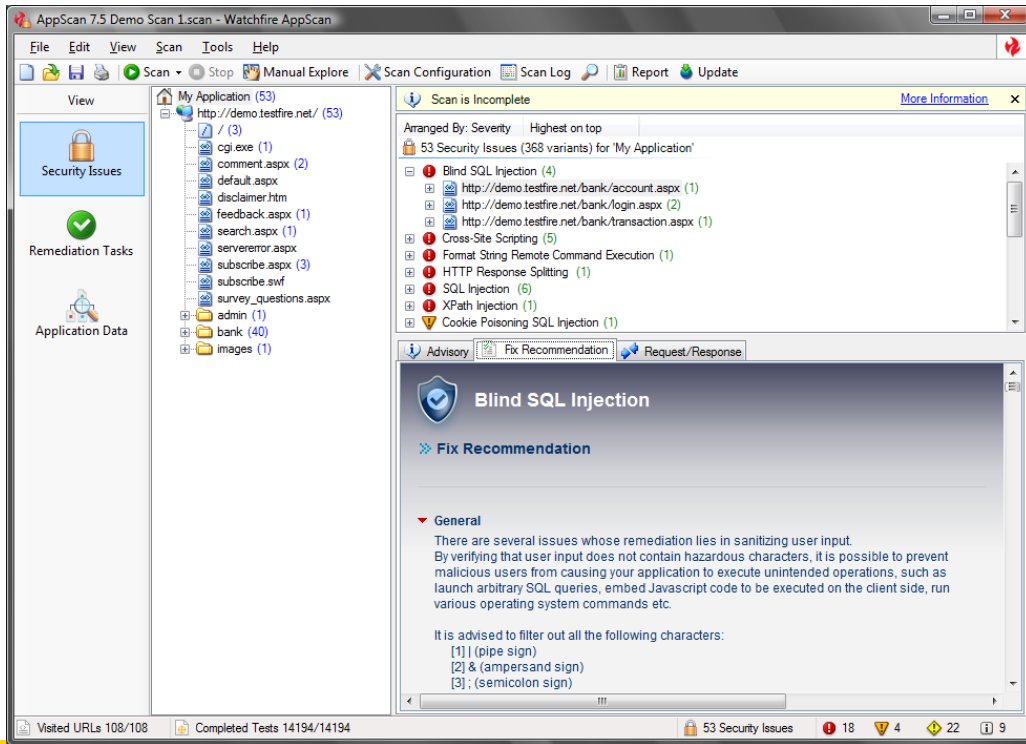
AppScan with QA Defect Logger for ClearQuest

Lab 3 overview

- The goal of this lab is to use AppScan in order to automate the detection of vulnerabilities within a Web application

IBM Software Group

Session summary

An IBM Proof of Technology

Session summary

- Understand the Web application environment
- Understand and differentiate between network and application level vulnerabilities
- Understand where the vulnerabilities exist
- Hands on exercises to understand types of vulnerabilities
- Hands on exercise to leverage automated scan for vulnerabilities

Questions

Next steps

- We can schedule a Vulnerability Assessment of one our your Applications -

Reference materials

- IBM.com
    - http://www-306.ibm.com/software/rational/welcome/watchfire/products.html

Thank You

We appreciate your feedback.
Please fill out the survey form in order
to improve this educational event.