IBM Software Group

Discovering the Value of Verifying Web Application Security

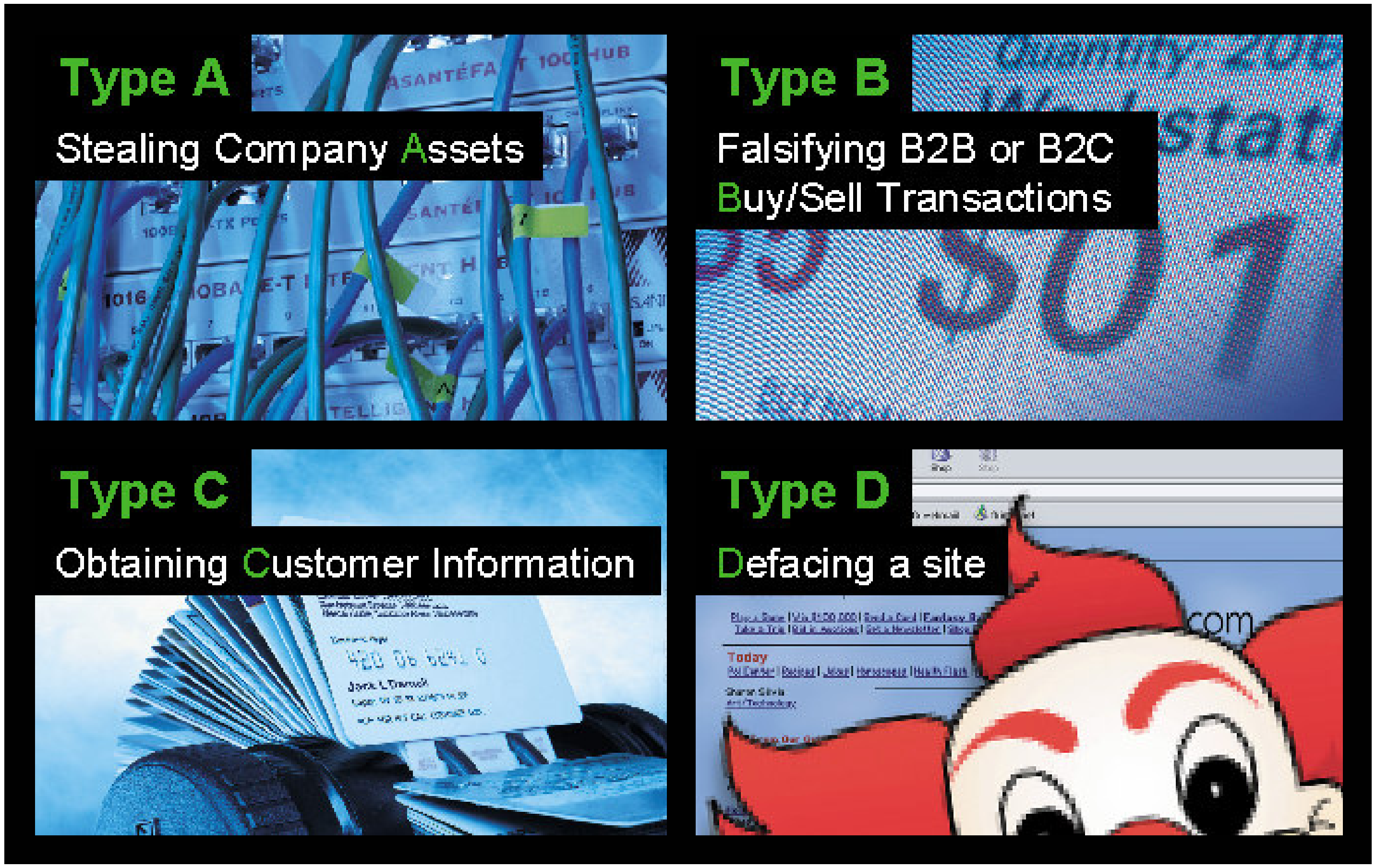
Martin Lee
IBM Rational Technical Consultant
yeekee@my.ibm.com

Agenda

- Security Landscape
- Top Hacker Attacks
- Automated Vulnerability Analysis

Web Attacks

The manipulation of web applications


Type A
Stealing Company Assets
Type B
Falsifying B2B or B2C Buy/Sell Transactions
Type C
Obtaining Customer Information
Type D
Defacing a site

The Myth: “Our Site Is Safe”

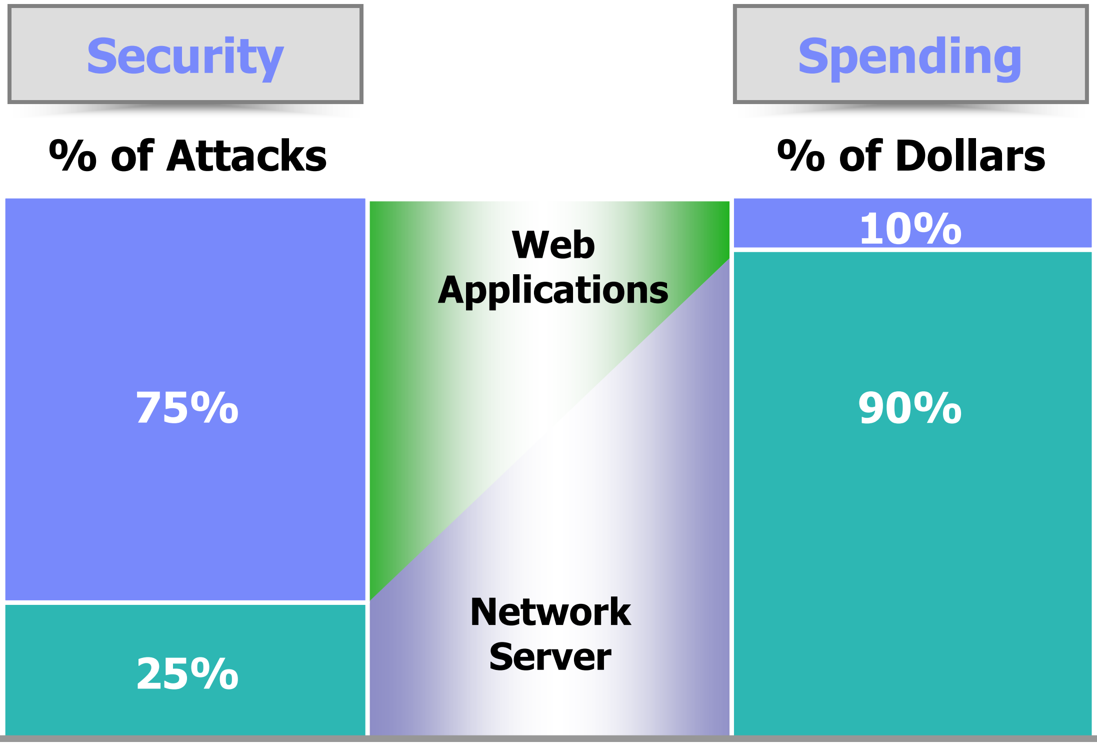
We Have Firewalls in Place

We Audit It Once a Quarter with Pen Testers

We Use Network Vulnerability Scanners

Security

The Reality: Security and Spending Are Unbalanced



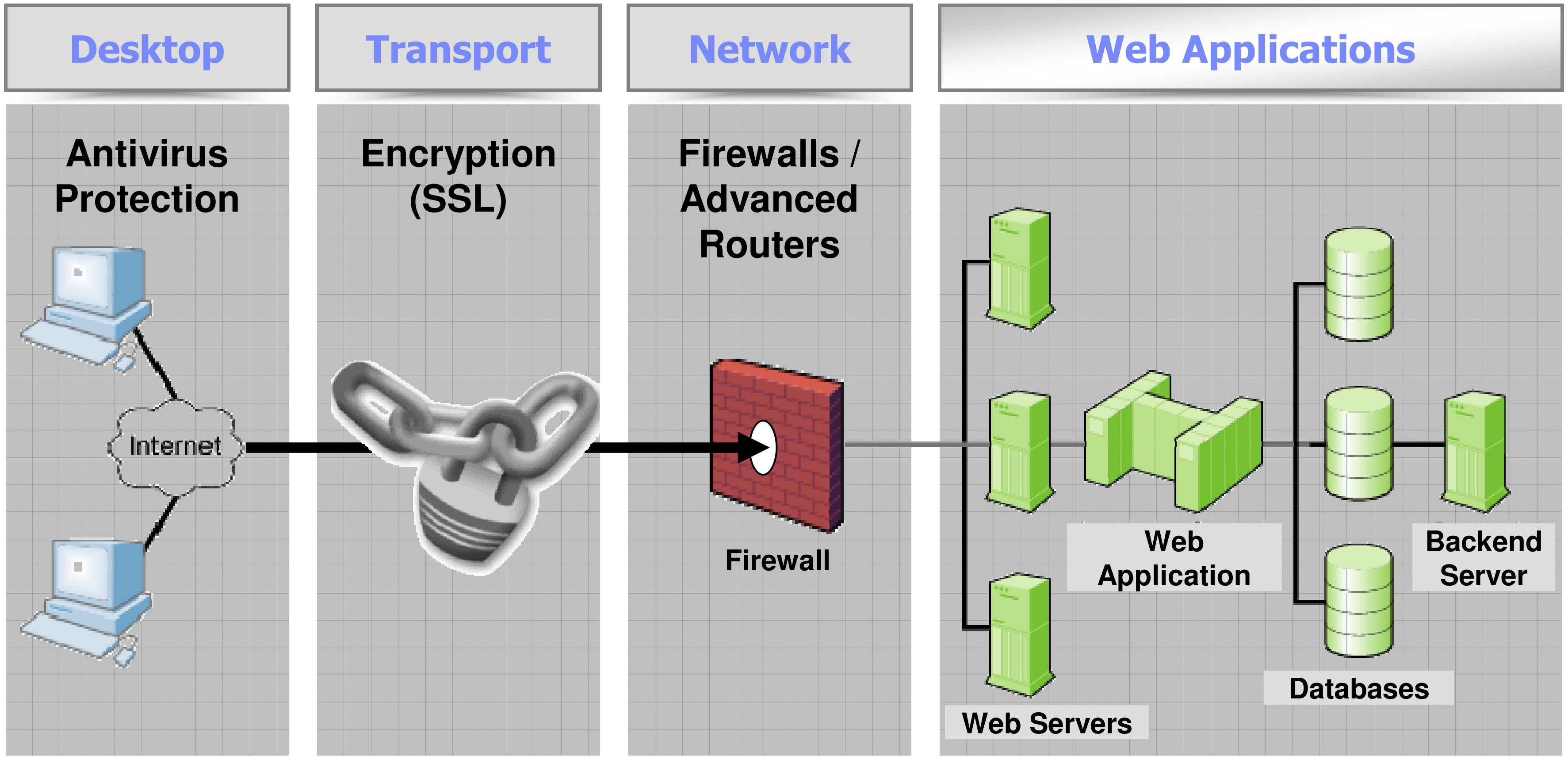
75% of All Attacks on Information Security Are Directed to the Web Application Layer

2/3 of All Web Applications Are Vulnerable

Gartner

Sources: Gartner, Watchfire

High Level Web Application Architecture



Why Application Security is a High Priority

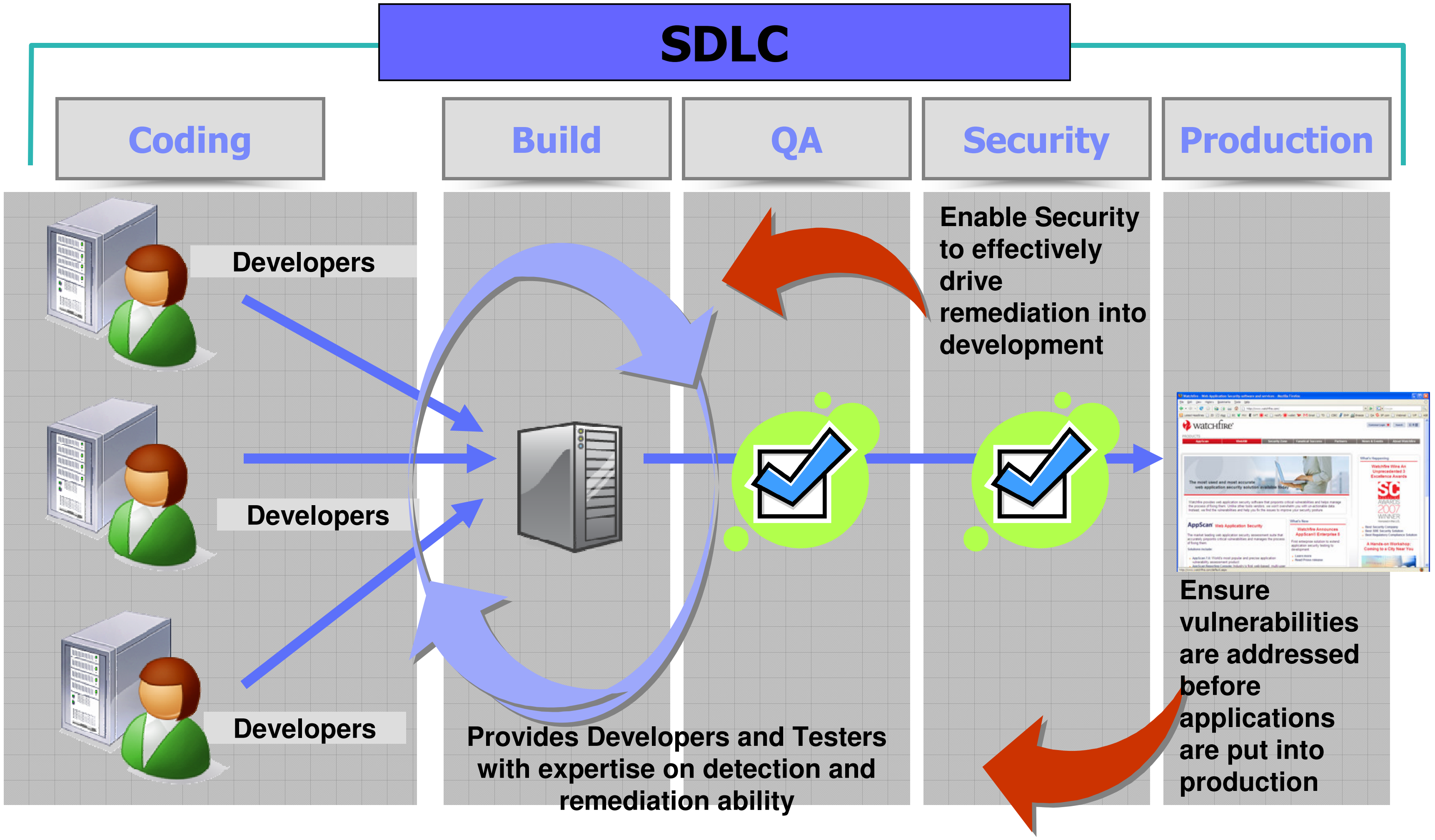
- **Web applications are the #1 focus of hackers:**
  - 75% of attacks at Application layer (Gartner)
  - XSS and SQL Injection are #1 and #2 reported vulnerabilities (Mitre)

- **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 SDLC


SDLC
Coding
Build
QA
Security
Production
Developers
Developers
Developers
Enable Security to effectively drive remediation into development
Provides Developers and Testers with expertise on detection and remediation ability
Ensure vulnerabilities are addressed before applications are put into production

Agenda

- Security Landscape
- Top Hacker Attacks
- Automated Vulnerability Analysis

WASC

- Web Application Security Consortium (WASC)

  Purpose:
  - To develop, adopt, and advocate standards for web application security
- Official web site: www.webappsec.org
- Web Security Threat Classification project

  http://www.webappsec.org/projects/threat/v1/WASC-TC-v1_0.pdf

  Purpose:
  - Clarify and organize the threats to the security of a web site
  - Develop and promote industry standard terminology for these issues

OWASP and the OWASP Top 10 list

- Open Web Application Security Project – 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

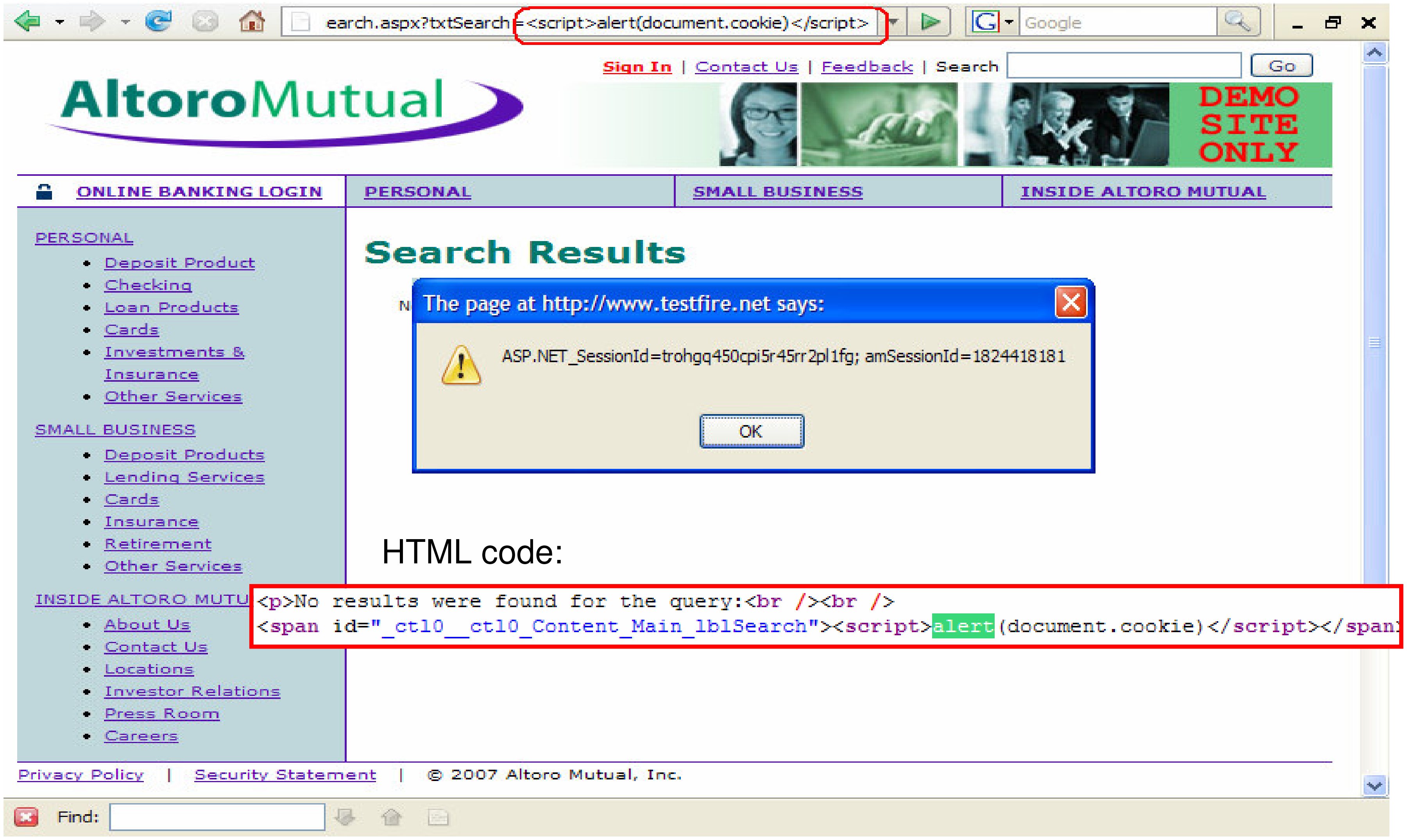
Application Threat	Negative Impact	Example Impact
Cross Site scripting	Identity Theft, Sensitive Information Leakage, …	Hackers can impersonate legitimate users, and control their accounts.
Injection Flaws	Attacker can manipulate queries to the DB / LDAP / Other system	Hackers can access backend database 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

XSS Example



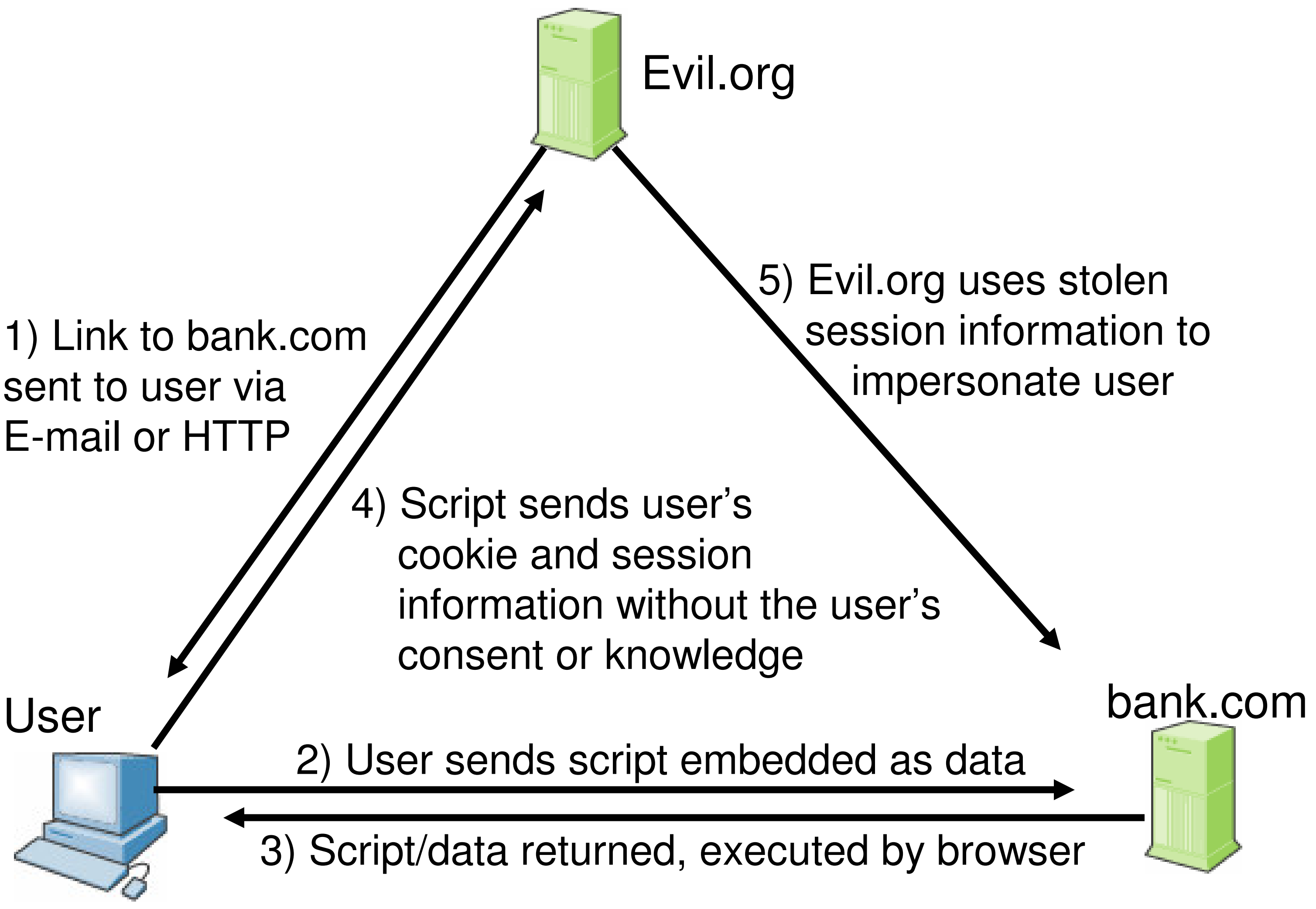
The page at http://www.testfire.net says:

```
ASP.NET_SessionId=trohgq450cpi5r45rr2pl1fg; amSessionId=1824418181
```

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
```

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
bank.com
2) User sends script embedded as data
3) Script/data returned, executed by browser

Exploiting XSS

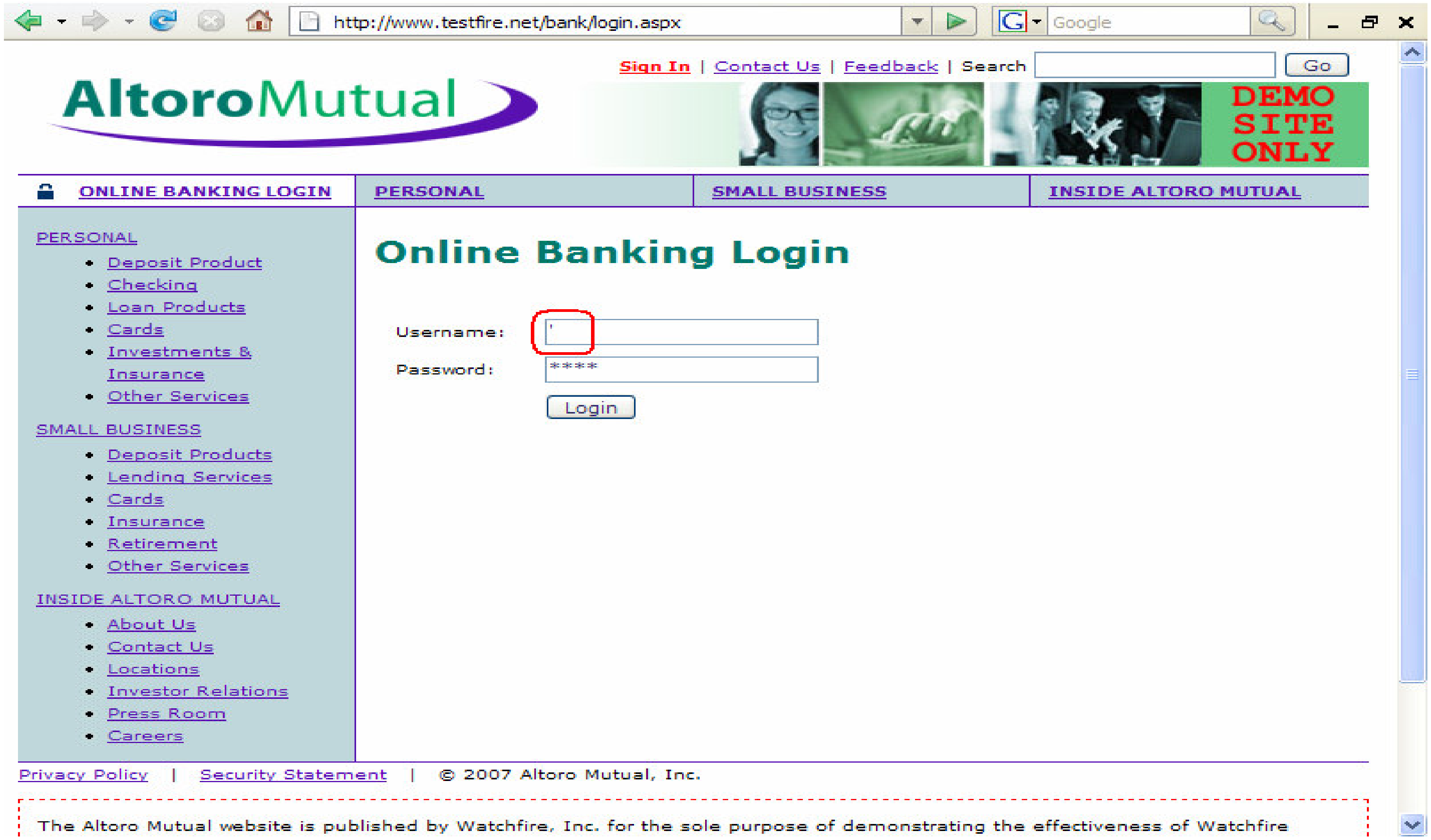
- If I can get you to run my JavaScript, I can…
  - Steal your cookies for the domain you're browsing
  - Track every action you do in that browser from now on
  - Redirect you to a Phishing site
  - Completely modify the content of any page you see on this domain
  - Exploit browser vulnerabilities to take over machine
  - …
- XSS is the Top Security Risk today (most exploited)

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 Example



http://www.testfire.net/bank/login.aspx

Sign In | Contact Us | Feedback | Search Go

AltoroMutual

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 MUTUAL
- About Us
- Contact Us
- Locations
- Investor Relations
- Press Room
- Careers

Online Banking Login

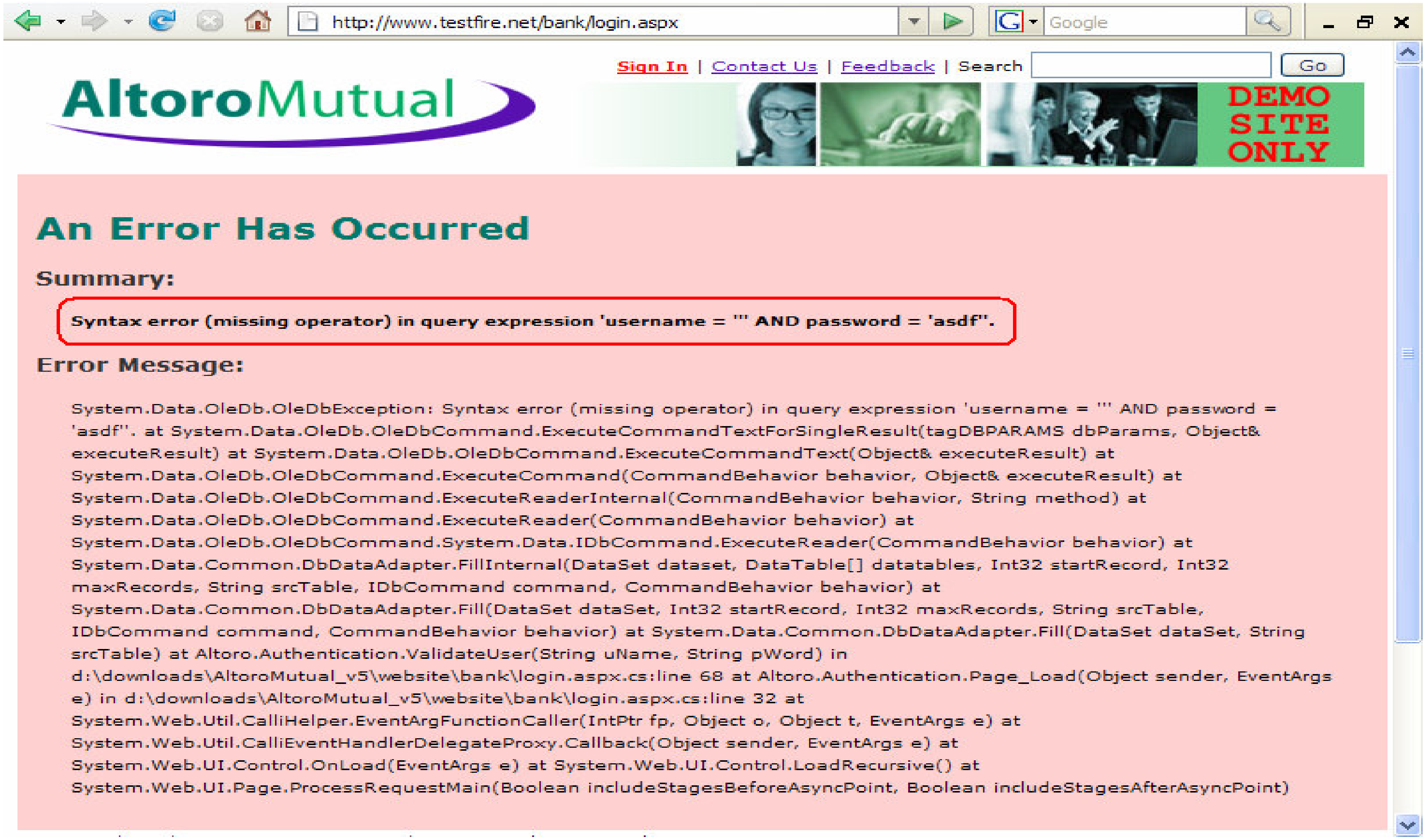
Username:	'
Password:	****

Login

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

The Altoro Mutual website is published by Watchfire, Inc. for the sole purpose of demonstrating the effectiveness of Watchfire

SQL Injection Example



http://www.testfire.net/bank/login.aspx
Google
Sign In | Contact Us | Feedback | Search
Go
AltoroMutual
DEMO SITE ONLY

An Error Has Occurred

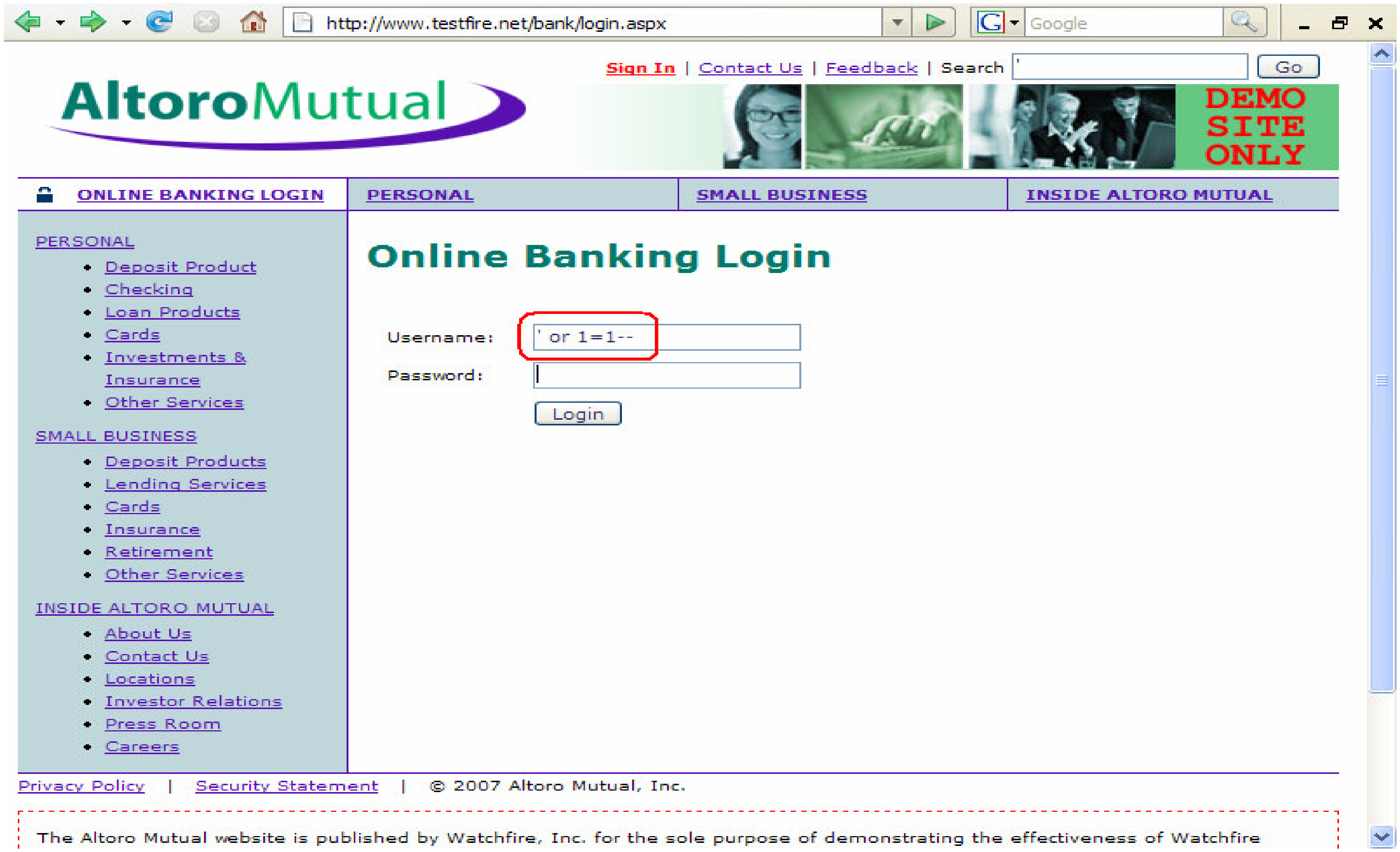
Summary:

Syntax error (missing operator) in query expression 'username = ''' AND password = 'asdf''.

Error Message:

```
System.Data.OleDb.OleDbException: Syntax error (missing operator) in query expression 'username = ''' AND password =
'asdf''. at System.Data.OleDb.OleDbCommand.ExecuteCommandTextForSingleResult(tagDBPARAMS dbParams, Object&
executeResult) at System.Data.OleDb.OleDbCommand.ExecuteCommandText(Object& executeResult) at
System.Data.OleDb.OleDbCommand.ExecuteCommand(CommandBehavior behavior, Object& executeResult) at
System.Data.OleDb.OleDbCommand.ExecuteReaderInternal(CommandBehavior behavior, String method) at
System.Data.OleDb.OleDbCommand.ExecuteReader(CommandBehavior behavior) at
System.Data.OleDb.OleDbCommand.System.Data.IDbCommand.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_v5\website\bank\login.aspx.cs:line 68 at Altoro.Authentication.Page_Load(Object sender, EventArgs
e) in d:\downloads\AltoroMutual_v5\website\bank\login.aspx.cs:line 32 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)
```

SQL Injection Example - Exploit



http://www.testfire.net/bank/login.aspx

Sign In | Contact Us | Feedback | Search '

AltoroMutual

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 MUTUAL
- About Us
- Contact Us
- Locations
- Investor Relations
- Press Room
- Careers

Online Banking Login

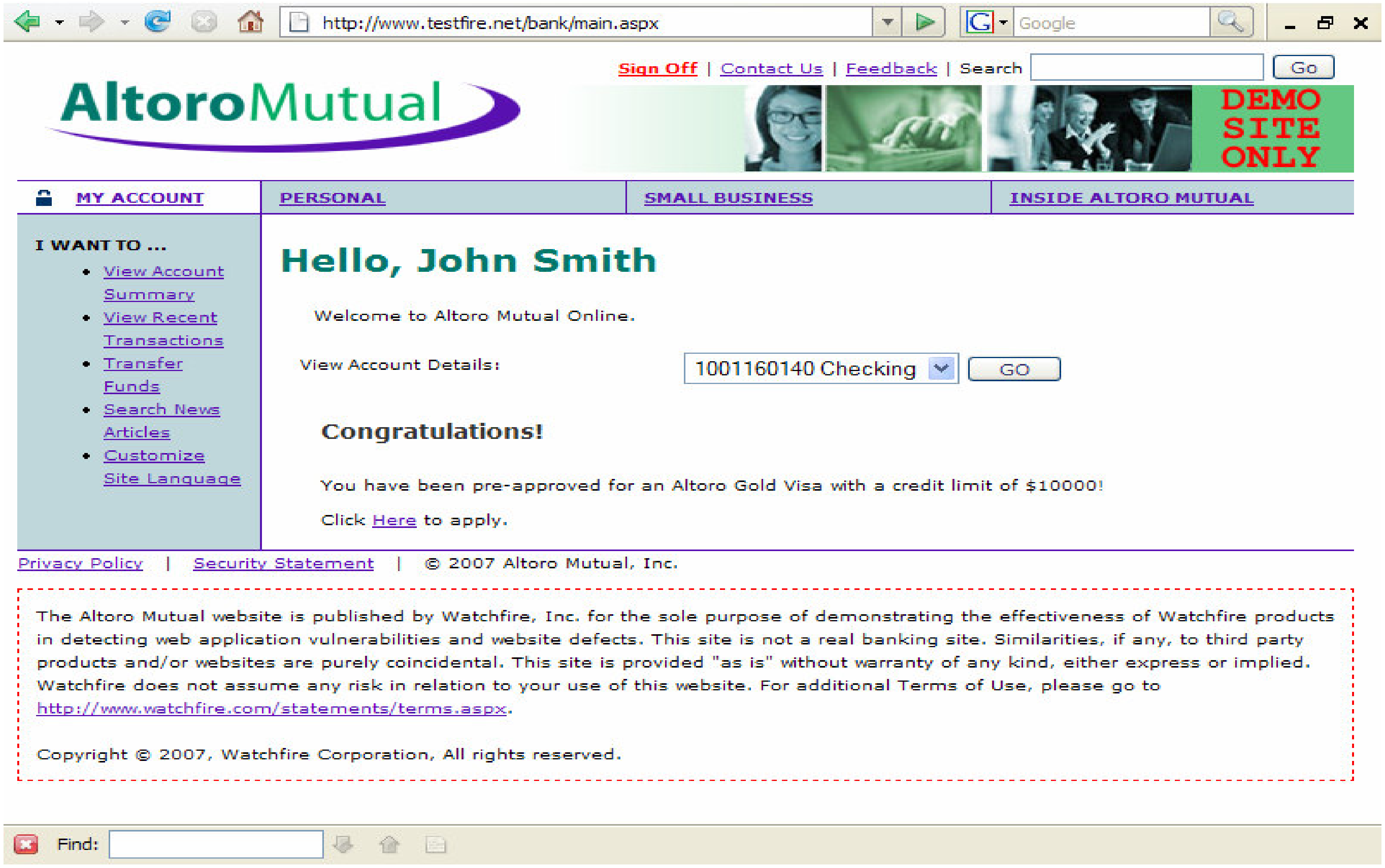
Username:	' or 1=1--
Password:	

Login

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

The Altoro Mutual website is published by Watchfire, Inc. for the sole purpose of demonstrating the effectiveness of Watchfire

SQL Injection Example - Outcome



Hello, John Smith

Welcome to Altoro Mutual Online.

View Account Details: 1001160140 Checking GO

Congratulations!

You have been pre-approved for an Altoro Gold Visa with a credit limit of $10000!

Click Here to apply.

Privacy Policy | Security Statement | © 2007 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.

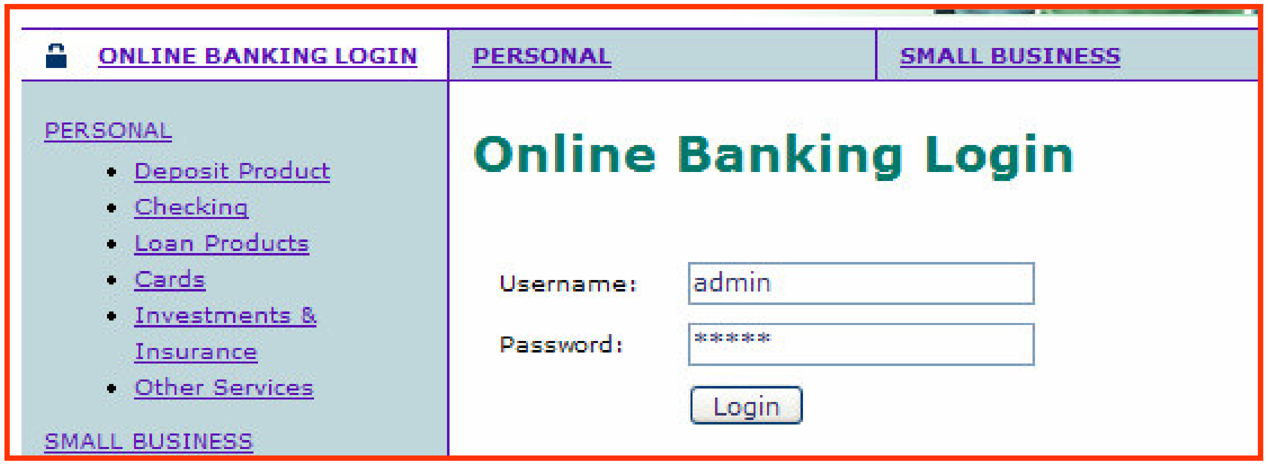
Copyright © 2007, Watchfire Corporation, All rights reserved.

3 - 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


ONLINE BANKING LOGIN | PERSONAL | SMALL BUSINESS

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

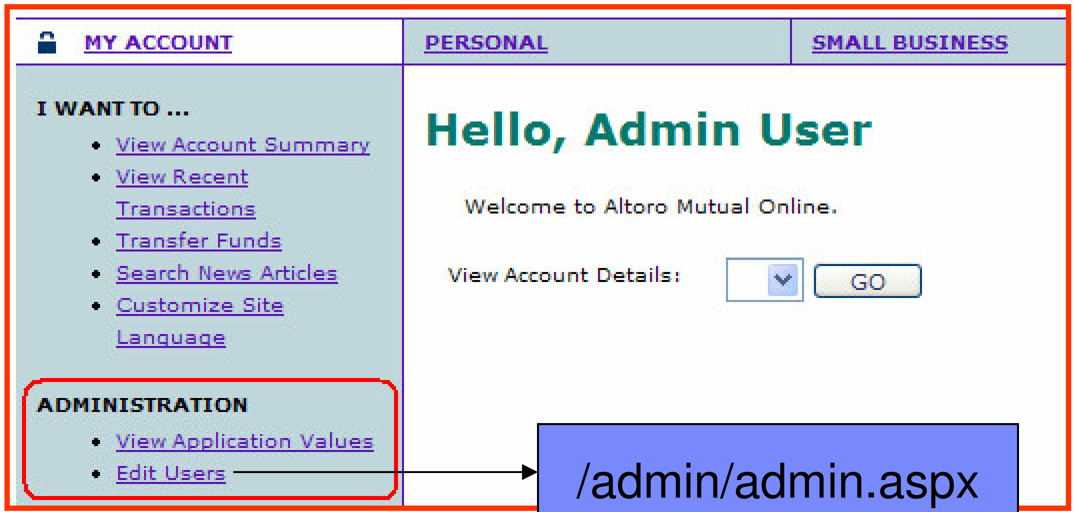
SMALL BUSINESS

Online Banking Login

Username: admin

Password: *****

Login


MY ACCOUNT | PERSONAL | SMALL BUSINESS

I WANT TO ...
- View Account Summary
- View Recent Transactions
- Transfer Funds
- Search News Articles
- Customize Site Language

ADMINISTRATION
- View Application Values
- Edit Users

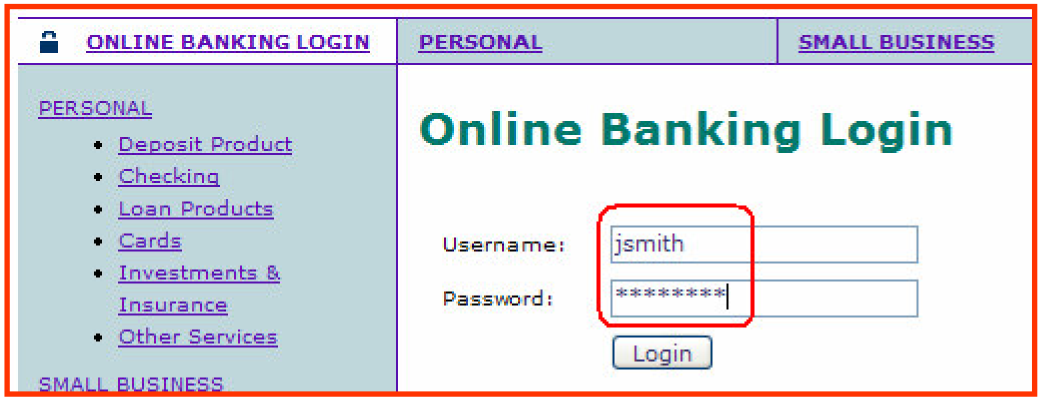
Hello, Admin User

Welcome to Altoro Mutual Online.

View Account Details: GO

/admin/admin.aspx

Simple user logs in, forcefully browses to admin page



ONLINE BANKING LOGIN | PERSONAL | SMALL BUSINESS

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

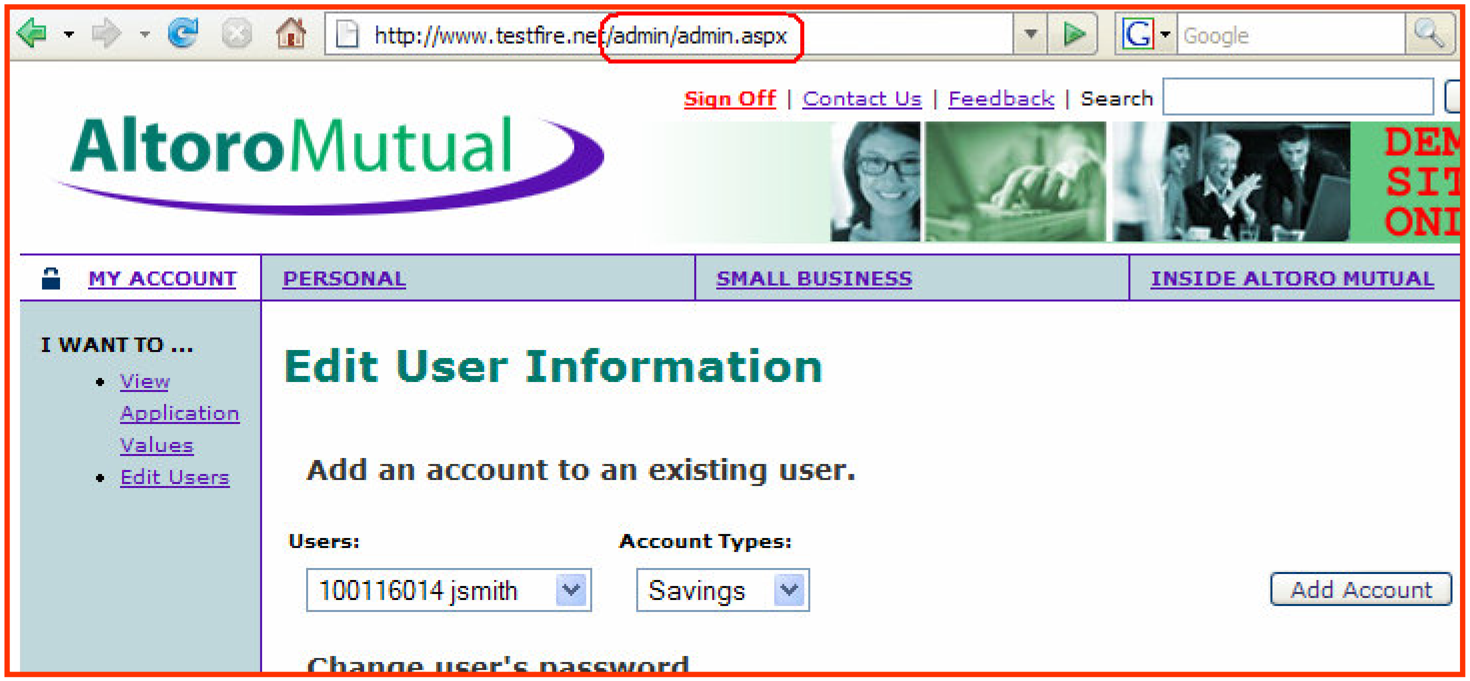
SMALL BUSINESS

Online Banking Login

Username: jsmith

Password: *********

Login



http://www.testfire.net/admin/admin.aspx

Sign Off | Contact Us | Feedback | Search

AltoroMutual

MY ACCOUNT | PERSONAL | SMALL BUSINESS | INSIDE ALTORO MUTUAL

I WANT TO ...
- View Application Values
- Edit Users

Edit User Information

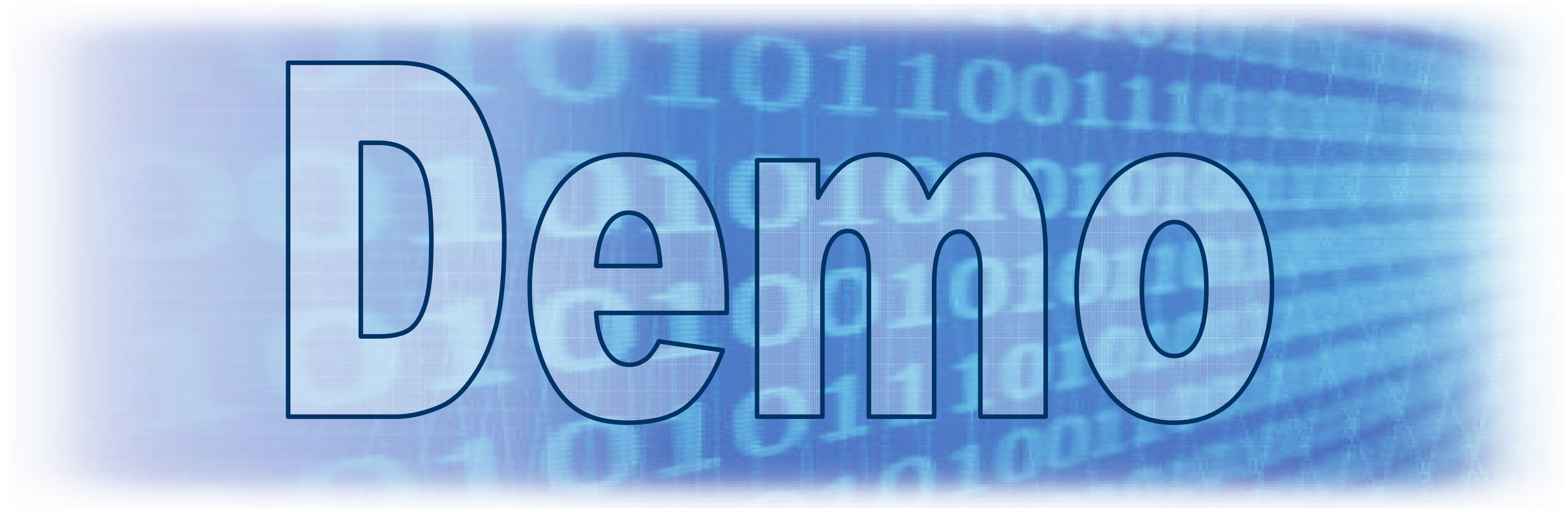
Add an account to an existing user.

Users: 100116014 jsmith

Account Types: Savings

Add Account

Change user's password



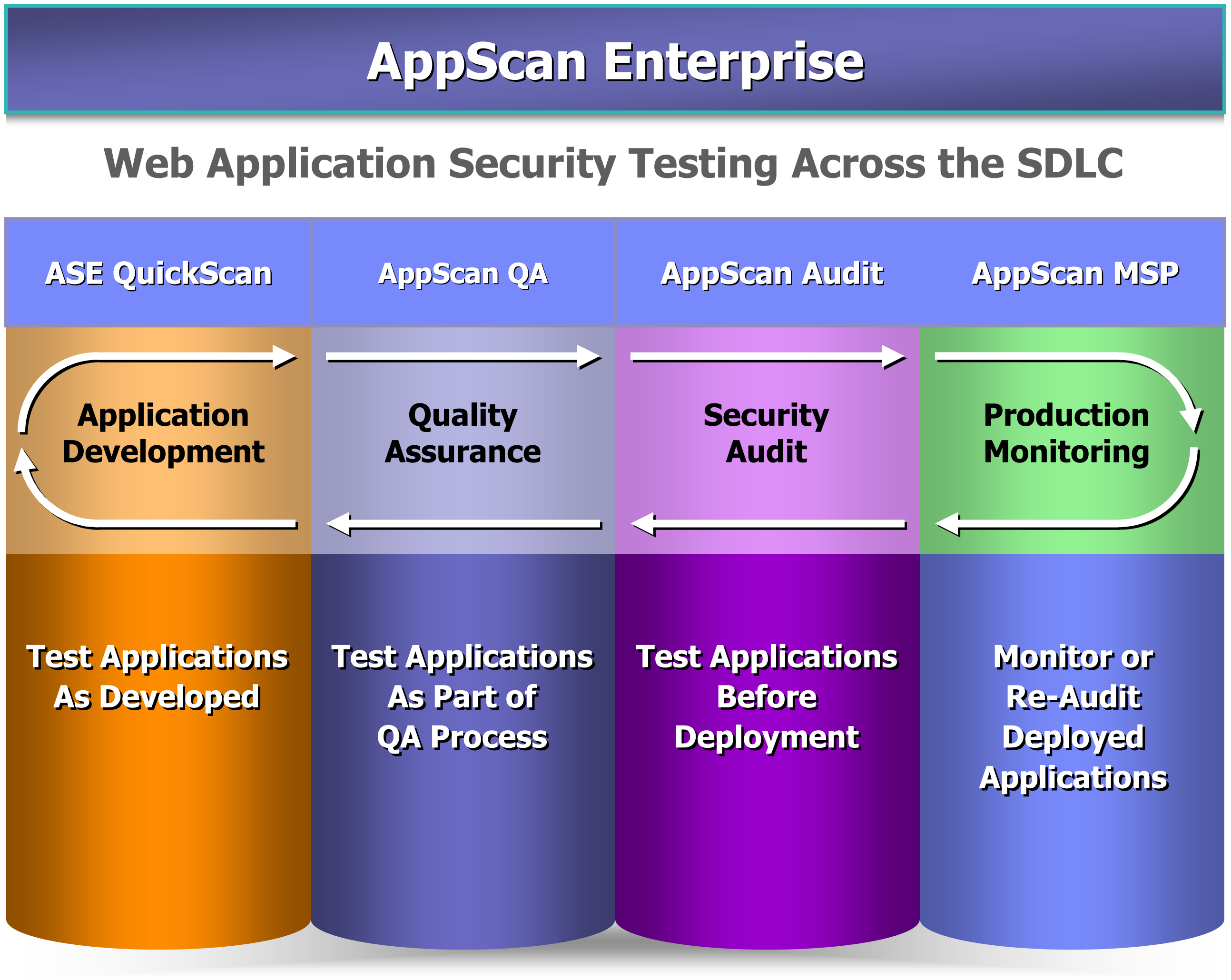
What You'll See:

- Cross Site Scripting
- SQL Injection
- Failure to Restrict URL Access

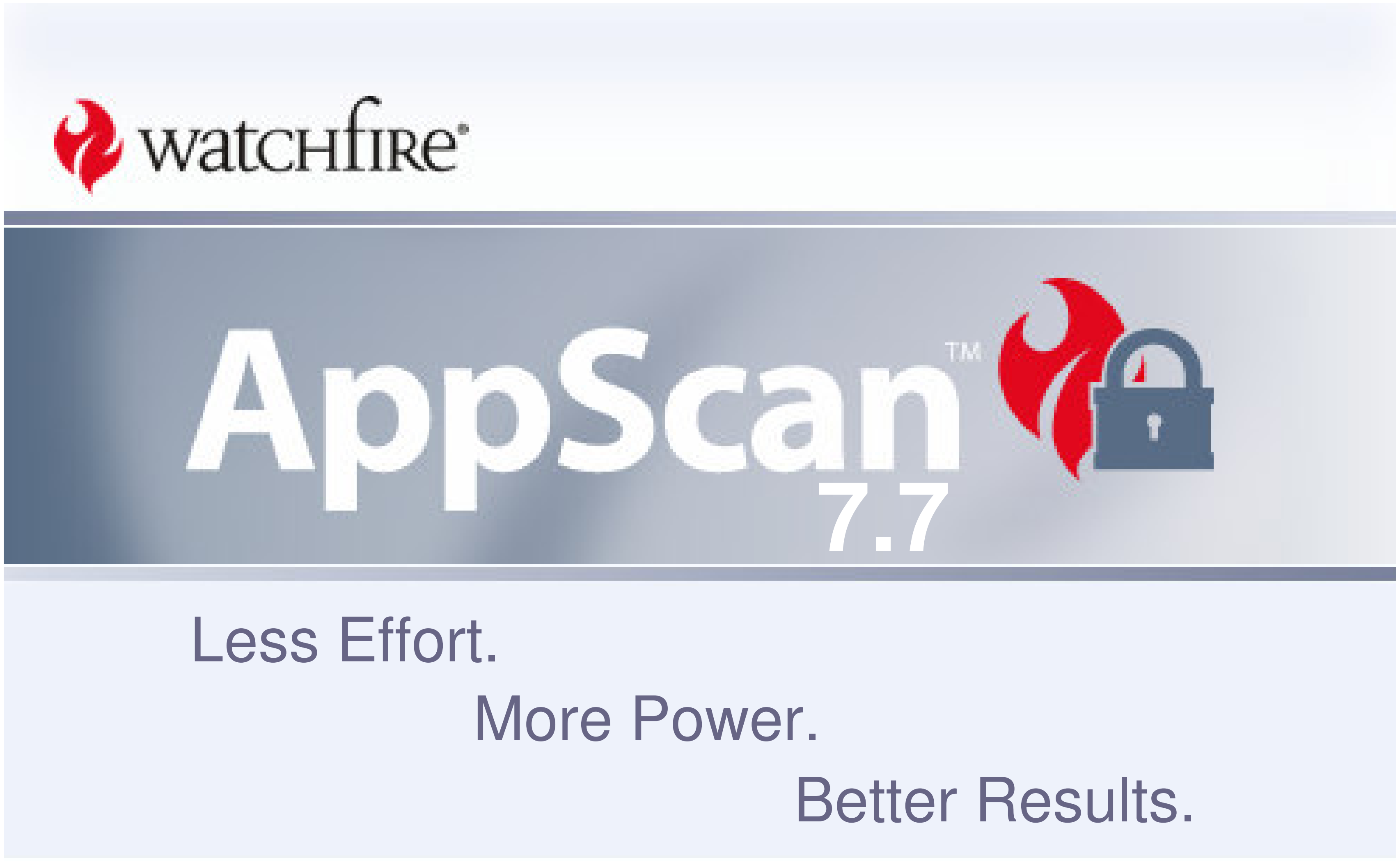
Agenda

- Security Landscape
- Top Hacker Attacks
- Automated Vulnerability Analysis

Watchfire Application Security Testing Products


AppScan Enterprise
Web Application Security Testing Across the SDLC
ASE QuickScan
AppScan QA
AppScan Audit
AppScan MSP
Application Development
Quality Assurance
Security Audit
Production Monitoring
Test Applications As Developed
Test Applications As Part of QA Process
Test Applications Before Deployment
Monitor or Re-Audit Deployed Applications

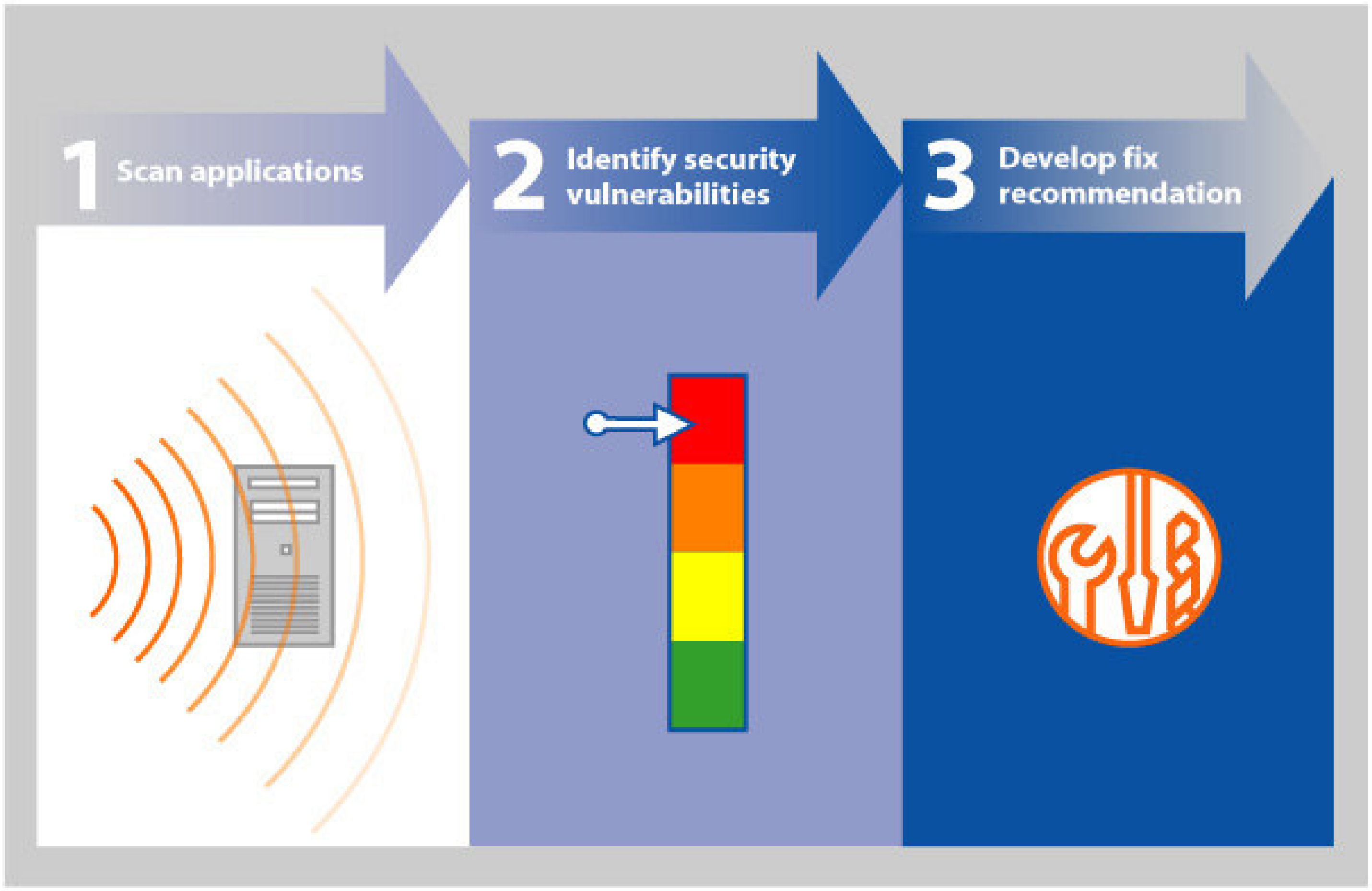
Introducing…



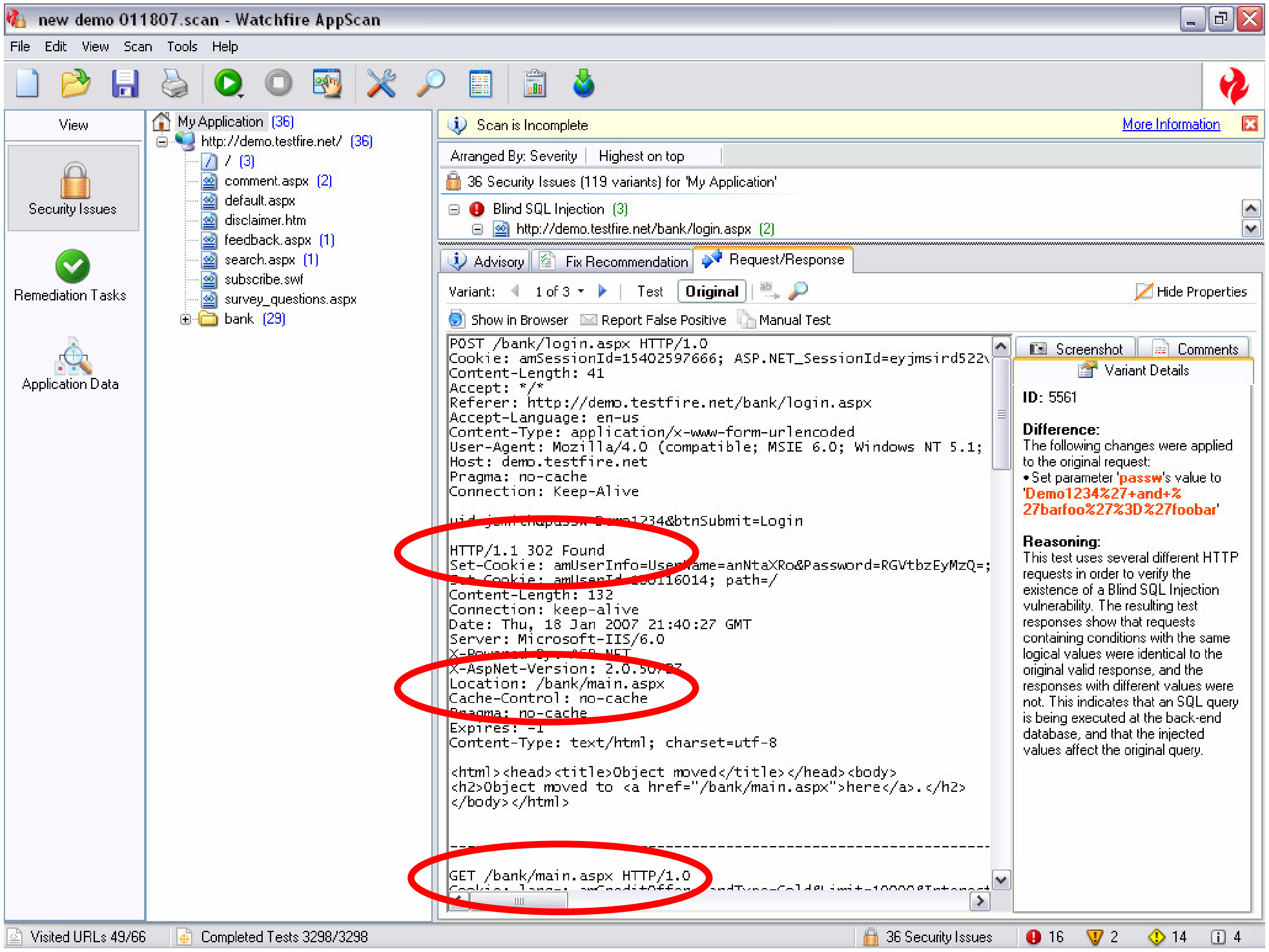
AppScan

- 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)

AppScan Goes Beyond Pointing out Problems


1 Scan applications
2 Identify security vulnerabilities
3 Develop fix recommendation

1. Review the original request



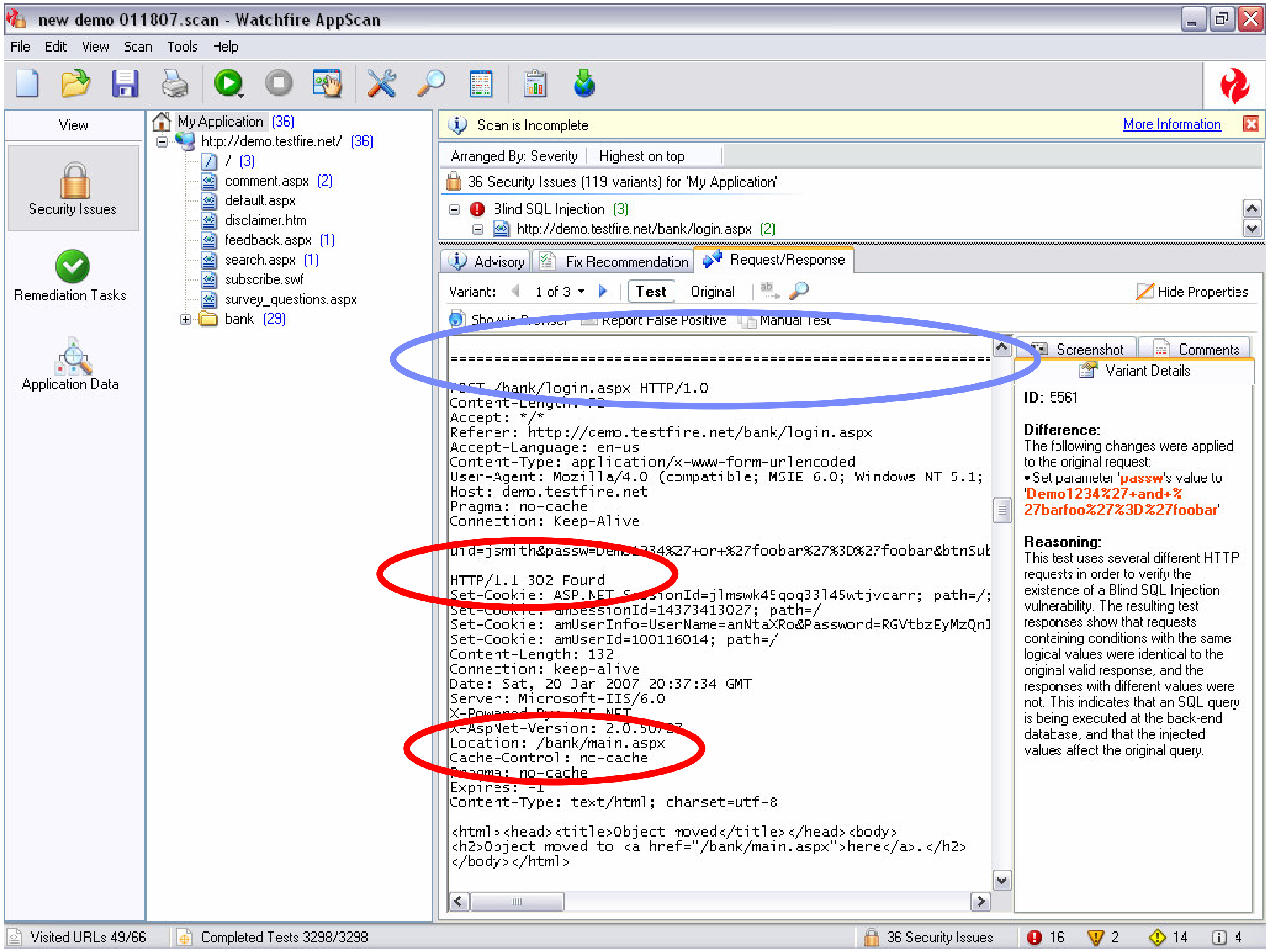
```
POST /bank/login.aspx HTTP/1.0
Cookie: amSessionId=15402597666; ASP.NET_SessionId=eyjmsird522v
Content-Length: 41
Accept: */*
Referer: http://demo.testfire.net/bank/login.aspx
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
Host: demo.testfire.net
Pragma: no-cache
Connection: Keep-Alive

uid=jsmith&passw=Demo1234&btnSubmit=Login

HTTP/1.1 302 Found
Set-Cookie: amUserInfo=UserName=anNtaXRo&Password=RGVtbzEyMzQ=;
Set-Cookie: amUserId=100116014; path=/
Content-Length: 132
Connection: keep-alive
Date: Thu, 18 Jan 2007 21:40:27 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Location: /bank/main.aspx
Cache-Control: no-cache
Pragma: no-cache
Expires: -1
Content-Type: text/html; charset=utf-8

<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="/bank/main.aspx">here</a>.</h2>
</body></html>


--------------------------------------------------------------------

GET /bank/main.aspx HTTP/1.0
Cookie: lang=; amCreditOffer=CardType=Gold&Limit=10000&Interest
```

ID: 5561

Difference:
The following changes were applied to the original request:
- Set parameter 'passw''s value to 'Demo1234%27+and+%27barfoo%27%3D%27foobar'

Reasoning:
This test uses several different HTTP requests in order to verify the existence of a Blind SQL Injection vulnerability. The resulting test responses show that requests containing conditions with the same logical values were identical to the original valid response, and the responses with different values were not. This indicates that an SQL query is being executed at the back-end database, and that the injected values affect the original query.

2. Review the result of the test



new demo 011807.scan - Watchfire AppScan

File Edit View Scan Tools Help

- My Application (36)
  - http://demo.testfire.net/ (36)
    - / (3)
    - comment.aspx (2)
    - default.aspx
    - disclaimer.htm
    - feedback.aspx (1)
    - search.aspx (1)
    - subscribe.swf
    - survey_questions.aspx
    - bank (29)

Scan is Incomplete — More Information

Arranged By: Severity | Highest on top

36 Security Issues (119 variants) for 'My Application'

- Blind SQL Injection (3)
  - http://demo.testfire.net/bank/login.aspx (2)

```
=================================================================

POST /bank/login.aspx HTTP/1.0
Content-Length: [illegible]
Accept: */*
Referer: http://demo.testfire.net/bank/login.aspx
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
Host: demo.testfire.net
Pragma: no-cache
Connection: Keep-Alive

uid=jsmith&passw=Demo1234%27+or+%27foobar%27%3D%27foobar&btnSub

HTTP/1.1 302 Found
Set-Cookie: ASP.NET_SessionId=jlmswk45qoq33145wtjvcarr; path=/;
Set-Cookie: amSessionId=14373413027; path=/
Set-Cookie: amUserInfo=UserName=anNtaXRo&Password=RGVtbzEyMzQnI
Set-Cookie: amUserId=100116014; path=/
Content-Length: 132
Connection: keep-alive
Date: Sat, 20 Jan 2007 20:37:34 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Location: /bank/main.aspx
Cache-Control: no-cache
Pragma: no-cache
Expires: -1
Content-Type: text/html; charset=utf-8

<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="/bank/main.aspx">here</a>.</h2>
</body></html>
```

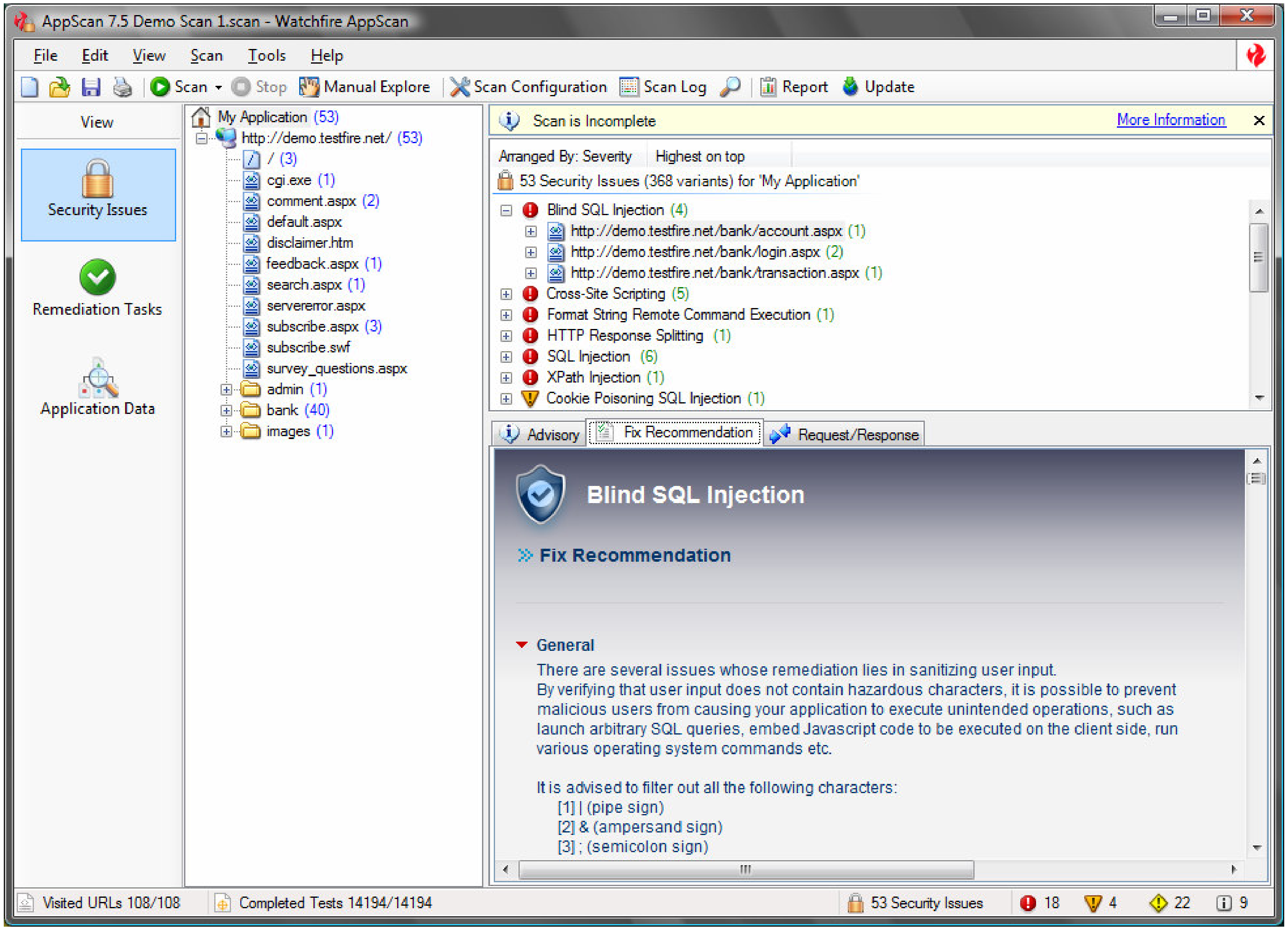
ID: 5561

Difference:
The following changes were applied to the original request:
- Set parameter 'passw''s value to 'Demo1234%27+and+%27barfoo%27%3D%27foobar'

Reasoning:
This test uses several different HTTP requests in order to verify the existence of a Blind SQL Injection vulnerability. The resulting test responses show that requests containing conditions with the same logical values were identical to the original valid response, and the responses with different values were not. This indicates that an SQL query is being executed at the back-end database, and that the injected values affect the original query.

Visited URLs 49/66 | Completed Tests 3298/3298 | 36 Security Issues | 16 | 2 | 14 | 4

3. Review Actionable Fix Recommendations


AppScan 7.5 Demo Scan 1.scan - Watchfire AppScan
File Edit View Scan Tools Help
Scan Stop Manual Explore Scan Configuration Scan Log Report Update
View
Security Issues
Remediation Tasks
Application Data
My Application (53)
http://demo.testfire.net/ (53)
/ (3)
cgi.exe (1)
comment.aspx (2)
default.aspx
disclaimer.htm
feedback.aspx (1)
search.aspx (1)
servererror.aspx
subscribe.aspx (3)
subscribe.swf
survey_questions.aspx
admin (1)
bank (40)
images (1)
Scan is Incomplete
More Information
Arranged By: Severity
Highest on top
53 Security Issues (368 variants) for 'My Application'
Blind SQL Injection (4)
http://demo.testfire.net/bank/account.aspx (1)
http://demo.testfire.net/bank/login.aspx (2)
http://demo.testfire.net/bank/transaction.aspx (1)
Cross-Site Scripting (5)
Format String Remote Command Execution (1)
HTTP Response Splitting (1)
SQL Injection (6)
XPath Injection (1)
Cookie Poisoning SQL Injection (1)
Advisory
Fix Recommendation
Request/Response
Visited URLs 108/108
Completed Tests 14194/14194
53 Security Issues
18
4
22
9

Blind SQL Injection

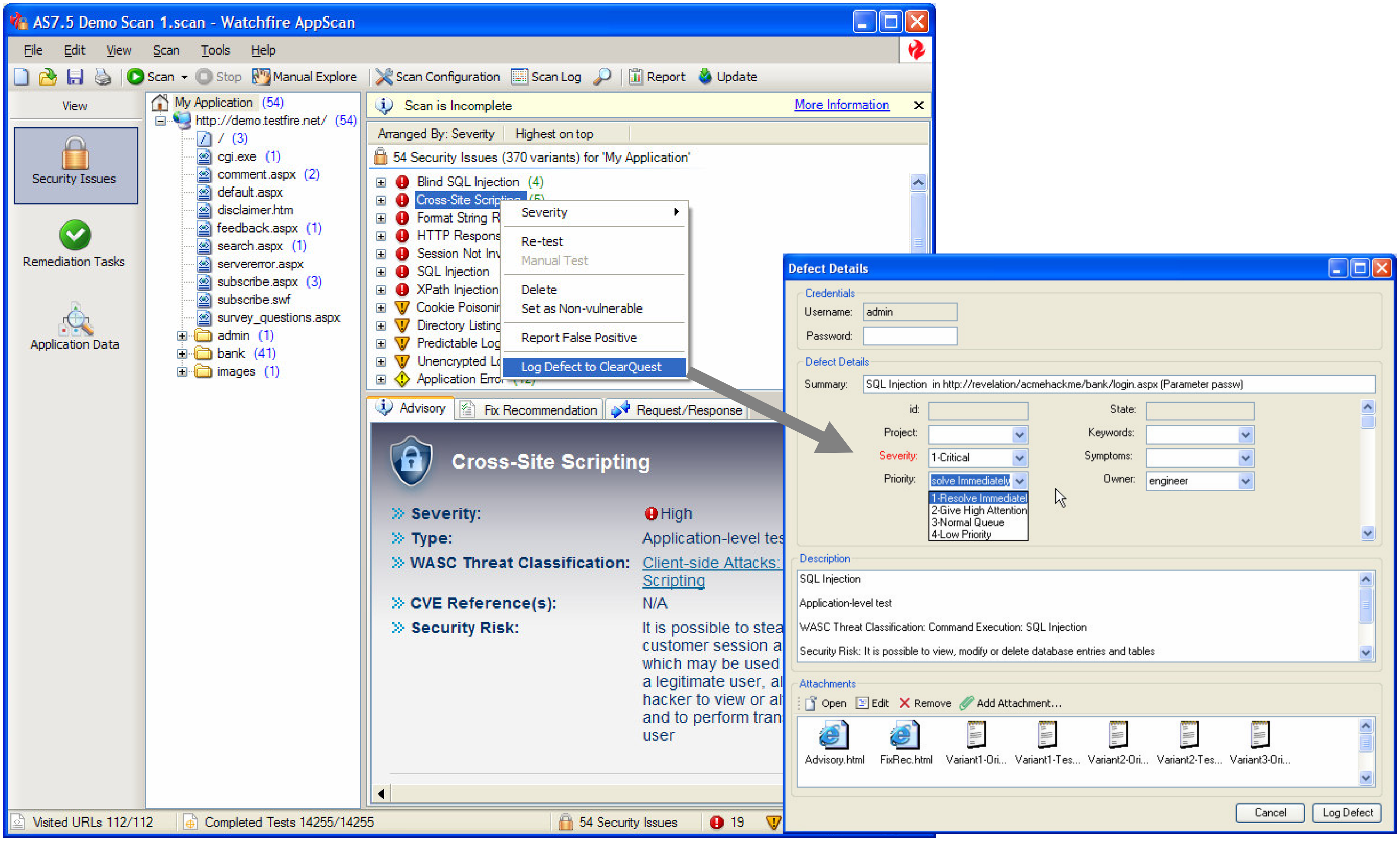
Fix Recommendation

General

There are several issues whose remediation lies in sanitizing user input.
By verifying that user input does not contain hazardous characters, it is possible to prevent malicious users from causing your application to execute unintended operations, such as launch arbitrary SQL queries, embed Javascript code to be executed on the client side, run various operating system commands etc.

It is advised to filter out all the following characters:
[1] | (pipe sign)
[2] & (ampersand sign)
[3] ; (semicolon sign)

4. Submit Defect to Development Team



AS7.5 Demo Scan 1.scan - Watchfire AppScan

File Edit View Scan Tools Help

Scan is Incomplete

Arranged By: Severity Highest on top

54 Security Issues (370 variants) for 'My Application'

- Blind SQL Injection (4)
- Cross-Site Scripting (5)
- Format String R
- HTTP Respons
- Session Not Inv
- SQL Injection
- XPath Injection
- Cookie Poisonin
- Directory Listing
- Predictable Log
- Unencrypted Lo
- Application Erro

My Application (54)
http://demo.testfire.net/ (54)
- / (3)
- cgi.exe (1)
- comment.aspx (2)
- default.aspx
- disclaimer.htm
- feedback.aspx (1)
- search.aspx (1)
- servererror.aspx
- subscribe.aspx (3)
- subscribe.swf
- survey_questions.aspx
- admin (1)
- bank (41)
- images (1)

Cross-Site Scripting

Severity:	High
Type:	Application-level tes
WASC Threat Classification:	Client-side Attacks: Scripting
CVE Reference(s):	N/A
Security Risk:	It is possible to stea customer session a which may be used a legitimate user, al hacker to view or al and to perform tran user

Visited URLs 112/112 Completed Tests 14255/14255 54 Security Issues 19

Defect Details

Credentials
Username: admin
Password:

Defect Details
Summary: SQL Injection in http://revelation/acmehackme/bank/login.aspx (Parameter passw)
id:
State:
Project:
Keywords:
Severity: 1-Critical
Symptoms:
Priority: Resolve Immediately
Owner: engineer

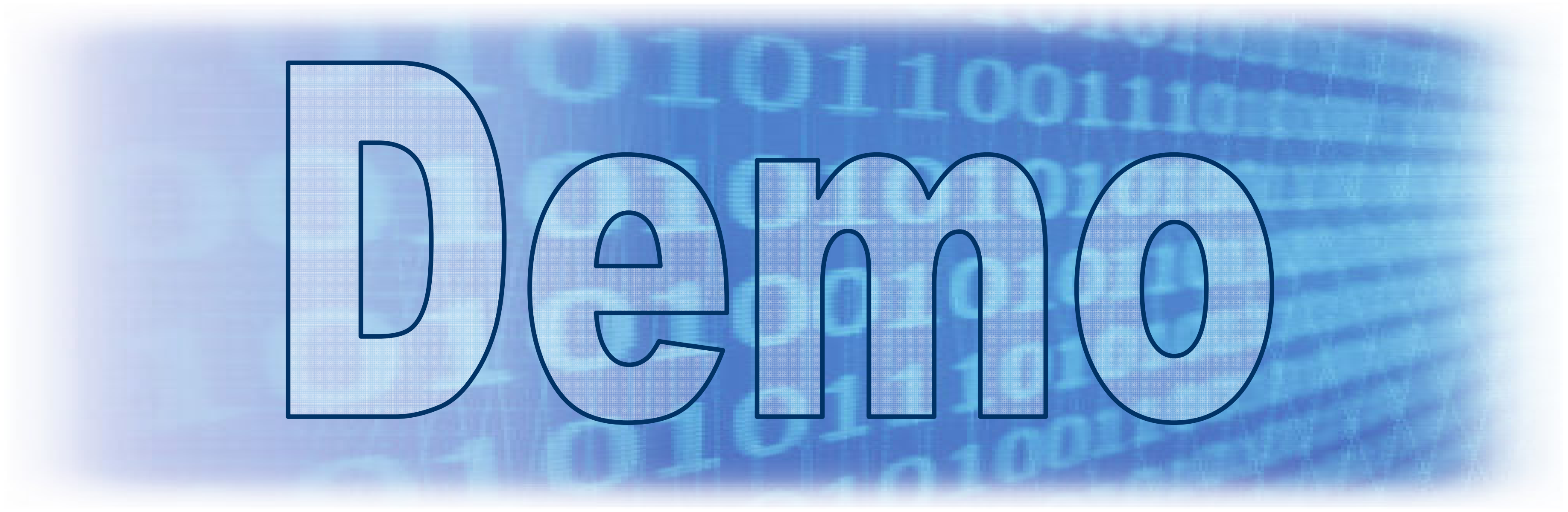
Description
SQL Injection

Application-level test

WASC Threat Classification: Command Execution: SQL Injection

Security Risk: It is possible to view, modify or delete database entries and tables

Attachments
Advisory.html FixRec.html Variant1-Ori... Variant1-Tes... Variant2-Ori... Variant2-Tes... Variant3-Ori...



What You'll See:

- AppScan @ Work
- Reporting AppScan Test Results

RSA Conference: Security Leaders Recognize Watchfire

Best Security Company - WINNER

Best SME Security Solution - WINNER

Best Regulatory Compliance Solution - WINNER

Watchfire selected for three out of only five total awards, selected by a panel of 17 expert judges made of up CISO's from companies such as Deloitte & Touche, eBay, Forrester Research, Frost & Sullivan, General Motors, KPMG, and The Burton Group

Best Audit/Vulnerability Assessment Solution - FINALIST

Best Managed Security Services - FINALIST

As selected by *SC Magazine* Readers

Company

800+ Companies Depend On Watchfire

9 of the Top 10 Banks	8 of the Top 10 Technology Companies	7 of the Top 10 Pharma / Clinical Companies	Telecommunication Companies
HSBC	Microsoft	Wyeth	AT&T
citi	Cisco Systems	NOVARTIS	Bell
AXA FINANCIAL PROTECTION	intel	AstraZeneca	cingular raising the bar
Fidelity INVESTMENTS	DELL	Abbott Laboratories	Lucent Technologies Bell Labs Innovations
SunTrust	IBM	Aventis	ROGERS
Liberty Mutual	SONY		Sprint Together with NEXTEL
Sovereign Bank	Adobe	Roche	verizon
	ANALOG DEVICES	Genentech IN BUSINESS FOR LIFE	

Multiple Large Government Agencies

DEPARTMENT OF VETERANS AFFAIRS · UNITED STATES OF AMERICA

SOCIAL SECURITY ADMINISTRATION · USA

DEPARTMENT OF THE NAVY · UNITED STATES OF AMERICA

UNITED STATES POSTAL SERVICE

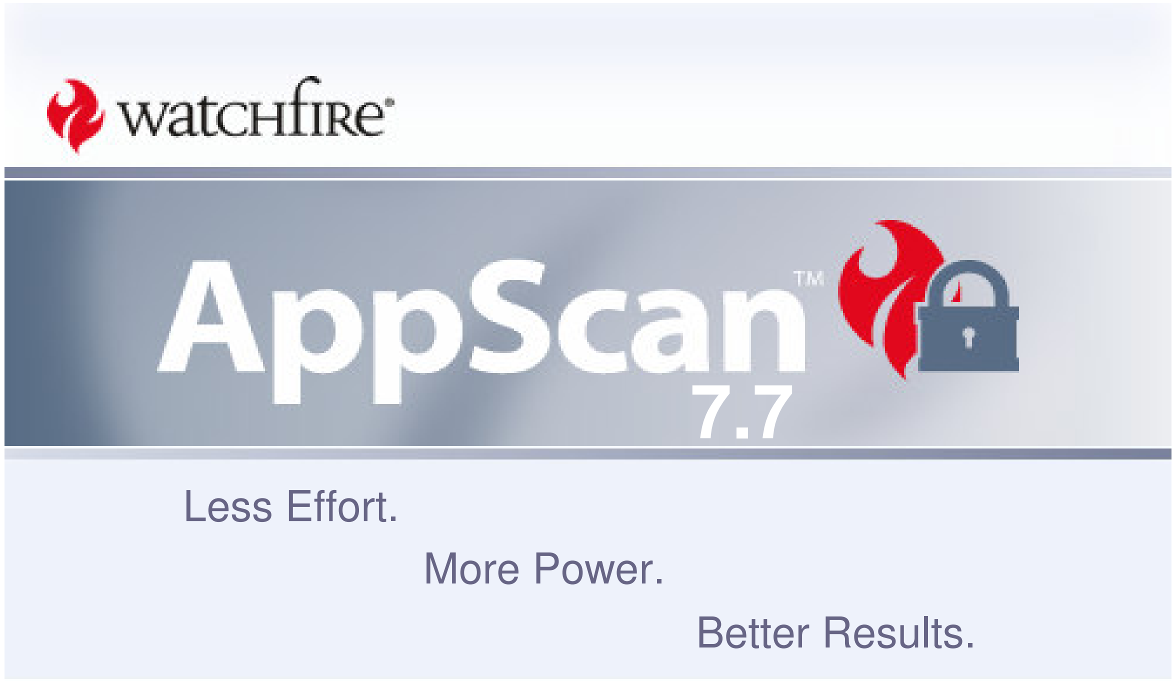
Summary

The current state

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

2/3 of All Web Applications Are Vulnerable

Gartner



Thank You

Martin Lee
IBM Rational Technical Consultant
yeekee@my.ibm.com