“Overview”

IBM Security AppScan Source for Analysis, Developer and Remediation

Arnab Roy

Ankur Bhargava

IBM Software
Innovate2012
The Premier Event for Software and Systems Innovation

Next NOW!

Agenda

- Trends in Application Security
- Introducing AppScan Source Edition
- Vulnerability Matrix
- Source and Sink View
- Traces
- Remediation Assistance
- Reporting
- Reporting Console
- Deployment Use Cases

IBM Security AppScan Source Edition

Trends in Application Security

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

Security Breaches and Newsflash

PCMAG.COM

Facebook to Encrypt UIDs After App Security Breach

By: Chloe Albanesius

10.21.2010

SearchSecurity.com

03 Aug 2012 Organizations have poor digital document security, survey reveals

At study by the Ponemon Institute shows 63% of organizations do not fully secure confidential documents.

SC MAGAZINE FOR IT SECURITY PROFESSIONALS

News

Hacker attacks against retailers up 43 percent

Much of the surge can be blamed on SQL injection and the use of exploit toolkits, according to researchers.

October 12, 2011

Security is an Enterprise Responsibility

Executives

Across 1000s of applications, from all different sources…. how do I know if and where I am at risk?

Security Analysts

How can I quickly assess applications, triage results and provide actionable guidance to my organization ?

Auditors

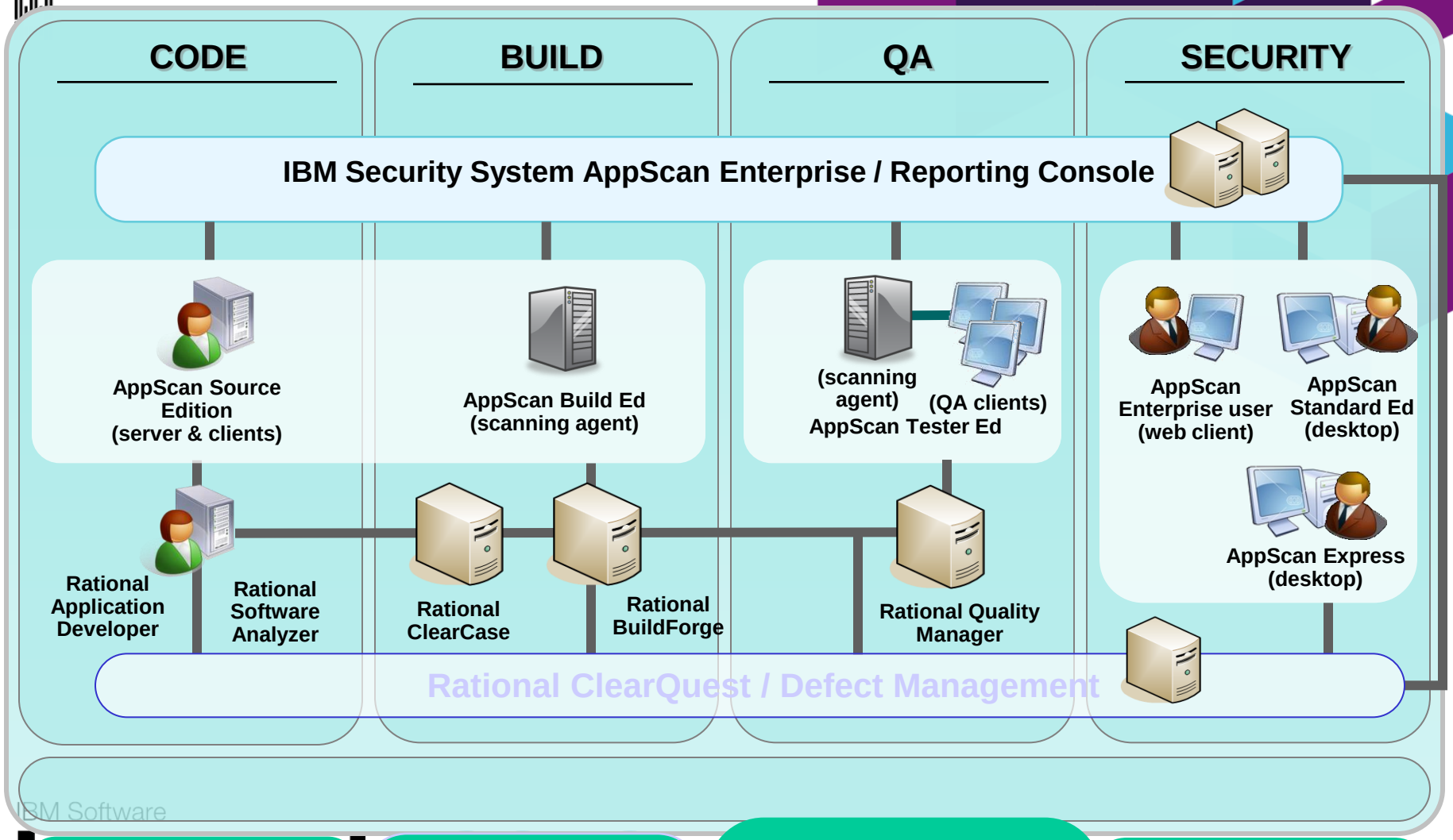
How can I quickly get the most complete and defensible assessment?

Developers

How can I prioritize and quickly clear security issues from my to do list?

Spectrum of skills and focus

IBM Security Systems AppScan – SDLC Coverage



Build security testing into the IDE*

Automate Security / Compliance testing in the Build Process

Security / compliance testing incorporated into testing & remediation workflows

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

IBM Security AppScan Source Edition

Introducing AppScan Source Edition

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

What is IBM Security AppScan Source Edition?

IBM Security AppScan Source for Analysis:

Workbench to configure applications and projects, scan code,

- ***Analyze,***
- ***Triage, and***
- ***Take action on priority vulnerabilities.***

IBM Security AppScan Source for Automation:

- ***-Allows you to automate key aspects of the Security AppScan Source workflow***
- ***- Integrate security with build environments (Ant, Make, Maven plugin) during the software development life cycle.***

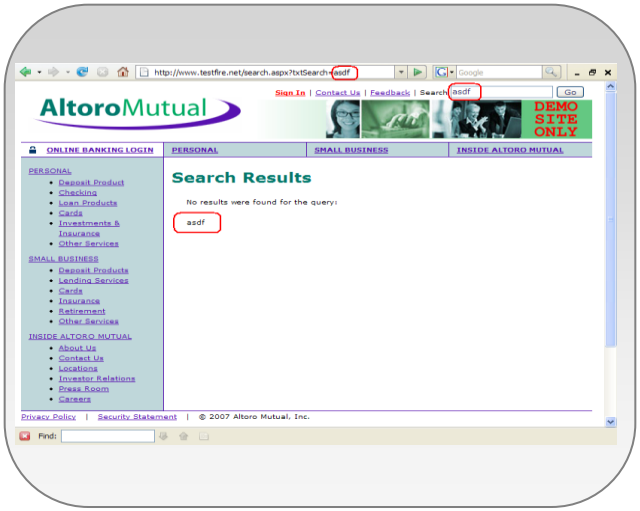
IBM Security AppScan Source for Development:

- ***Integration with***
- ***-Visual Studio,***
- ***-Eclipse workbench,***
- ***-Rational® Application Developer for WebSphere® Software (RAD)***

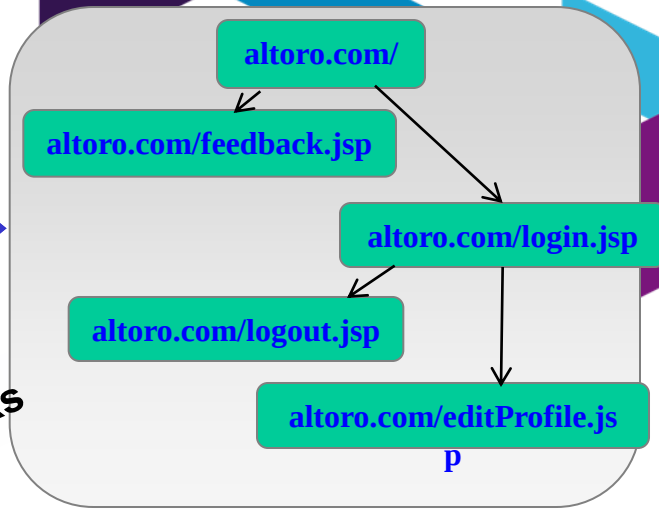
Differences Between SAST and DAST Approaches

	Static Analysis	Dynamic Analysis
Scan input	Scans source code and bytecode for security and quality issues. Requires access to source or bytecode	Scans running web applications. Requires starting point URL, and login credentials where relevant
Assessment techniques	Uses “taint analysis” and pattern matching techniques to locate issues	Tampering of HTTP messages to locate application and infrastructure layer issues
Where does it fit in application development lifecycle	Early – fits best during application development and build automation	Later – fits best in QA and security verification of production applications
Results & Output	Results are presented by line of code, source to sink functions flow	Results are presented as HTTP messages (exploit requests)

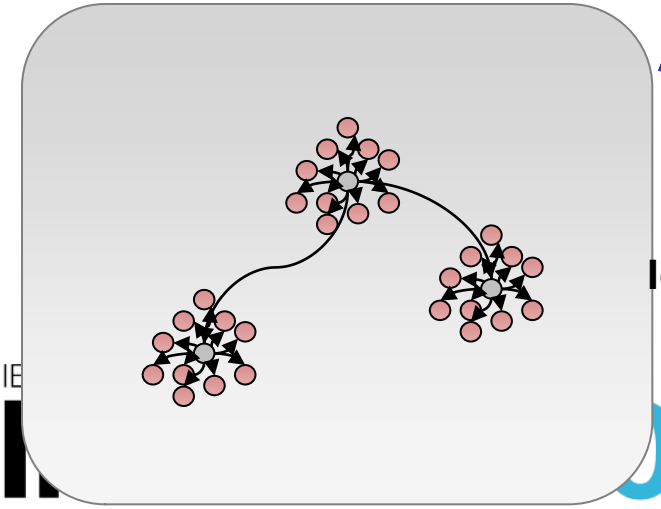
Dynamic Security Analysis – Simplified Example



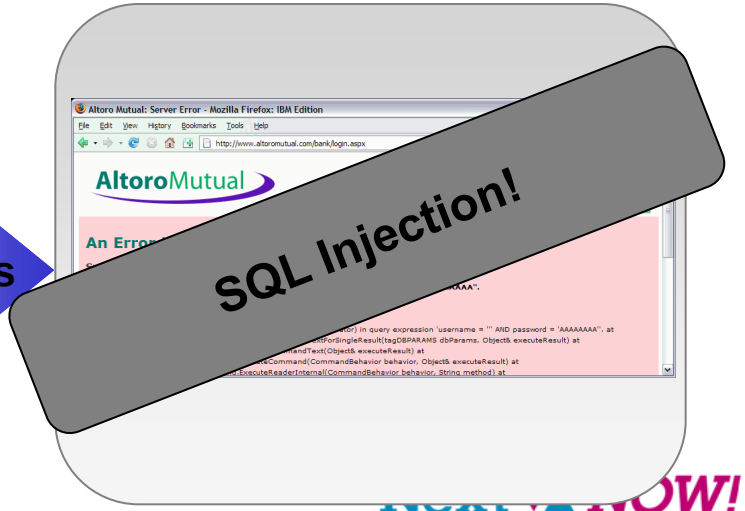
Crawl Site



Fuzz with Known Attacks

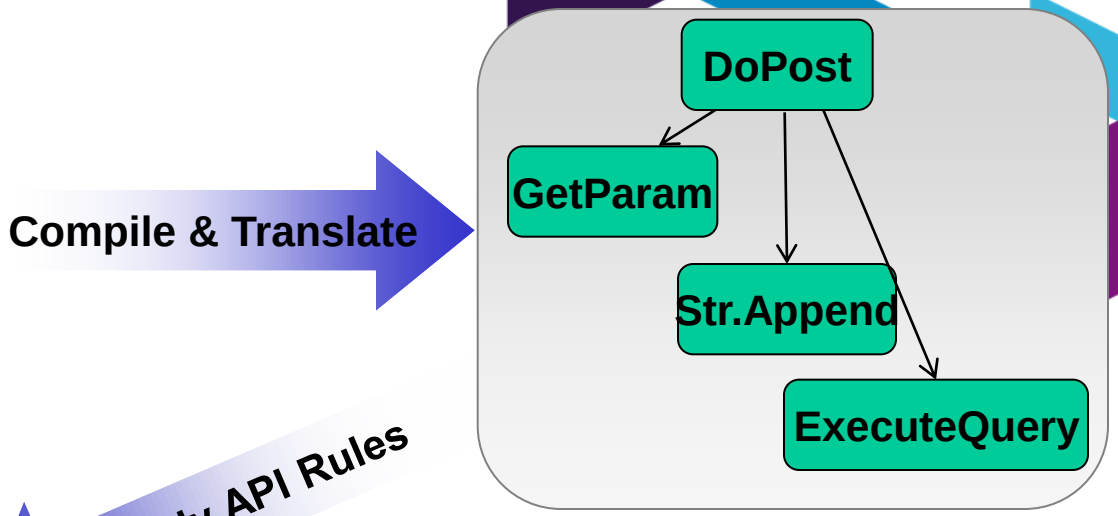


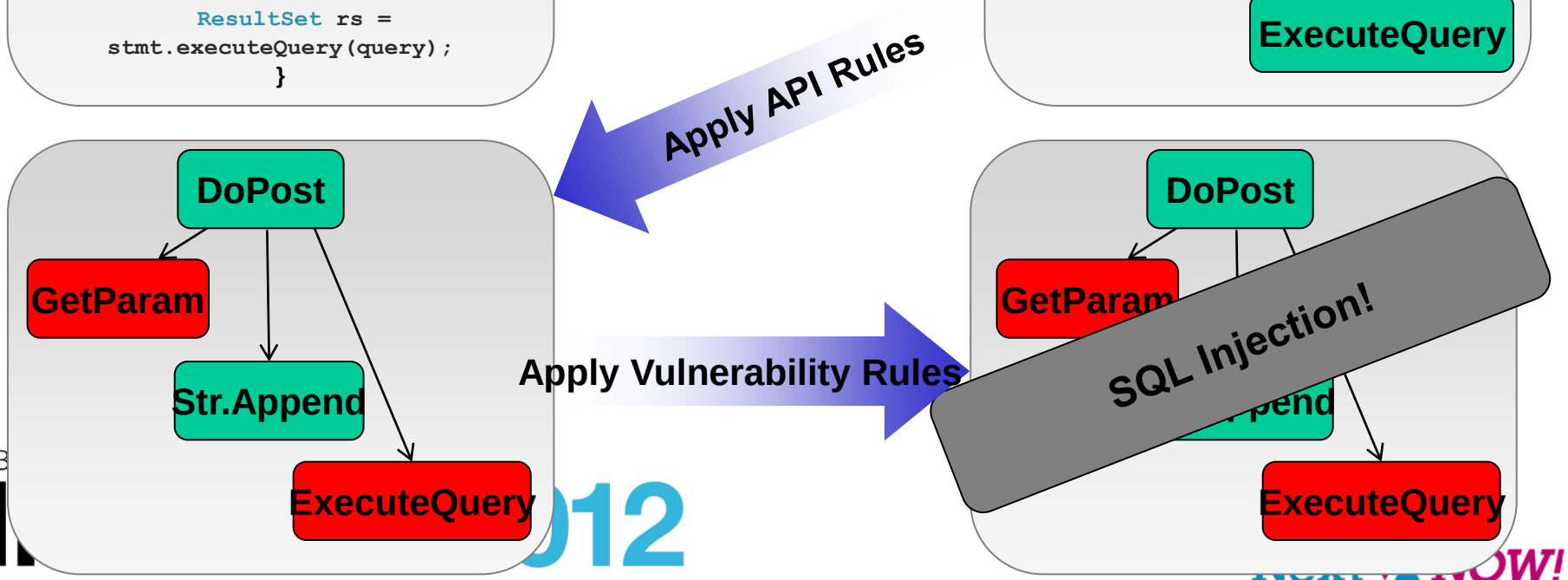
Identify Vulnerabilities



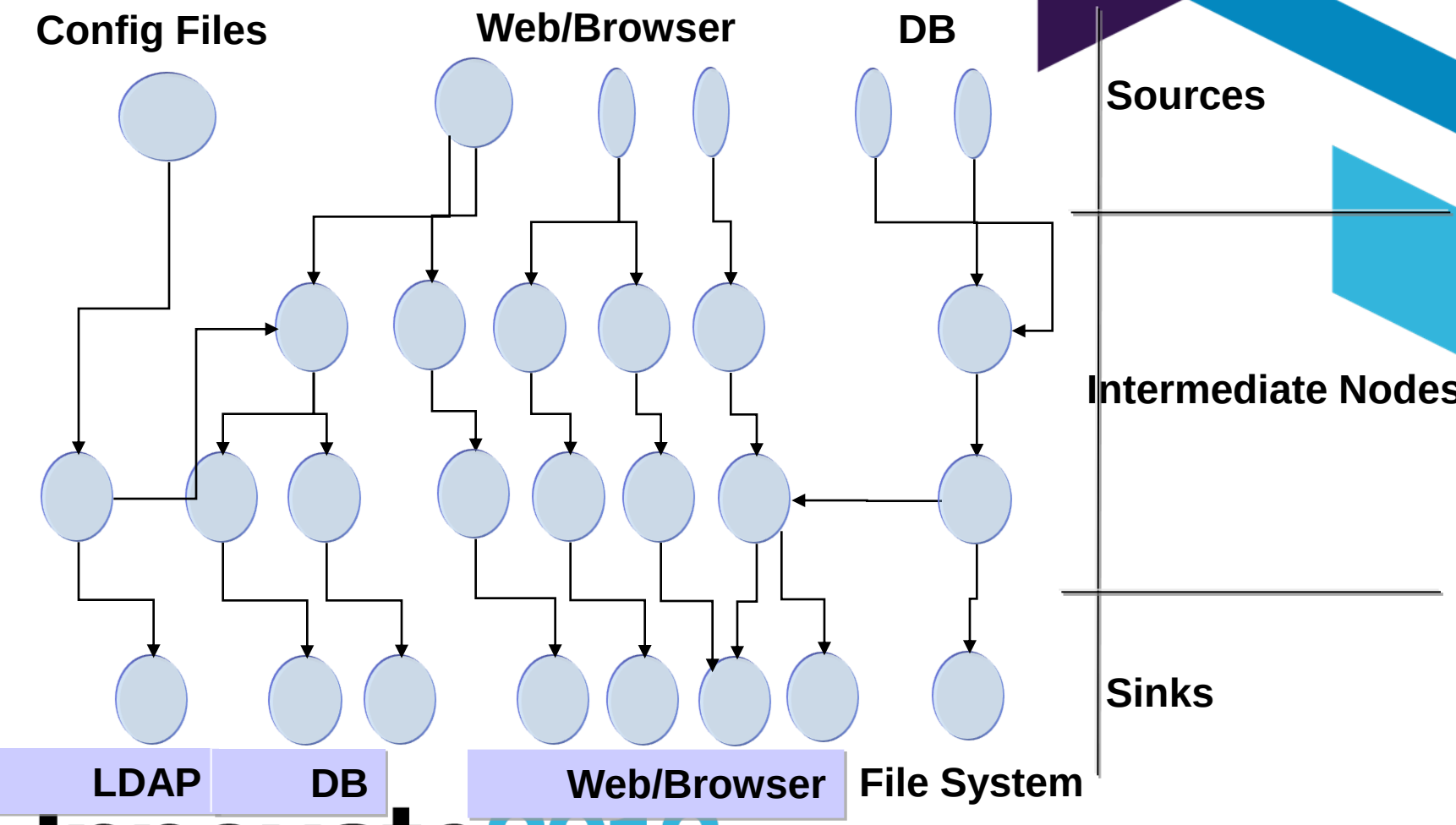
Static Security Analysis – Simplified Example

```
DoPost() {
String username =
request.getParameter("username");
String password =
request.getParameter("password");

String query = "SELECT * from
tUsers where " + "userid='" +
username + "' " + "AND
password='" + password + "'";

ResultSet rs =
stmt.executeQuery(query);
}
```





Data Flow Analysis



AppScan Source Workflow…

IBM Security AppScan Source for Analysis with Developer plug-in and Remediation

- **Scan Source Code**
- **Triage Results**
- **Resolve issues**
- **Verify fixes**

IBM Security AppScan Source Edition

Configuration

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

Broad Application Language Support

Out-of-the-Box

- **Android**
- **Java**
- **JSP**
- **C**
- **C++**
- **.NET**
- **C#**
- **VB.NET**
- **ASP.NET**
- **Classic ASP (VB6)**

- PHP
- HTML
- Perl
- ColdFusion
- Client-Side JavaScript
- Server-Side JavaScript
- VBScript

Import Eclipse and RAD Workspaces
Import .NET Solutions

Build Management Integration
Ant
Maven
Make
Automated Wizards

Scan Debug-Flagged Bytecode
Java
.NET

IBM Security AppScan Source Edition

Vulnerability Matrix

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

The Vulnerability Matrix

-Isolates confirmed vulnerabilities

-Speeds security triage

-Overcomes lack of security expertise

By separating findings by confidence level tradeoffs between false positive and false negative reduction are greatly reduced

- Type I: may have validator
- Type II: Unknown sink or no confirmed dangerous source

Confirmed Vulnerabilities

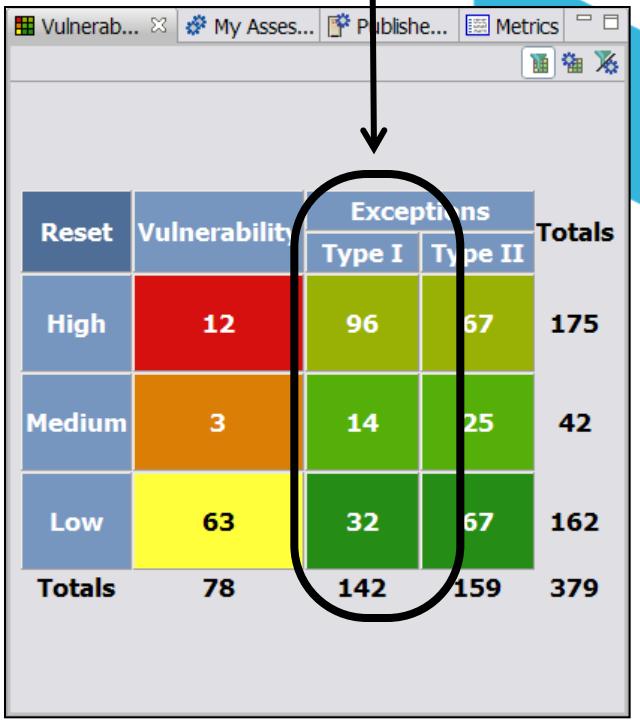
Vulnerab... | My Asses... | Publishe... | Metrics

Reset	Vulnerability	Exceptions		Totals
		Type I	Type II	
High	12	96	67	175
Medium	3	14	25	42
Low	63	32	67	162
Totals	78	142	159	379

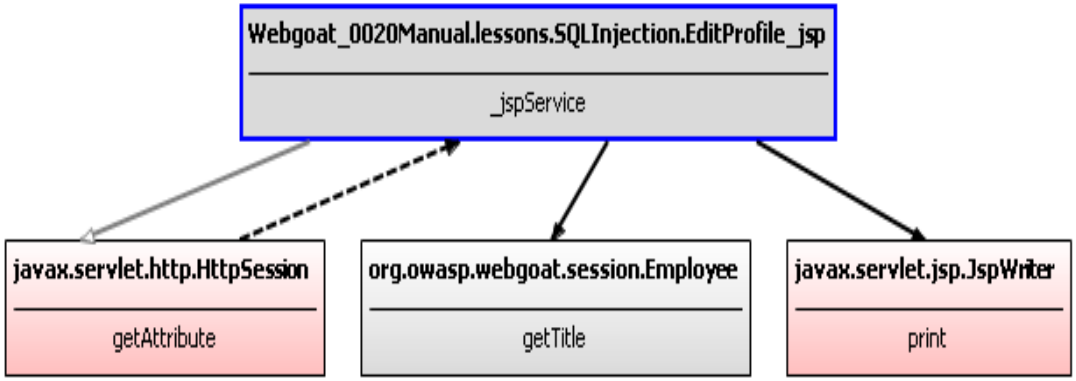
Type I Findings

- A code element or structure that can create a vulnerability when used incorrectly. A Type I Exception appears more likely to be vulnerable based on the information available to the IBM Security AppScan Source analytics.
- Gray boxes in the Trace Diagram indicate a 'taint propagator'

'Medium Trust' Findings

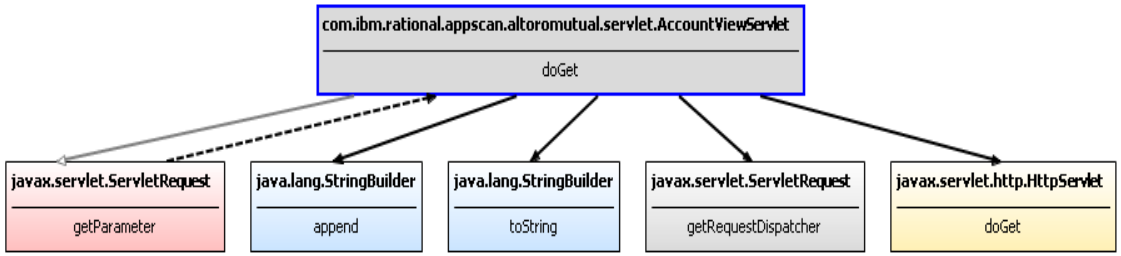


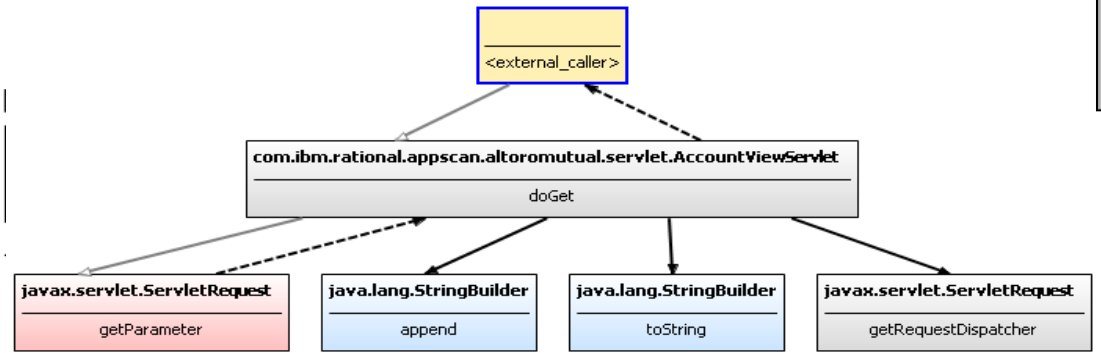
Reset	Vulnerability	Exceptions		Totals
		Type I	Type II	
High	12	96	67	175
Medium	3	14	25	42
Low	63	32	67	162
Totals	78	142	159	379



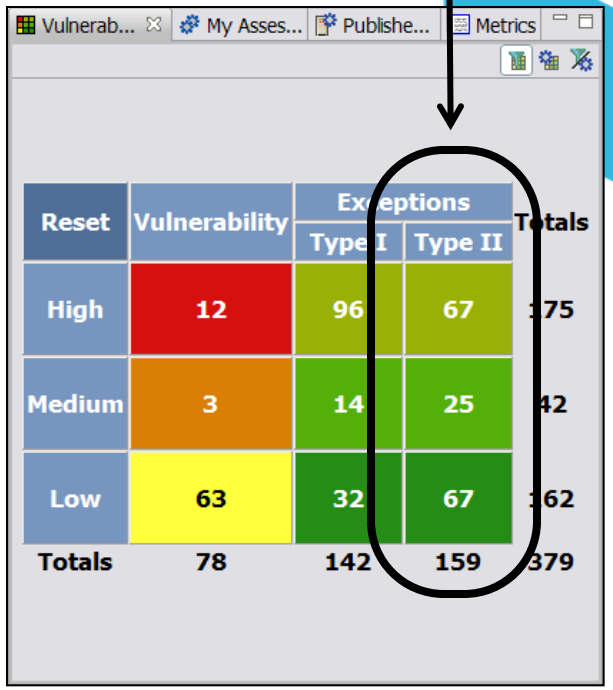
Type II Findings

- There are two types of Type II findings
- Findings without traces are dangerous sinks that potentially do not have a dangerous trace that reaches them
- Findings with traces are “lost sinks” that means that AppScan Source does not know what the risk of the trace could be because
  - -The Trace ends in a Sink that has no rule in the database
  - -The trace reaches the end of the call tree without ever hitting a potential sink





‘Low Trust’ Findings



Reset	Vulnerability	Exceptions		Totals
		Type I	Type II	
High	12	96	67	175
Medium	3	14	25	42
Low	63	32	67	162
Totals	78	142	159	379

Next NOW!

IBM Security AppScan Source Edition

Sources & Sinks View

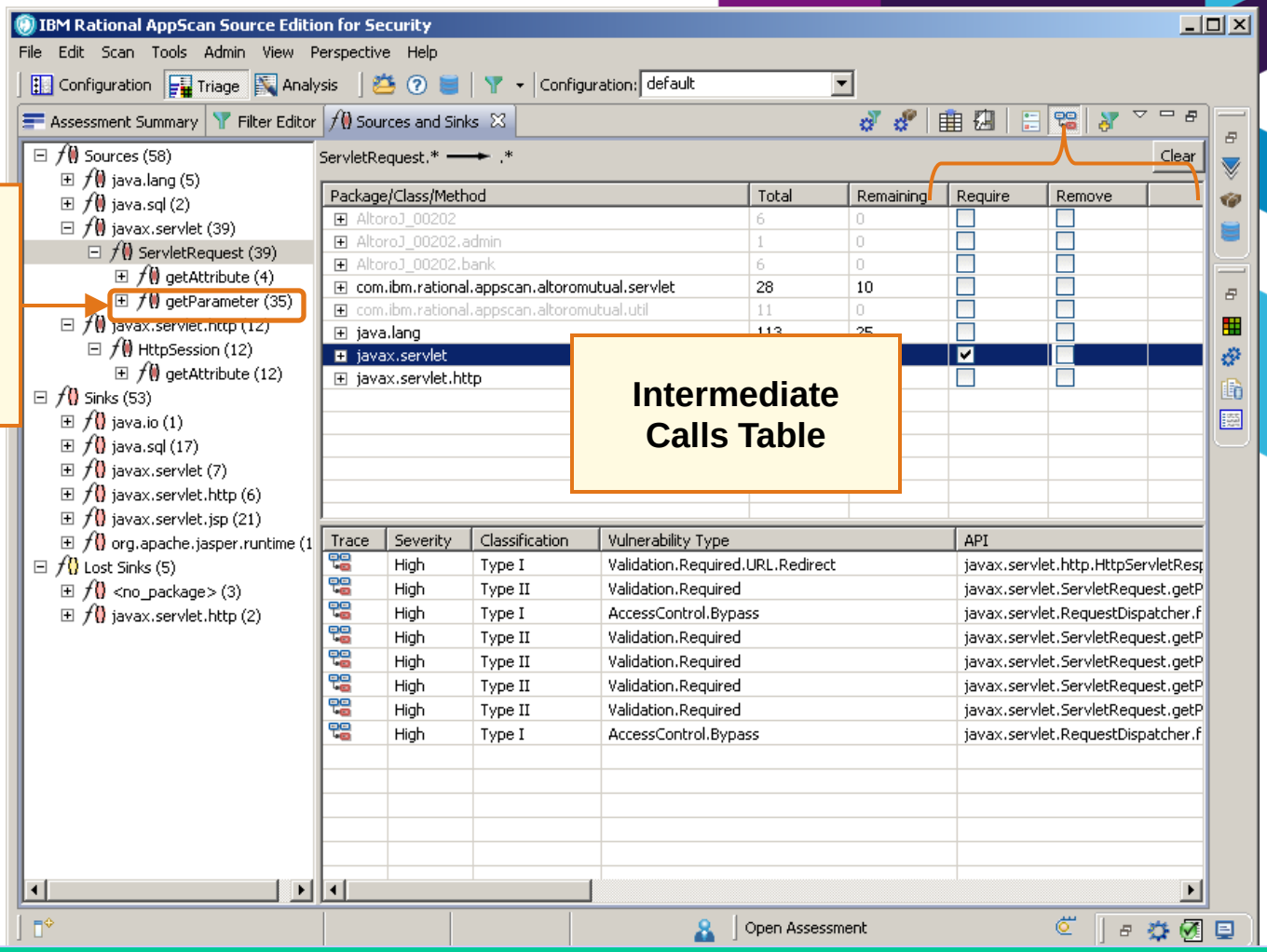
IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

Sources & Sinks View



Find Important Sources and Sinks

Intermediate Calls Table

- Review all Sources and Sinks found in an Application scan
- Alerts you to places rules may need to be set (missing sources or lost sinks entries)
- User Intermediate Calls Table to filter unwanted calls

IBM Security AppScan Source Edition

Trace

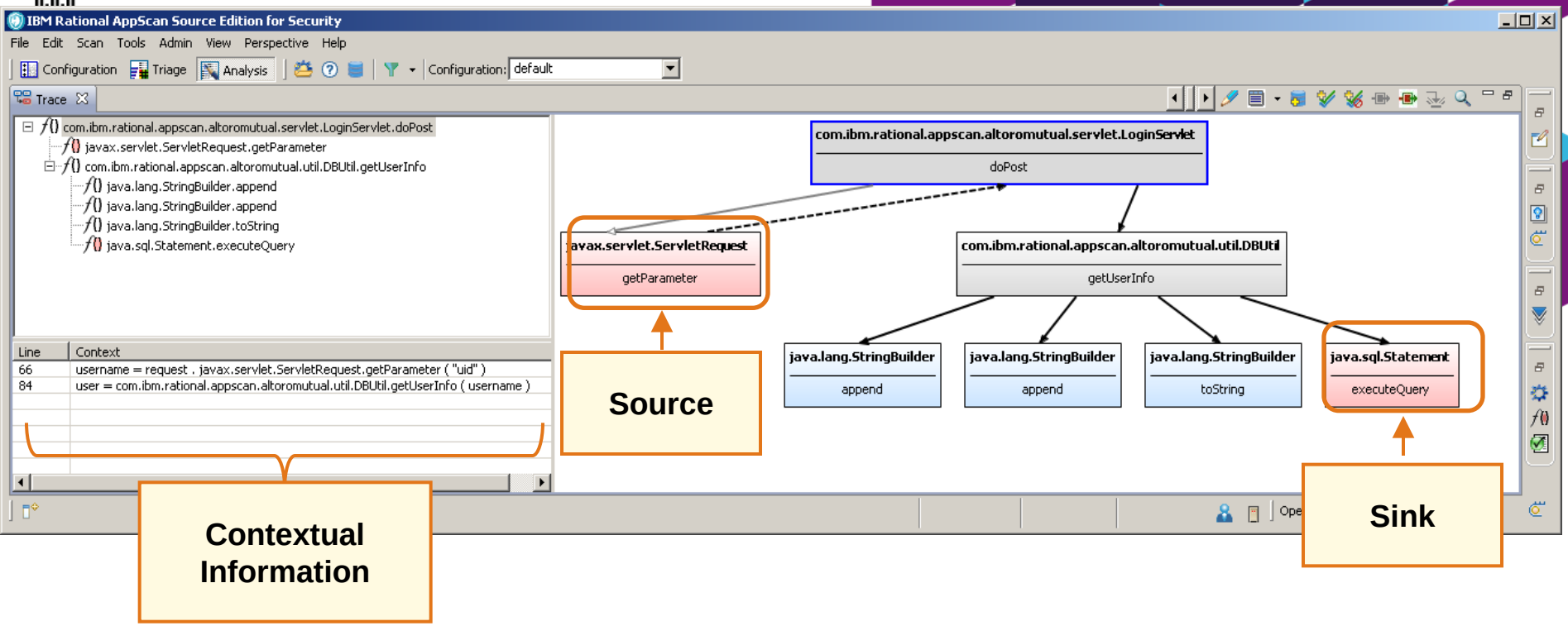
IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

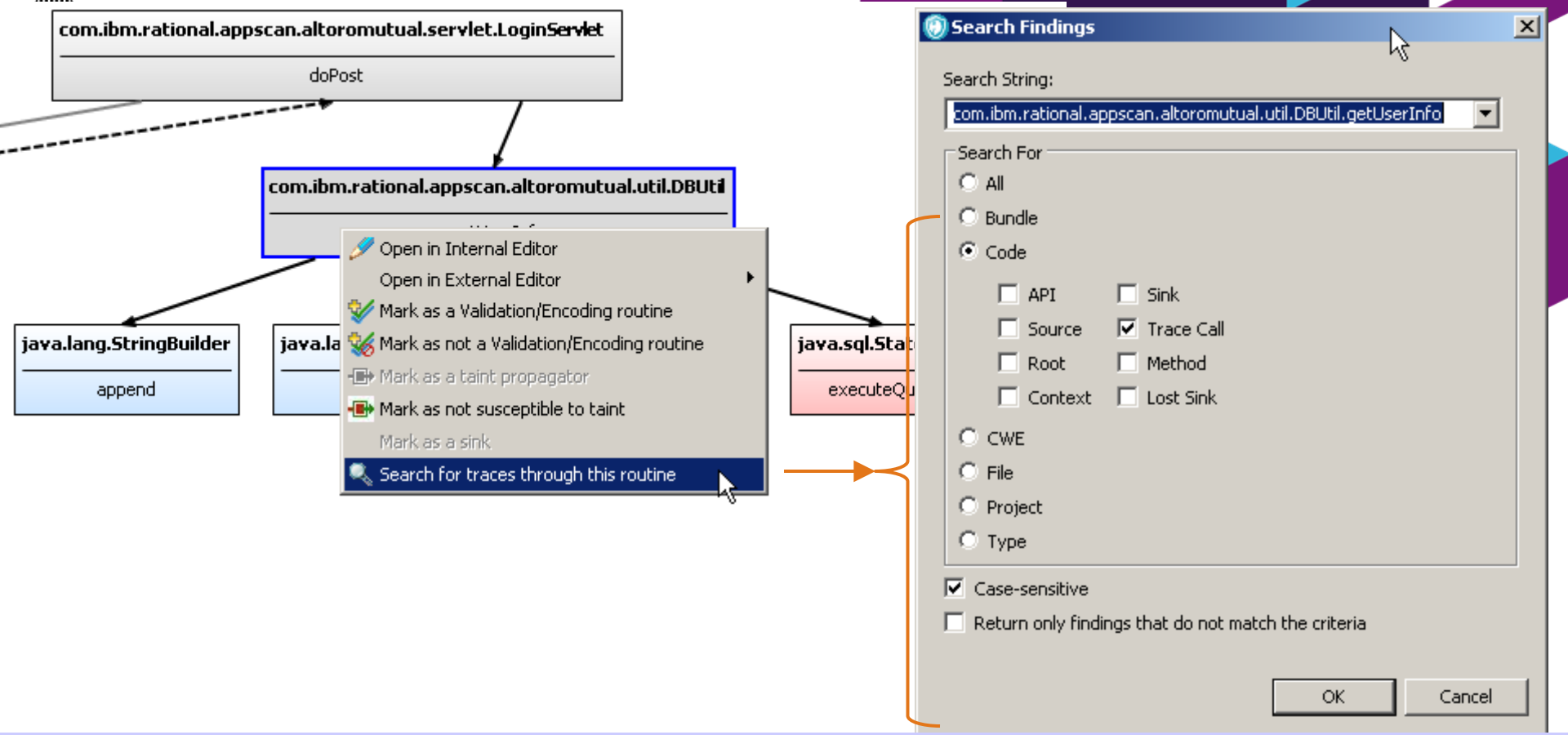
Next NOW!

Trace View



- Graphical representation of the data flow of a vulnerability from the entry point (Source) to the exit point (Sink)
- Speeds understanding of a vulnerability by seeing all ‘touch-points’ within the codebase
- Quickly review the relevant source code by double clicking each node, to take you directly to its location in the relevant source

Trace View - Searching



- Right click to on a node and click Search
  - returns all Traces with that node
- Example: Group all findings that pass through a node where validation should have occurred
- Example: Group all findings that go to a particular sink the customer is concerned about

IBM Security AppScan Source Edition

Remediation Assistance

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

Remediation Assistance

Remediation Assistance

JDBC executeQuery | Injection.SQL | Injection | Database | 89 - SQL injection

CWE Link

Vulnerability

JDBC executeQuery

The method java.sql.Statement.executeQuery() is a common interface to stored procedures, prepared statements, and dyn ed to execute dynamic SQL statements, executeQuery() is vulner All three SQL statement types may also be vulnerable to depending on the use of the data.

SQL injection is the insertion of malicious S commands generated by the application. Typically, an attacker uses SQL special characters to prematurely terminate the intended command and execute another command that requests data from a completely different column or table, or perhaps deletes or modifies such data.

The database behind a web application is a very rich target and SQL injection has been developed into an art of sorts. If a web application accepts text field values from the client, there is a strong likelihood that SQL injection attacks will be directed through it from time to time.

An easier attack through executeQuery() is to modify fields such as a user id to get access to another user's account or a price to get a super deal on merchandise.

Example *Ja*

```
final String custID = httpRequest.getParameter("custId");
final String sql = "Select * From Customer Where CustomerID = '" + custId +"'";
final Statement statement = connection.createStatement();
final boolean rsReturned = statement.execute(sql);
while (true)
{
    if (rsReturned)
    {
        ResultSet rs = s.getResultSet();
        // do something with result set
        rs.close();
    }
    if (!s.getMoreResults())
    {
        // no more results so exit loop
        break;
    }
}
s.close();
con.close();
```

Example *user inp*

- **In Context Remediation Assistance**
- **Available for every finding created by AppScan Source**

Provides

- **Information about the vulnerability**
- **Bad coding examples**
- **Good coding examples**
- **Links to the Common Weakness Enumeration (CWE) database, which is a third party maintained database with additional detailed information**

IBM Security AppScan Source Edition

Reporting

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!

Reporting Fundamentals

- IBM Security AppScan Source provides two distinct options for reporting
  - Built-In reporting allows high level reports from any *single* assessment
  - Online reporting through the IBM AppScan Reporting Console allows reporting on all scans in a collaborative web-based environment.

- Reports built into AppScan Source
  - DISA Application Security and Developerment STIG V2 R1
  - DISA Application Security and Developerment STIG V2 R1 - Checklist
  - OWASP Top Ten
  - OWASP Top Ten 2007
  - PCI Data Security Standard
  - Software Security Profile (Our own deep audit report)

- Additional 40 reports, including trending reports, in Reporting Console

Reporting in IBM Security AppScan Source – Report Example

Webgoat Manual - OWASP Top Ten 2007 Report- - Mozilla Firefox

File Edit View History Bookmarks Tools Help

file:///C:/Program%20Files/IBM/AppScan%20Source/reports/OWASP%

Getting Started Latest Headlines

Rational AppScan Source Edition Report
for the OWASP Top 10 2007

IBM

Webgoat Manual - OWASP Top Ten 2007 Report

Scan Date: Jan 19, 2011 10:37:57 PM

Report Generated: Feb 9, 2011 2:47:58 PM

OWASP Top Ten Report Card

- A1 - Cross Site Scripting (XSS)
- A2 - Injection Flaws
- A3 - Malicious File Execution
- A4 - Insecure Direct Object Reference
- A5 - Cross Site Request Forgery (CSRF)

About the OWASP Top Ten

The OWASP Top Ten describes the most critical vulnerabilities that may
the Open Web Application Security Project (OWASP).

Detailed Findings by Category

A1 - Cross Site Scripting (XSS)

Description:

XSS flaws occur whenever an application takes user suppl
validating or encoding that content. XSS allows attacker
hijack user sessions, deface web sites, etc.

This applicatio

A2 - Injectio

Done

Call Flow and Source Code

Class	Type	File	Line	CWE ID	Trace	Notes
	Injection.SQL	C:\WebGoat-5.1\project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	108	89		

```
statement.executeUpdate(arrSQL[1]);

getLessonTracker(s).setStage(2);
s.setMessage("You have succeeded in exploiting the vulnerable query and created
another SQL statement. Now move to stage 2 to
earn how to create a backdoor or a DB worm");
}

ResultSet rs = statement.executeQuery(arrSQL[0]);
if (rs.next())
{
Table t = new Table(0).setCellSpacing(0).setCellPadding(0).setBorder(1);
TR tr = new TR();
```

Method	File	Line	Tainted Arg
f() org.owasp.webgoat.lessons.BackDoors.concept1	project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	92	
f() org.owasp.webgoat.session.ParameterParser.getRawParameter	project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	92	
f() org.owasp.webgoat.session.ParameterParser.getRawParameter	project\JavaSource\org\owasp\webgoat\session\ParameterParser.java	613	
f() javax.servlet.ServletRequest.getParameterValues	project\JavaSource\org\owasp\webgoat\session\ParameterParser.java	632	
f() java.lang.StringBuilder.append	project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	95	
f() java.lang.StringBuilder.toString	project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	95	
f() java.lang.String.split	project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	96	
f() java.sql.Statement.executeQuery	project\JavaSource\org\owasp\webgoat\lessons\BackDoors.java	108	

NOTE! Remediation Assistance is not included in these reports!
Link to CWE ID present in the result redirecting to the CWE website for mitigation assistance.

IBM AppScan Reporting Console – Aggregated Reporting

Reporting – aggregation & correlation of static analysis and dynamic analysis assessment results

IBM. Rational. AppScan. Enterprise Edition

Common ASE Service Account | Help | Support | About | Log Out

Training | Jobs & Reports | Administration

Jobs & Reports > Default Folder > Altoro Assessment > Altoro - security assessments > Security Issues

Security Issues Export Email

Last Updated: 8/8/2010 10:28:18 PM

Summary | Group | Show | Search | Layout

There are **299** issues of 64 different types across 6 URLs

All items

Items 26-50 of 299 Go to page: 2 of 12 Apply

Action: Export to Excel Apply

Status	Issue	Issue Type	Test URL	Element	Sour	API	Threat Class	Type
Open	90*	Communications.Unencrypted			C:\ \Default.aspx.cs	System.Web.UI.We...	Information Disclosure: Inf...	Application
Open	20*	Cross-Site Scripting	http://revelation/acmehack...	uid			Client-side Attacks: Cross-s...	Application
Open	30*	CrossSiteScripting			C:\WebTest\Default.aspx.cs	System.Web.UI.We...	Client-side Attacks: Cross-s...	Application
Open	89*	Cryptography.InsecureAlg...			C:\WebTest\Default.aspx.cs	System.Web.UI.We...	Application Privacy Tests	Application
Open	297*	Denial-of-Service	http://revelation/acmehack...	uid			Logical Attacks: Denial of S...	Application
Open	222*	ErrorHandling.RevealDetail...			C:\WebTest\Default.aspx.cs	System.Web.UI.We...	Information Disclosure: Inf...	Application
Open	298*	File Parameter Shell Comma...	http://revelation/ac	uid			Command Execution: OS C...	Application
Open	169*	FileInclusion			C:\WebTest\Default.aspx.cs	System.Web.UI.We...	Logical Attacks: Abuse of F...	Application
Open	293*	Format String Remote Com...	http://revelation/acmehack...	uid			Command Execution: Form...	Application
Open	11*	Inadequate Account Lockout	http://revelation/acmehack...	uid			Authentication: Brute Force	Application
Open	117*	Injection					Logical Attacks: Abuse of F...	Application

Aggregated report – issue discovered using static analysis (source file, API, etc.)

Aggregated report – issue discovered using dynamic analysis (URL, element, etc.)

IBM Software

Innovate2012

The Premier Event for Software and Systems Innovation

Next NOW!