CYBER SECURITY DIVISION

2013 PRINCIPAL INVESTIGATORS'

Code Dx: Visual analytics for triage of source code vulnerabilities

Secure Decisions, a division of Applied Visions, Inc.

Anita D'Amico

Ken Prole

September 17, 2013

Homeland Security

Science and Technology

Secure Decisions

SECURE DECISIONS
A DIVISION OF APPLIED VISIONS, INC.

We help you **make sense of data**

- Analyze security *decision-making* processes
- Build *visual analytics* to enhance security decisions and training

Our expertise starts where automated security sensors and scanners leave off

We **transition** our R&D into **operational use**, in government and industry

Secure SCOPE

VIASSIST

MeerCAT

Grounded in commercial software and product development

- Division of Applied Visions, developer of commercial software
- 40 people, most with clearances, and secure facilities

Hackers are paid bounties to find software flaws

The New York Times

JULY 13, 2013

"In 2010, Google started paying hackers up to $3,133.70 ... for bugs in its Web browser.

Last month, Microsoft sharply increased the amount it was willing to pay for such flaws, raising its top offer to $150,000.

Software Assurance
SwA Terminology

Weakness Source code defect that an attacker *might* exploit

Vulnerability Source code defect *known* to be exploitable

- For simplicity, we'll use "vulnerability" in this presentation

SAST Static Application Security Testing tools

- Find vulnerabilities and poor quality in static source code
- Rapidly growing market
  Commercial: Fortify, AppScan, Armorize, Coverity ...
  Open source: FindBugs, Jlint, cppcheck ...

Focus of Code Dx

Other categories of tools

DAST Dynamic Application Security Testing tools

- Penetration testing of web applications during execution

Binary code analysis

- Finding vulnerabilities through analysis of complied code

The Need
Stop shipping insecure software

> **90%** of reported security incidents result from **exploits** of application software **defects**
>
> **Build Security In Website, DHS**
> https://buildsecurityin.us-cert.gov/bsi/mission.html

On average, one SAST tool finds only **14%** of vulnerabilities; you need lots of different tools to **cover** the vulnerabilities

50,000 weaknesses in 200,000 lines of code ...
Where do I start? What's most important?

Code Dx Approach
Find the most important *vulnerabilities*

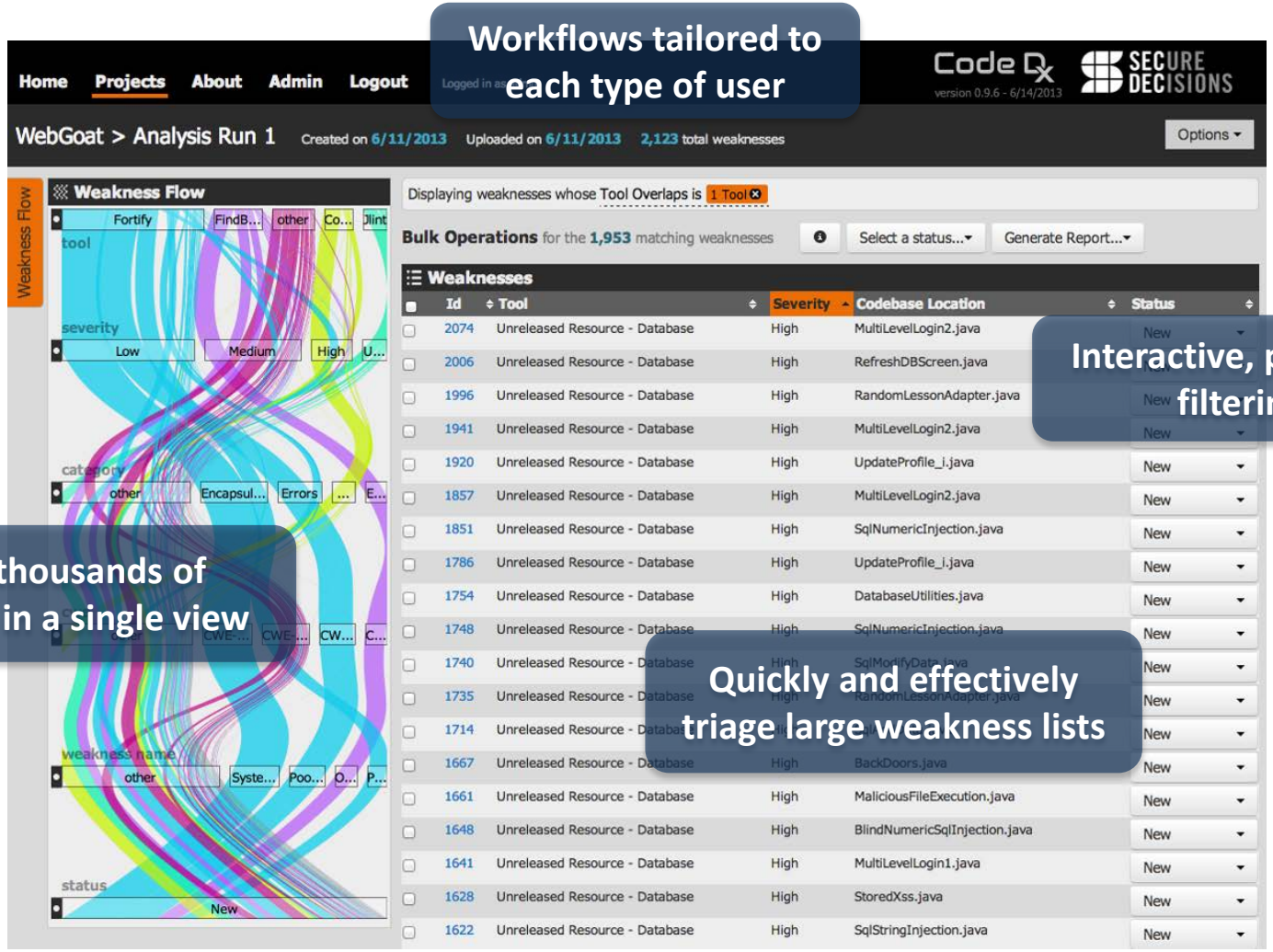
Challenge

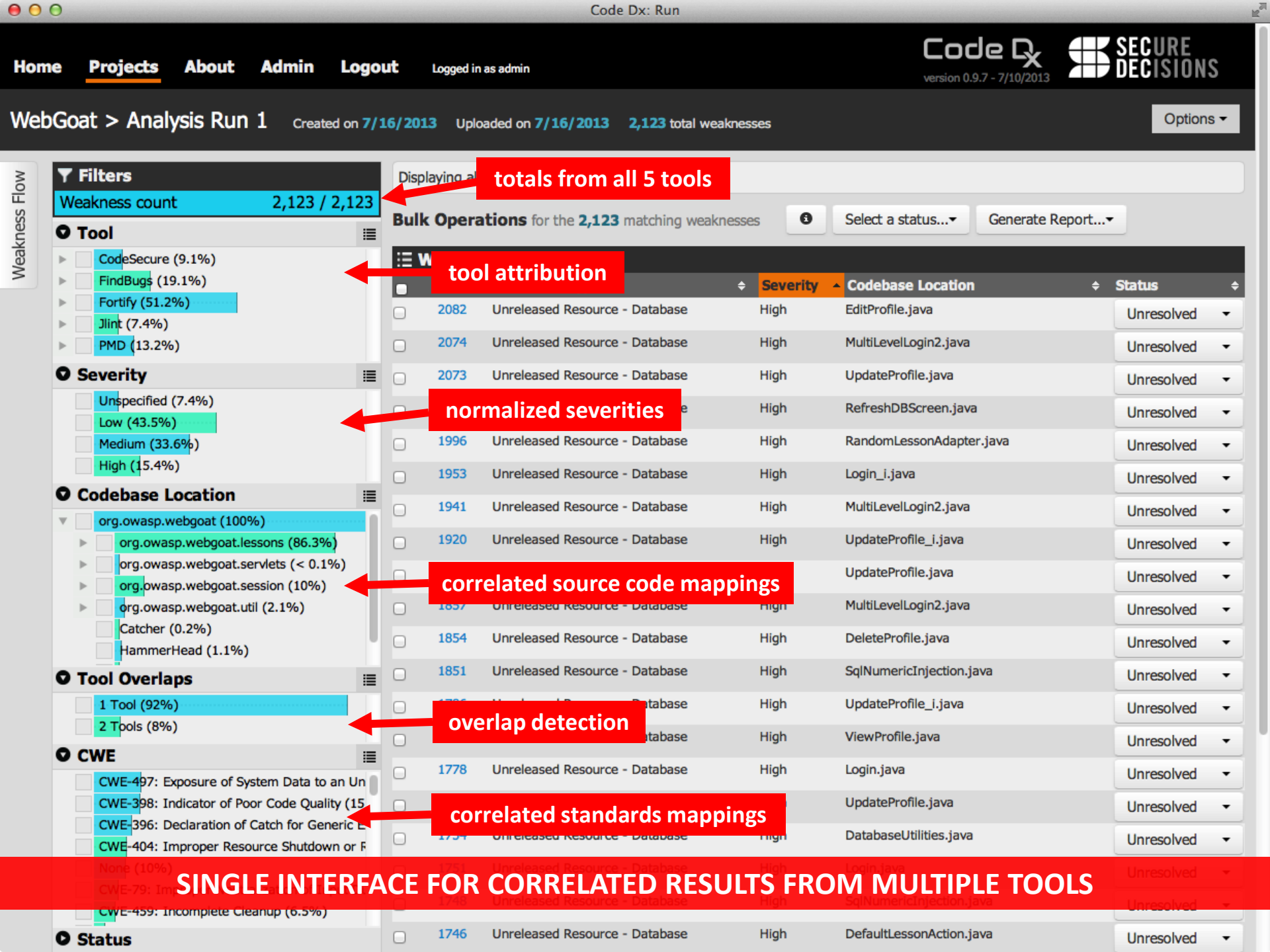
- Incomplete vulnerability coverage by single tool
- Difficult to compare tool results; different semantics
- Tens of thousands of vulnerabilities reported
- Format of results impedes communication and collaboration
- Expensive tools; hard to use for non-experts

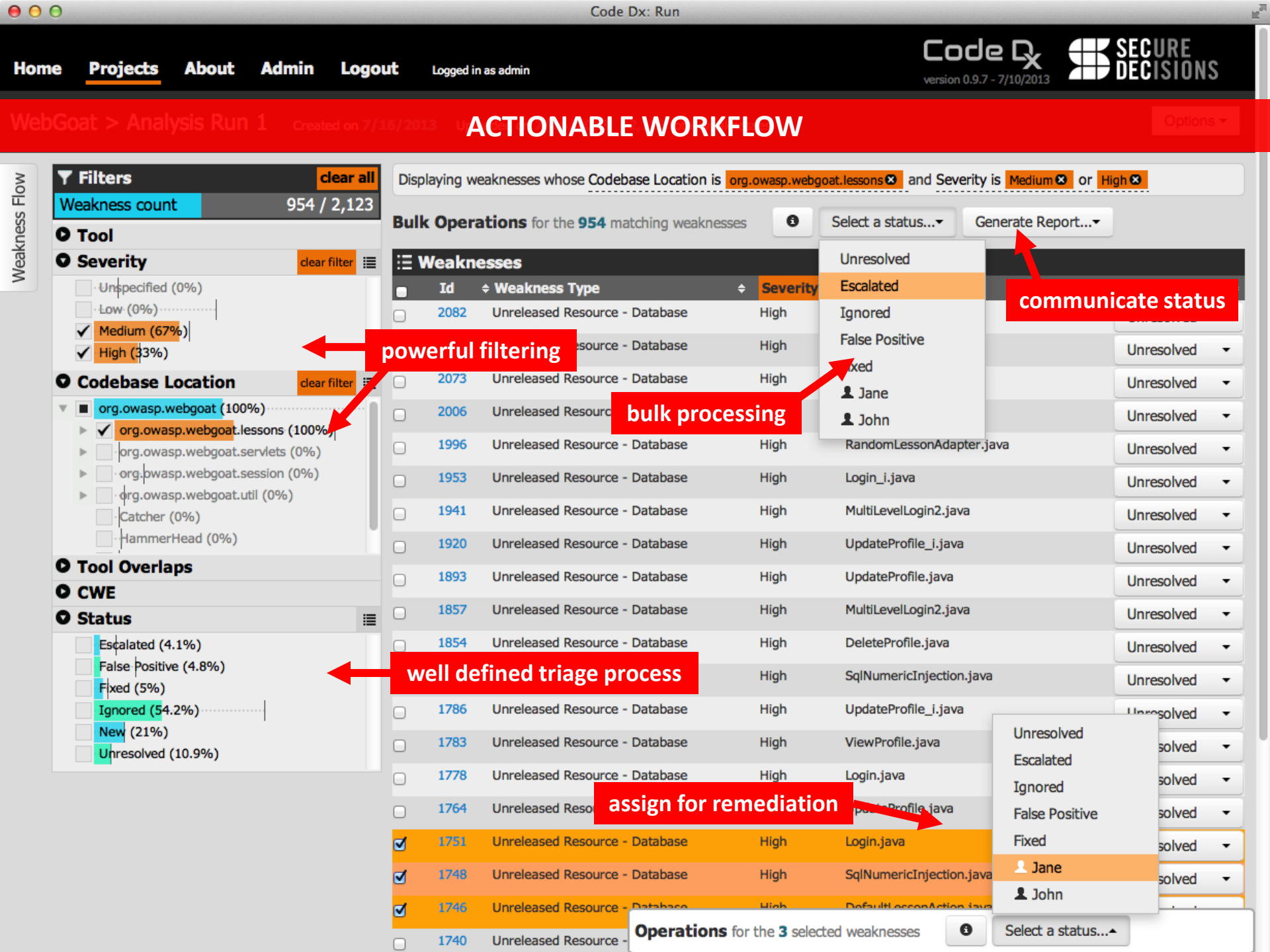
Code Dx Solution

- Imports and correlates results from multiple tools
- Normalizes results; common severity scale
- Visual analytics to rapidly triage results
- Common UI with custom detail for security analysts, developers, and CISOs
- *Code Dx* will embed open source SAST tools for use with or without commercial tools

Visual Analytics
for triage, remediation, and communication




Code Dx: Run
Home
Projects
About
Admin
Logout
Logged in as admin
Code Dx
version 0.9.7 - 7/10/2013
SECURE DECISIONS
WebGoat > Analysis Run 1
Created on 7/16/2013
Uploaded on 7/16/2013
2,123 total weaknesses
Options
Weakness Flow
Filters
Weakness count 2,123 / 2,123
totals from all 5 tools
Tool
CodeSecure (9.1%)
FindBugs (19.1%)
Fortify (51.2%)
Jlint (7.4%)
PMD (13.2%)
tool attribution
Severity
Unspecified (7.4%)
Low (43.5%)
Medium (33.6%)
High (15.4%)
normalized severities
Codebase Location
org.owasp.webgoat (100%)
org.owasp.webgoat.lessons (86.3%)
org.owasp.webgoat.servlets (< 0.1%)
org.owasp.webgoat.session (10%)
org.owasp.webgoat.util (2.1%)
Catcher (0.2%)
HammerHead (1.1%)
correlated source code mappings
Tool Overlaps
1 Tool (92%)
2 Tools (8%)
overlap detection
CWE
CWE-497: Exposure of System Data to an Un
CWE-398: Indicator of Poor Code Quality (15
CWE-396: Declaration of Catch for Generic E
CWE-404: Improper Resource Shutdown or F
correlated standards mappings
Status
Displaying all
Bulk Operations for the 2,123 matching weaknesses
Select a status...
Generate Report...
Severity
Codebase Location
Status
2082 Unreleased Resource - Database High EditProfile.java Unresolved
2074 Unreleased Resource - Database High MultiLevelLogin2.java Unresolved
2073 Unreleased Resource - Database High UpdateProfile.java Unresolved
High RefreshDBScreen.java Unresolved
1996 Unreleased Resource - Database High RandomLessonAdapter.java Unresolved
1953 Unreleased Resource - Database High Login_i.java Unresolved
1941 Unreleased Resource - Database High MultiLevelLogin2.java Unresolved
1920 Unreleased Resource - Database High UpdateProfile_i.java Unresolved
UpdateProfile.java Unresolved
MultiLevelLogin2.java Unresolved
1854 Unreleased Resource - Database High DeleteProfile.java Unresolved
1851 Unreleased Resource - Database High SqlNumericInjection.java Unresolved
High UpdateProfile_i.java Unresolved
High ViewProfile.java Unresolved
1778 Unreleased Resource - Database High Login.java Unresolved
UpdateProfile.java Unresolved
DatabaseUtilities.java Unresolved
1746 Unreleased Resource - Database High DefaultLessonAction.java Unresolved
SINGLE INTERFACE FOR CORRELATED RESULTS FROM MULTIPLE TOOLS


Code Dx: Run
Home
Projects
About
Admin
Logout
Logged in as admin
Code Dx
version 0.9.7 - 7/10/2013
SECURE DECISIONS
WebGoat > Analysis Run 1
ACTIONABLE WORKFLOW
Options
Weakness Flow
Filters
clear all
Weakness count
954 / 2,123
Tool
Severity
clear filter
Unspecified (0%)
Low (0%)
Medium (67%)
High (33%)
powerful filtering
Codebase Location
clear filter
org.owasp.webgoat (100%)
org.owasp.webgoat.lessons (100%)
org.owasp.webgoat.servlets (0%)
org.owasp.webgoat.session (0%)
org.owasp.webgoat.util (0%)
Catcher (0%)
HammerHead (0%)
Tool Overlaps
CWE
Status
Escalated (4.1%)
False Positive (4.8%)
Fixed (5%)
Ignored (54.2%)
New (21%)
Unresolved (10.9%)
well defined triage process
Displaying weaknesses whose Codebase Location is org.owasp.webgoat.lessons and Severity is Medium or High
Bulk Operations for the 954 matching weaknesses
Select a status...
Generate Report...
Unresolved
Escalated
Ignored
False Positive
Fixed
Jane
John
communicate status
bulk processing
Weaknesses
Id
Weakness Type
Severity
2082 Unreleased Resource - Database High
2073 Unreleased Resource - Database High
2006 Unreleased Resource - Database High
1996 Unreleased Resource - Database High RandomLessonAdapter.java
1953 Unreleased Resource - Database High Login_i.java
1941 Unreleased Resource - Database High MultiLevelLogin2.java
1920 Unreleased Resource - Database High UpdateProfile_i.java
1893 Unreleased Resource - Database High UpdateProfile.java
1857 Unreleased Resource - Database High MultiLevelLogin2.java
1854 Unreleased Resource - Database High DeleteProfile.java
High SqlNumericInjection.java
1786 Unreleased Resource - Database High UpdateProfile_i.java
1783 Unreleased Resource - Database High ViewProfile.java
1778 Unreleased Resource - Database High Login.java
1764 Unreleased Reso High
1751 Unreleased Resource - Database High Login.java
1748 Unreleased Resource - Database High SqlNumericInjection.java
1746 Unreleased Resource - Database High
1740 Unreleased Resource -
Unresolved
assign for remediation
Unresolved
Escalated
Ignored
False Positive
Fixed
Jane
John
Operations for the 3 selected weaknesses
Select a status...

WebGoat > Analysis Run 1 > Weakness 944

SQL_NONCONSTANT_STRING_PASSED_TO_EXECUTE detected by FindBugs with High severity

First seen on 7/16/2013 19 weaknesses in this file 13 similar weaknesses in this analysis run

CWE 89 - Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') [CWEVis | Mitre]

jump to weakness

Status

John

Activity Stream

@Jane You're right. I'll fix this right away!

You can write your comments using markdown

Jane commented

@John Looks like we didn't sanitize the user's input!

2 minutes ago

Jane changed status to **Assigned to John**

4 minutes ago

admin changed status to **New**

3 days ago

- 944 **(Current Weakness)**
- 302 **SQL** line 101
- 538 **CheckResultSet** line 101

```
68      */
69     protected Element createContent(WebSession s)
70     {
71         return super.createStagedContent(s);
72     }
73
74     protected Element doStage1(WebSession s) throws Exception
75     {
76         return injectableQuery(s);
77     }
78
79     protected Element doStage2(WebSession s) throws Exception
80     {
81         return parameterizedQuery(s);
82     }
84     protected Element injectableQuery(WebSession s)
85     {
86         ElementContainer ec = new ElementContainer();
87
88         try
89         {
90             Connection connection = DatabaseUtilities.getConnection(s);
91
92             ec.addElement(makeAccountLine(s));
93
94             String query = "SELECT * FROM user_data WHERE last_name = '" + accountName + "'";
95             ec.addElement(new PRE(query));
97             try
98             {
99                 Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
100                                                                  ResultSet.CONCUR_READ_ONLY);
101                ResultSet results = statement.executeQuery(query);
102
103                if ((results != null) && (results.first() == true))
104                {
105                    ResultSetMetaData resultsMetaData = results.getMetaData();
106                    ec.addElement(DatabaseUtilities.writeTable(results, resultsMetaData));
107                    results.last();
108
109                    // If they get back more than one user they succeeded
110                    if (results.getRow() >= 6)
111                    {
112                        makeSuccess(s);
115                    StringBuffer msg = new StringBuffer();
116
```

detailed guidance

real-time collaboration

correlated weaknesses

impacted source

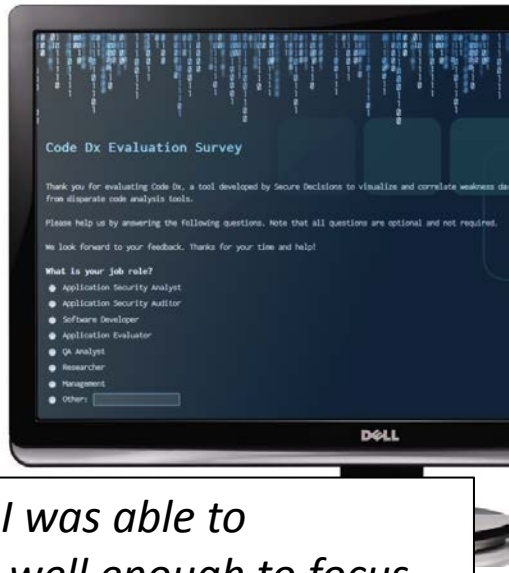
DEVELOPER FOCUSED REMEDIATION

Benefits

- ***Better Coverage*** - Find **more important vulnerabilities**
  - **Combine** multiple tool results to find **more** vulnerabilities
  - **Prioritize** combined results to highlight **most important**
  - **Filter out** overlapping results and false positives
- ***Efficiency* -** Save remediation **time** and **resources**
  - Developers can remediate **highest priority** vulnerabilities first
    - *Remediation can take 7–10 hours per vulnerability*
- ***Communicate*** more effectively up and down the chain
  - Visual analytics and reports, based on **roles** and **expertise**
- ***Easy*** to get started
  - *Code Dx 1.0* (Q4 2013) **auto-runs** open source tools
  - **Affordable** to small and mid-sized businesses

Current Status
Technology Readiness Level 7

- More than **10 beta testers**, incl. ITT, NSA, Raytheon, RTI, Univ. of Nebraska …
- Systematic collection of **feedback**



"I really like the visualization. ..tying [the tools] together and being able to work with that data is very useful."

"…at the present state, it seems to require the user to do a lot of work and formatting that the software itself could do."

"After a few minutes, I was able to manipulate the filters well enough to focus on particular discoveries."

- Currently being **evaluated** by NIST, DHS S&T CIO, TSA, McAfee, Domestic Nuclear Detection Office, Indiana Univ.
- Working with Morgridge Institute to integrate into **SWAMP**
- **Training** program being refined

Code Dx Roadmap

Version	Major Features	Target Users	13	2014				2015			
			Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Code Dx 1.0	Embeds and **auto-runs open source tools**; produces consolidated normalized results	Users seeking low-cost, easy-to-use bundle of SAST tools (e.g. small biz)	▲								
Code Dx 1.1	More open source tools with **additional languages**, e.g. .Net; **Enhanced reporting**	As above			▲						
Enterprise 1.0	Correlates **commercial and open source SAST tools**; Enhanced analytics/reporting	Users of commercial SAST who want to extend their vulnerability coverage	▲								
Enterprise 1.5	**Dynamic tracing** from Code Pulse; **enhanced visual analytics**	As above			▲						
Enterprise 2.0	Maps weaknessses to **compliance** (e.g. HIPAA) and industry standards (e.g. OWASP Top Ten)	As above, plus industry verticals, e.g. health					▲				
Enterprise 3.0	Addition of **hybrid analysis** correlating SAST and DAST	As above, with focus on web application risk assessment									▲
SWAMP 1.0	More tool adaptors; Modified for **SWAMP beta**	SWAMP users	▲								
SWAMP 1.1	Upgrades for **SWAMP IOC**	SWAMP users		▲							
SWAMP 2.0	SWAMP Yr 2 added **scalability** and functions	SWAMP users					▲				
SWAMP 3.0	Upgrades for SWAMP Yr 3 custom needs	SWAMP users									▲
SIEM Beta	**Feed pre-correlated SAST data to SIEM**; Beta to be integrated into McAfee ESM	SIEM vendors; SIEM users	▲								
ED 1.0	**Free** version of Code Dx **for education**	Academia; training organizations			▲						

Next 120 days towards transition

1. Transition Code Dx into **government programs**
   - Integrate into SWAMP Beta version
   - Determine effectiveness in NIST SATE program
   - Have other government agencies evaluate Code Dx
2. Initiate **operational pilots** to reach TRL 8
   - Recovery Accountability and Transparency Board; Commonwealth of PA
3. Conduct full **commercialization**
   - Integrate with McAfee ESM to demonstrate value proposition to Security Information Event Management (SIEM) users
   - Determine pricing model; Set up reseller program
   - Gain active use by at least one Fortune 500 company
4. Continue collaboration with academia

What do you think?

Diagnosis and triage of source code vulnerabilities

Anita D'Amico, Ph.D.
Director, Secure Decisions
(631) 759-3909
anita.damico@securedecisions.com

Ken Prole
Principal Investigator
(631) 759-3907
ken.prole@securedecisions.com