Fortify Software Security Center

Developer Workbook

demoaa - v1

Table of Contents

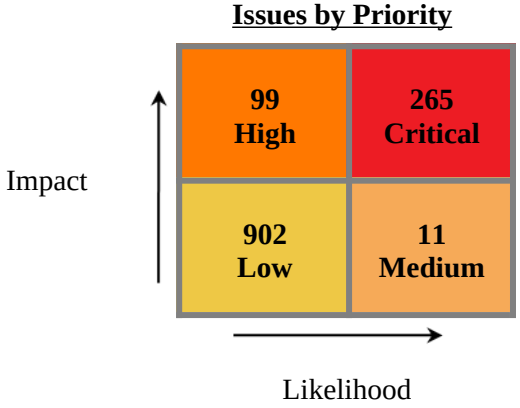
Executive Summary

This workbook is intended to provide all necessary details and information for a developer to understand and remediate the different issues discovered during the demoaa - v1 project audit. The information contained in this workbook is targeted at project managers and developers.

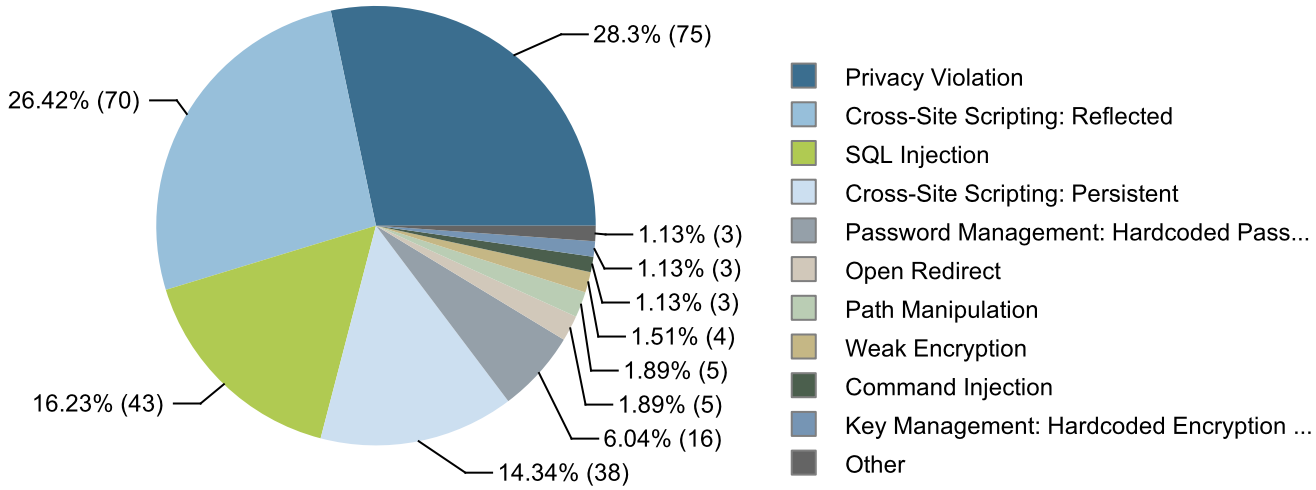
This section provides an overview of the issues uncovered during analysis.

Project Name:	demoaa
Project Version:	v1
SCA:	Results Present
WebInspect:	Results Not Present
WebInspect Agent:	Results Not Present
Other:	Results Not Present

Issues by Priority



Top Ten Critical Categories



Project Description

This section provides an overview of the Fortify scan engines used for this project, as well as the project meta-information.

SCA

Date of Last Analysis:	Oct 10, 2018, 4:15 PM	**Engine Version:**	18.10.0192
Host Name:	Zacharys-MacBook-Air.local	**Certification:**	VALID
Number of Files:	190	**Lines of Code:**	16,926

Accessibility	Internal Network Access Required
Application Classification	<none>
Authentication System	<none>
Business Risk	High
Data Classification	<none>
Development Languages	<none>
Development Phase	Active Development
Development Strategy	Internally Developed
Interfaces	<none>
Known Compliance Obligations	<none>

Issue Breakdown by Fortify Categories

The following table depicts a summary of all issues grouped vertically by Fortify Category. For each category, the total number of issues is shown by Fortify Priority Order, including information about the number of audited issues.

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Access Control: Database	0	0 / 9	0	10 / 23	10 / 32
Axis 2 Misconfiguration: Debug Information	0	0	0 / 6	0	0 / 6
Code Correctness: Byte Array to String Conversion	0	0	0	0 / 2	0 / 2
Code Correctness: Constructor Invokes Overridable Function	0	0	0	0 / 50	0 / 50
Code Correctness: Erroneous Class Compare	0	0	0	1 / 1	1 / 1
Code Correctness: Erroneous String Compare	0	0	0	0 / 4	0 / 4
Code Correctness: Multiple Stream Commits	0	0	0	0 / 3	0 / 3
Command Injection	0 / 3	0	0	0	0 / 3
Cookie Security: Cookie not Sent Over SSL	0	0	0	1 / 4	1 / 4
Cookie Security: HTTPOnly not Set	0	0	0	0 / 4	0 / 4
Cross-Site Request Forgery	0	0	0	0 / 27	0 / 27
Cross-Site Scripting: Persistent	17 / 38	0	0	0	17 / 38
Cross-Site Scripting: Reflected	19 / 70	0	0	0	19 / 70
Dead Code: Expression is Always false	0	0	0	0 / 3	0 / 3
Dead Code: Expression is Always true	0	0	0	0 / 1	0 / 1
Dead Code: Unused Method	0	0	0	0 / 2	0 / 2
Denial of Service	0	0	0	0 / 7	0 / 7
Denial of Service: Parse Double	0	0	0	0 / 1	0 / 1
Denial of Service: Regular Expression	0	4 / 4	0	0	4 / 4
Denial of Service: StringBuilder	0	0	0	1 / 32	1 / 32
Dynamic Code Evaluation: Code Injection	1 / 1	0	0	0	1 / 1
File Disclosure: J2EE	0	0 / 1	0	0	0 / 1
Header Manipulation	0	0 / 2	0	0	0 / 2
Hidden Field	0	0	0	0 / 15	0 / 15
Insecure Randomness	0	0 / 1	0	0	0 / 1
J2EE Bad Practices: getConnection()	0	0	0	0 / 5	0 / 5
J2EE Bad Practices: Leftover Debug Code	0	0	0	0 / 4	0 / 4
J2EE Bad Practices: Non-Serializable Object Stored in Session	0	0 / 3	0	0	0 / 3
J2EE Bad Practices: Sockets	0	0	0	0 / 1	0 / 1
J2EE Bad Practices: Threads	0	0	0	0 / 6	0 / 6
J2EE Misconfiguration: Excessive Servlet Mappings	0	0	0	0 / 1	0 / 1
J2EE Misconfiguration: Excessive Session Timeout	0	0	0	0 / 1	0 / 1
J2EE Misconfiguration: Missing Data Transport Constraint	0	0	0	0 / 2	0 / 2
J2EE Misconfiguration: Missing Error Handling	0	0	0	0 / 3	0 / 3
J2EE Misconfiguration: Missing Servlet Mapping	0	0	0	0 / 1	0 / 1
Key Management: Hardcoded Encryption Key	0 / 3	0	0	0	0 / 3
Log Forging	0	1 / 12	0	0	1 / 12
Missing Check against Null	0	0	0	3 / 4	3 / 4
Missing Check for Null Parameter	0	0	0	0 / 2	0 / 2
Null Dereference	0	0 / 8	0	0	0 / 8
Object Model Violation: Just one of equals() and hashCode() Defined	0	0	0	0 / 2	0 / 2
Open Redirect	0 / 5	0	0	0	0 / 5
Password Management: Empty Password	0	0 / 2	0	0	0 / 2
Password Management: Hardcoded Password	0 / 16	0 / 4	0	0	0 / 20
Password Management: Password in Comment	0	0	0	0 / 28	0 / 28

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Password Management: Password in Configuration File	0	0 / 2	0	0	0 / 2
Path Manipulation	0 / 5	0	0	0	0 / 5
Poor Error Handling: Empty Catch Block	0	0	0	0 / 32	0 / 32
Poor Error Handling: Overly Broad Catch	0	0	0	0 / 149	0 / 149
Poor Error Handling: Overly Broad Throws	0	0	0	0 / 15	0 / 15
Poor Error Handling: Throw Inside Finally	0	0	0	0 / 1	0 / 1
Poor Logging Practice: Use of a System Output Stream	0	0	0	0 / 137	0 / 137
Poor Style: Confusing Naming	0	0	0	0 / 1	0 / 1
Poor Style: Non-final Public Static Field	0	0	0	0 / 6	0 / 6
Poor Style: Redundant Initialization	0	0	0	0 / 4	0 / 4
Poor Style: Value Never Read	0	0	0	0 / 4	0 / 4
Portability Flaw: Locale Dependent Comparison	0	0 / 11	0	0	0 / 11
Privacy Violation	13 / 75	0 / 3	0	0	13 / 78
Race Condition: Singleton Member Field	0	0 / 1	0	0	0 / 1
Race Condition: Static Database Connection	0	0 / 2	0	0	0 / 2
Redundant Null Check	0	0	0	1 / 3	1 / 3
Resource Injection	0	0	0	0 / 2	0 / 2
SQL Injection	5 / 43	0	0	0 / 19	5 / 62
System Information Leak	0	0	0	0 / 192	0 / 192
System Information Leak: External	0	0 / 3	0	0	0 / 3
System Information Leak: HTML Comment in JSP	0	0	0	0 / 3	0 / 3
System Information Leak: Incomplete Servlet Error Handling	0	0	0	0 / 3	0 / 3
System Information Leak: Internal	0	0	0	31 / 62	31 / 62
Trust Boundary Violation	0	0	0	0 / 25	0 / 25
Unchecked Return Value	0	0	0	0 / 2	0 / 2
Unreleased Resource: Database	0	0 / 13	0	0	0 / 13
Unreleased Resource: Sockets	0	0 / 3	0	0	0 / 3
Unreleased Resource: Streams	0	3 / 15	0	0	3 / 15
Unsafe Reflection	0	0	0	0 / 1	0 / 1
Weak Cryptographic Hash	0	0	0	0 / 2	0 / 2
Weak Cryptographic Hash: Insecure PBE Iteration Count	0	0	0 / 4	0	0 / 4
Weak Encryption	0 / 4	0	0	0	0 / 4
XML Entity Expansion Injection	0	0	0 / 1	0	0 / 1
XML External Entity Injection	0 / 1	0	0	0	0 / 1
XPath Injection	0 / 1	0	0	0	0 / 1

Results Outline

Access Control: Database (32 issues)

Abstract

Without proper access control, executing a SQL statement that contains a user-controlled primary key can allow an attacker to view unauthorized records.

Explanation

Database access control errors occur when:

1. Data enters a program from an untrusted source.

2. The data is used to specify the value of a primary key in a SQL query.

Example 1: The following code uses a parameterized statement, which escapes metacharacters and prevents SQL injection vulnerabilities, to construct and execute a SQL query that searches for an invoice matching the specified identifier [1]. The identifier is selected from a list of all invoices associated with the current authenticated user.

```
...
id = Integer.decode(request.getParameter("invoiceID"));
String query = "SELECT * FROM invoices WHERE id = ?";
PreparedStatement stmt = conn.prepareStatement(query);
stmt.setInt(1, id);
ResultSet results = stmt.execute();
...
```

The problem is that the developer has failed to consider all of the possible values of `id`. Although the interface generates a list of invoice identifiers that belong to the current user, an attacker may bypass this interface to request any desired invoice. Because the code in this example does not check to ensure that the user has permission to access the requested invoice, it will display any invoice, even if it does not belong to the current user.

Some think that in the mobile world, classic web application vulnerabilities, such as database access control errors, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 2: The following code adapts Example 1 to the Android platform.

```
...
        String id = this.getIntent().getExtras().getString("invoiceID");
        String query = "SELECT * FROM invoices WHERE id = ?";
        SQLiteDatabase db = this.openOrCreateDatabase("DB", MODE_PRIVATE,
null);
        Cursor c = db.rawQuery(query, new Object[]{id});
...
```

A number of modern web frameworks provide mechanisms for performing validation of user input. Struts and Spring MVC are among them. To highlight the unvalidated sources of input, the rulepacks dynamically re-prioritize the

issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

Recommendation

Rather than relying on the presentation layer to restrict values submitted by the user, access control should be handled by the application and database layers. Under no circumstances should a user be allowed to retrieve or modify a row in the database without the appropriate permissions. Every query that accesses the database should enforce this policy, which can often be accomplished by simply including the current authenticated username as part of the query.

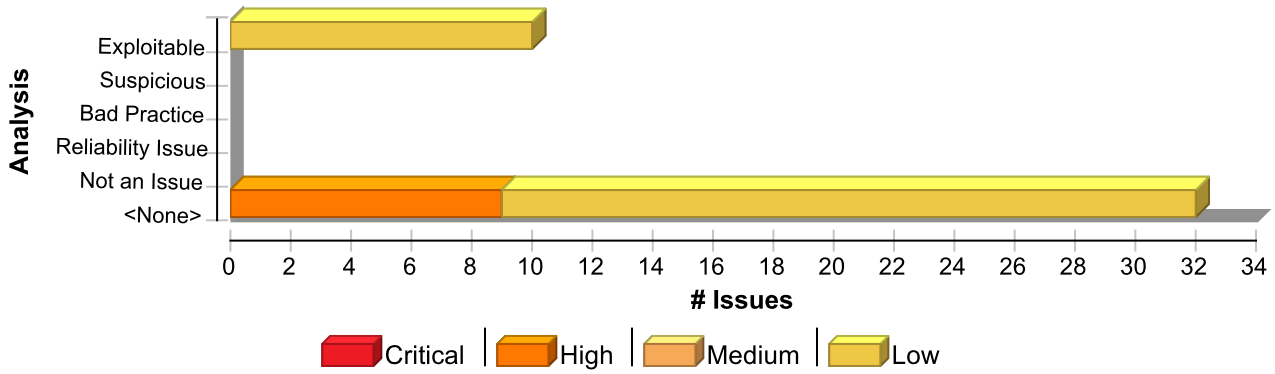
Example 3: The following code implements the same functionality as Example 1 but imposes an additional constraint requiring that the current authenticated user have specific access to the invoice.

```
...
userName = ctx.getAuthenticatedUserName();
id = Integer.decode(request.getParameter("invoiceID"));
String query =
        "SELECT * FROM invoices WHERE id = ? AND user = ?";
PreparedStatement stmt = conn.prepareStatement(query);
stmt.setInt(1, id);
stmt.setString(2, userName);
ResultSet results = stmt.execute();
...
```

And here is an Android equivalent:

```
...
        PasswordAuthentication pa = authenticator.getPasswordAuthentication();
        String userName = pa.getUserName();
        String id = this.getIntent().getExtras().getString("invoiceID");
        String query = "SELECT * FROM invoices WHERE id = ? AND user = ?";
        SQLiteDatabase db = this.openOrCreateDatabase("DB", MODE_PRIVATE,
null);
        Cursor c = db.rawQuery(query, new Object[]{id, userName});
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Access Control: Database	32	32	0	64
Total	**32**	**32**	**0**	**64**

Access Control: Database — High

Package: org.owasp.webgoat.lessons

StoredXss.java, line 223 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: makeCurrent()
File: StoredXss.java:223
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
220 ResultSet.TYPE_SCROLL_INSENSITIVE,
221 ResultSet.CONCUR_READ_ONLY);
222 statement.setString(1, getNameroot(s.getUserName()) + "%");
223 statement.setInt(2, messageNum);
224 ResultSet results = statement.executeQuery();
225
226 if ((results != null) && results.first())
```

DefaultLessonAction.java, line 313 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Access Control: Database — High

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 313 (Access Control: Database) — High

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: isAuthorizedForEmployee()
File: DefaultLessonAction.java:313
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
310 PreparedStatement answer_statement = WebSession.getConnection(s).prepareStatement( query,
311 ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
312 answer_statement.setInt(1, userId);
313 answer_statement.setInt(2, employeeId);
314 ResultSet answer_results = answer_statement.executeQuery();
315 authorized = answer_results.first();
316 }
```

CSRF.java, line 240 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

Access Control: Database — High

Package: org.owasp.webgoat.lessons

CSRF.java, line 240 (Access Control: Database) — High

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: makeCurrent()
File: CSRF.java:240
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
237  String query = "SELECT * FROM messages WHERE user_name LIKE ? and num = ?";
238  PreparedStatement statement = connection.prepareStatement( query, ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
239  statement.setString(1, getNameroot( s.getUserName() ) + "%");
240  statement.setInt(2, messageNum);
241  ResultSet results = statement.executeQuery();
242
243  if ( ( results != null ) && results.first() )
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 112 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
```

Access Control: Database — High

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 112 (Access Control: Database) — High

```
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:112
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
109  Statement answer_statement = WebSession.getConnection(s)
110  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
111  ResultSet.CONCUR_READ_ONLY);
112  ResultSet answer_results = answer_statement.executeQuery(query);
113  if (answer_results.next())
114  {
115  // Note: Do NOT get the password field.
```

UpdateProfile.java, line 248 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: NUMBER, PRIMARY_KEY, WEB

Access Control: Database — High

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (Access Control: Database) — High

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

EditProfile.java, line 96 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:96
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
93  .getConnection(s).prepareStatement(query,
94  ResultSet.TYPE_SCROLL_INSENSITIVE,
95  ResultSet.CONCUR_READ_ONLY);
96  answer_statement.setInt(1, subjectUserId);
97  ResultSet answer_results = answer_statement.executeQuery();
98  if (answer_results.next())
99  {
```

Access Control: Database — High

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

EditProfile.java, line 96 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:96
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
93 .getConnection(s).prepareStatement(query,
94 ResultSet.TYPE_SCROLL_INSENSITIVE,
95 ResultSet.CONCUR_READ_ONLY);
96 answer_statement.setInt(1, subjectUserId);
97 ResultSet answer_results = answer_statement.executeQuery();
98 if (answer_results.next())
99 {
```

ViewProfile.java, line 132 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Access Control: Database — High

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

ViewProfile.java, line 132 (Access Control: Database) — High

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:132
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
129  Statement answer_statement = WebSession.getConnection(s)
130  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
131  ResultSet.CONCUR_READ_ONLY);
132  ResultSet answer_results = answer_statement.executeQuery(query);
133  if (answer_results.next())
134  {
135  // Note: Do NOT get the password field.
```

DeleteProfile.java, line 115 (Access Control: Database) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
```

Access Control: Database — High

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 115 (Access Control: Database) — High

```
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: deleteEmployeeProfile()
File: DeleteProfile.java:115
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
112  Statement statement = WebSession.getConnection(s)
113  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
114  ResultSet.CONCUR_READ_ONLY);
115  statement.executeUpdate(query);
116  }
117  catch (SQLException sqle)
118  {
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 281 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getLoginCount(0)
From: org.owasp.webgoat.lessons.WSDLScanning.getLoginCount
File: JavaSource/org/owasp/webgoat/lessons/WSDLScanning.java:333

```
330  }
331
332
333  public String getLoginCount(int id)
334  {
335  String result = getResults(id, "login_count");
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 281 (Access Control: Database) — Low

```
336  if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: WSDLScanning.java:281
Taint Flags: NUMBER, WEBSERVICE

```
278  }
279  PreparedStatement ps = connection
280  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
281  ps.setInt(1, id);
282  try
283  {
284  ResultSet results = ps.executeQuery();
```

SoapRequest.java, line 419 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getCreditCard(0)
From: org.owasp.webgoat.lessons.SoapRequest.getCreditCard
File: JavaSource/org/owasp/webgoat/lessons/SoapRequest.java:437

```
434  }
435
436
437  public String getCreditCard(int id)
438  {
439  String result = getResults(id, "cc_number");
440  //SoapRequest.completed = true;
```

Sink Details

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 419 (Access Control: Database) — Low

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: SoapRequest.java:419
Taint Flags: NUMBER, WEBSERVICE

```
416  }
417  PreparedStatement ps = connection
418  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
419  ps.setInt(1, id);
420  try
421  {
422  ResultSet results = ps.executeQuery();
```

WSDLScanning.java, line 281 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getCreditCard(0)
From: org.owasp.webgoat.lessons.WSDLScanning.getCreditCard
File: JavaSource/org/owasp/webgoat/lessons/WSDLScanning.java:299

```
296  }
297
298
299  public String getCreditCard(int id)
300  {
301  String result = getResults(id, "cc_number");
302  if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: WSDLScanning.java:281
Taint Flags: NUMBER, WEBSERVICE

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 281 (Access Control: Database) — Low

```
278  }
279  PreparedStatement ps = connection
280  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
281  ps.setInt(1, id);
282  try
283  {
284  ResultSet results = ps.executeQuery();
```

WSDLScanning.java, line 281 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getFirstName(0)
From: org.owasp.webgoat.lessons.WSDLScanning.getFirstName
File: JavaSource/org/owasp/webgoat/lessons/WSDLScanning.java:311

```
308  }
309
310
311  public String getFirstName(int id)
312  {
313  String result = getResults(id, "first_name");
314  if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: WSDLScanning.java:281
Taint Flags: NUMBER, WEBSERVICE

```
278  }
279  PreparedStatement ps = connection
280  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 281 (Access Control: Database) — Low

```
281  ps.setInt(1, id);
282  try
283  {
284  ResultSet results = ps.executeQuery();
```

WSDLScanning.java, line 281 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getLastName(0)
From: org.owasp.webgoat.lessons.WSDLScanning.getLastName
File: JavaSource/org/owasp/webgoat/lessons/WSDLScanning.java:322

```
319  }
320
321
322  public String getLastName(int id)
323  {
324  String result = getResults(id, "last_name");
325  if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: WSDLScanning.java:281
Taint Flags: NUMBER, WEBSERVICE

```
278  }
279  PreparedStatement ps = connection
280  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
281  ps.setInt(1, id);
282  try
283  {
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 281 (Access Control: Database) — Low

```
284  ResultSet results = ps.executeQuery();
```

CSRF.java, line 96 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: addMessage()
File: CSRF.java:96
Taint Flags: WEB, XSS

```
93  PreparedStatement statement = connection.prepareStatement( query, ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
94  statement.setInt(1, count++);
95  statement.setString(2, title);
96  statement.setString(3, message);
97  statement.setString(4, s.getUserName());
98  statement.executeQuery();
99  }
```

SoapRequest.java, line 419 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 419 (Access Control: Database) — Low

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getLastName(0)
From: org.owasp.webgoat.lessons.SoapRequest.getLastName
File: JavaSource/org/owasp/webgoat/lessons/SoapRequest.java:467

```
464  }
465
466
467  public String getLastName(int id)
468  {
469  String result = getResults(id, "last_name");
470  if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: SoapRequest.java:419
Taint Flags: NUMBER, WEBSERVICE

```
416  }
417  PreparedStatement ps = connection
418  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
419  ps.setInt(1, id);
420  try
421  {
422  ResultSet results = ps.executeQuery();
```

SoapRequest.java, line 419 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 419 (Access Control: Database) — Low

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getLoginCount(0)
From: org.owasp.webgoat.lessons.SoapRequest.getLoginCount
File: JavaSource/org/owasp/webgoat/lessons/SoapRequest.java:481

```
478  }
479
480
481  public String getLoginCount(int id)
482  {
483  String result = getResults(id, "login_count");
484  if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: SoapRequest.java:419
Taint Flags: NUMBER, WEBSERVICE

```
416  }
417  PreparedStatement ps = connection
418  .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
419  ps.setInt(1, id);
420  try
421  {
422  ResultSet results = ps.executeQuery();
```

SqlStringInjection.java, line 193 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 193 (Access Control: Database) — Low

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: parameterizedQuery()
File: SqlStringInjection.java:193
Taint Flags: WEB, XSS

```
190  PreparedStatement statement = connection.prepareStatement(
191  query, ResultSet.TYPE_SCROLL_INSENSITIVE,
192  ResultSet.CONCUR_READ_ONLY);
193  statement.setString(1, accountName);
194  ResultSet results = statement.executeQuery();
195
196  if ((results != null) && (results.first() == true))
```

SqlNumericInjection.java, line 207 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 207 (Access Control: Database) — Low

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: parameterizedQuery()
File: SqlNumericInjection.java:207
Taint Flags: NUMBER, WEB

```
204 PreparedStatement statement = connection.prepareStatement(
205 query, ResultSet.TYPE_SCROLL_INSENSITIVE,
206 ResultSet.CONCUR_READ_ONLY);
207 statement.setInt(1, Integer.parseInt(station));
208 ResultSet results = statement.executeQuery();
209
210 if ((results != null) && (results.first() == true))
```

StoredXss.java, line 105 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: addMessage()
File: StoredXss.java:105
Taint Flags: WEB, XSS

```
102 ResultSet.TYPE_SCROLL_INSENSITIVE,
103 ResultSet.CONCUR_READ_ONLY);
104 statement.setInt(1, count++);
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

StoredXss.java, line 105 (Access Control: Database) — Low

```
105 statement.setString(2, title);
106 statement.setString(3, message);
107 statement.setString(4, s.getUserName());
108 statement.executeQuery();
```

SoapRequest.java, line 419 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: getFirstName(0)
From: org.owasp.webgoat.lessons.SoapRequest.getFirstName
File: JavaSource/org/owasp/webgoat/lessons/SoapRequest.java:453

```
450 }
451
452
453 public String getFirstName(int id)
454 {
455 String result = getResults(id, "first_name");
456 if (result != null)
```

Sink Details

Sink: java.sql.PreparedStatement.setInt()
Enclosing Method: getResults()
File: SoapRequest.java:419
Taint Flags: NUMBER, WEBSERVICE

```
416 }
417 PreparedStatement ps = connection
418 .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
419 ps.setInt(1, id);
420 try
421 {
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 419 (Access Control: Database) — Low

```
422  ResultSet results = ps.executeQuery();
```

CSRF.java, line 95 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: addMessage()
File: CSRF.java:95
Taint Flags: WEB, XSS

```
92
93  PreparedStatement statement = connection.prepareStatement( query, ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
94  statement.setInt(1, count++);
95  statement.setString(2, title);
96  statement.setString(3, message);
97  statement.setString(4, s.getUserName());
98  statement.executeQuery();
```

StoredXss.java, line 106 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Access Control: Database — Low

Package: org.owasp.webgoat.lessons

StoredXss.java, line 106 (Access Control: Database) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: addMessage()
File: StoredXss.java:106
Taint Flags: WEB, XSS

```
103 ResultSet.CONCUR_READ_ONLY);
104 statement.setInt(1, count++);
105 statement.setString(2, title);
106 statement.setString(3, message);
107 statement.setString(4, s.getUserName());
108 statement.executeQuery();
109 }
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (Access Control: Database) — Low

le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:14
5

```
142  .getParameter(CrossSiteScripting.SALARY));
143  String ccn = request.getParameter(CrossSiteScripting.CCN);
144  int ccnLimit = Integer.parseInt(request
145  .getParameter(CrossSiteScripting.CCN_LIMIT));
146  String disciplinaryActionDate = request
147  .getParameter(CrossSiteScripting.DISCIPLINARY_DATE);
148  String disciplinaryActionNotes = request
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: NUMBER, WEB

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 340 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
9

```
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (Access Control: Database) — Low

```
139  .getParameter(CrossSiteScripting.MANAGER));
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141  int salary = Integer.parseInt(request
142  .getParameter(CrossSiteScripting.SALARY));
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: NUMBER, WEB

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 248 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:14
5

```
142  .getParameter(CrossSiteScripting.SALARY));
143  String ccn = request.getParameter(CrossSiteScripting.CCN);
144  int ccnLimit = Integer.parseInt(request
145  .getParameter(CrossSiteScripting.CCN_LIMIT));
146  String disciplinaryActionDate = request
147  .getParameter(CrossSiteScripting.DISCIPLINARY_DATE);
148  String disciplinaryActionNotes = request
```

Sink Details

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (Access Control: Database) — Low

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: NUMBER, WEB

```
245 Statement answer_statement = WebSession.getConnection(s)
246 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247 ResultSet.CONCUR_READ_ONLY);
248 answer_statement.executeUpdate(query);
249 }
250 catch (SQLException sqle)
251 {
```

UpdateProfile.java, line 340 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:14
2

```
139 .getParameter(CrossSiteScripting.MANAGER));
140 String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141 int salary = Integer.parseInt(request
142 .getParameter(CrossSiteScripting.SALARY));
143 String ccn = request.getParameter(CrossSiteScripting.CCN);
144 int ccnLimit = Integer.parseInt(request
145 .getParameter(CrossSiteScripting.CCN_LIMIT));
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: NUMBER, WEB

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
```

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (Access Control: Database) — Low

```
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

FindProfile.java, line 176 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:176
Taint Flags: WEB, XSS

```
173 .getConnection(s).prepareStatement(query,
174 ResultSet.TYPE_SCROLL_INSENSITIVE,
175 ResultSet.CONCUR_READ_ONLY);
176 answer_statement.setString(1, "%" + pattern + "%");
177 answer_statement.setString(2, "%" + pattern + "%");
178 ResultSet answer_results = answer_statement.executeQuery();
179
```

UpdateProfile.java, line 248 (Access Control: Database) — Low

Issue Details

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (Access Control: Database) — Low

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
9

```
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
139  .getParameter(CrossSiteScripting.MANAGER));
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141  int salary = Integer.parseInt(request
142  .getParameter(CrossSiteScripting.SALARY));
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: NUMBER, WEB

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

FindProfile.java, line 177 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

FindProfile.java, line 177 (Access Control: Database) — Low

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:177
Taint Flags: WEB, XSS

```
174  ResultSet.TYPE_SCROLL_INSENSITIVE,
175  ResultSet.CONCUR_READ_ONLY);
176  answer_statement.setString(1, "%" + pattern + "%");
177  answer_statement.setString(2, "%" + pattern + "%");
178  ResultSet answer_results = answer_statement.executeQuery();
179
180  // Just use the first hit.
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 145 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter

Access Control: Database — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 145 (Access Control: Database) — Low

File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:145
Taint Flags: WEB, XSS

```
142  ResultSet.TYPE_SCROLL_INSENSITIVE,
143  ResultSet.CONCUR_READ_ONLY);
144  answer_statement.setString(1, "%" + pattern + "%");
145  answer_statement.setString(2, "%" + pattern + "%");
146  ResultSet answer_results = answer_statement.executeQuery();
147
148  // Just use the first hit.
```

FindProfile.java, line 144 (Access Control: Database) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
```

Access Control: Database	Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 144 (Access Control: Database)	Low

```
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.PreparedStatement.setString()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:144
Taint Flags: WEB, XSS

```
141  .getConnection(s).prepareStatement(query,
142  ResultSet.TYPE_SCROLL_INSENSITIVE,
143  ResultSet.CONCUR_READ_ONLY);
144  answer_statement.setString(1, "%" + pattern + "%");
145  answer_statement.setString(2, "%" + pattern + "%");
146  ResultSet answer_results = answer_statement.executeQuery();
147
```

Axis 2 Misconfiguration: Debug Information (6 issues)

Abstract

The SOAP Monitor module allows attackers to sniff SOAP traffic.

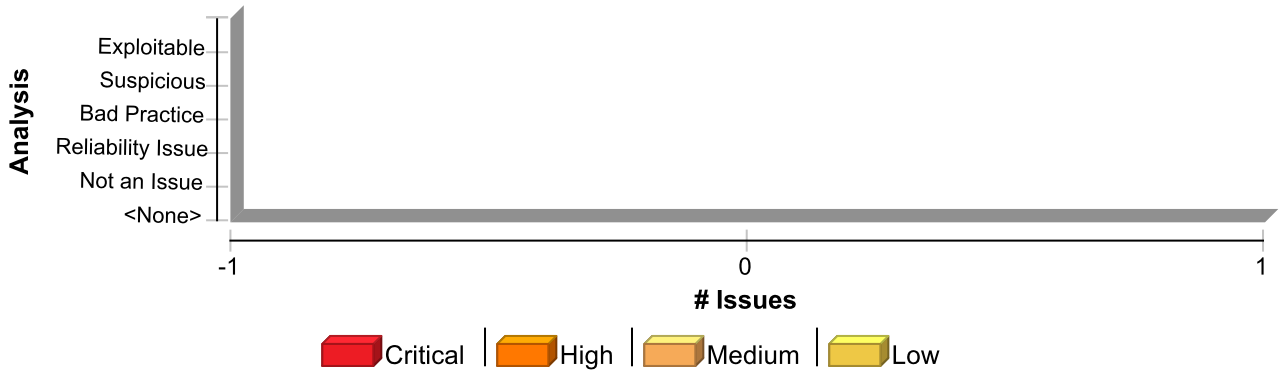
Explanation

Apache Axis 2 provides developers with a utility to monitor incoming and outgoing SOAP messages through a Java applet. The SOAP Monitor will show all SOAP messages used to invoke a Web Service. Attackers may use the utility to eavesdrop on traffic between a Web Service and its clients.

Recommendation

Disable the SOAP Monitor by removing references to it from your configuration files. Specifically, ensure that `web.xml` does not contain a servlet or servlet-mapping to the SOAP Monitor service. Also make sure that `axis2.xml` has no references such as `<module ref="soapmonitor"/>`

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Axis 2 Misconfiguration: Debug Information	6	6	0	12
Total	**6**	**6**	**0**	**12**

Axis 2 Misconfiguration: Debug Information	Medium
Package: WebContent.WEB-INF	
web.xml, line 89 (Axis 2 Misconfiguration: Debug Information)	**Medium**

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:89

Axis 2 Misconfiguration: Debug Information — Medium

Package: WebContent.WEB-INF

web.xml, line 89 (Axis 2 Misconfiguration: Debug Information) — Medium

```
86 </servlet>
87
88 <servlet>
89 <servlet-name>SOAPMonitorService</servlet-name>
90 <display-name>SOAPMonitorService</display-name>
91 <servlet-class>
92 org.apache.axis.monitor.SOAPMonitorService
```

web.xml, line 90 (Axis 2 Misconfiguration: Debug Information) — Medium

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:90

```
87
88 <servlet>
89 <servlet-name>SOAPMonitorService</servlet-name>
90 <display-name>SOAPMonitorService</display-name>
91 <servlet-class>
92 org.apache.axis.monitor.SOAPMonitorService
93 </servlet-class>
```

web.xml, line 91 (Axis 2 Misconfiguration: Debug Information) — Medium

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:91

```
88 <servlet>
89 <servlet-name>SOAPMonitorService</servlet-name>
90 <display-name>SOAPMonitorService</display-name>
91 <servlet-class>
92 org.apache.axis.monitor.SOAPMonitorService
93 </servlet-class>
94 <init-param>
```

Axis 2 Misconfiguration: Debug Information — Medium

Package: WebContent.WEB-INF

web.xml, line 91 (Axis 2 Misconfiguration: Debug Information) — Medium

web.xml, line 95 (Axis 2 Misconfiguration: Debug Information) — Medium

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:95

```
92 org.apache.axis.monitor.SOAPMonitorService
93 </servlet-class>
94 <init-param>
95 <param-name>SOAPMonitorPort</param-name>
96 <param-value>5001</param-value>
97 </init-param>
98 <load-on-startup>100</load-on-startup>
```

web.xml, line 216 (Axis 2 Misconfiguration: Debug Information) — Medium

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:216

```
213 </servlet-mapping>
214
215 <servlet-mapping>
216 <servlet-name>SOAPMonitorService</servlet-name>
217 <url-pattern>/SOAPMonitor</url-pattern>
218 </servlet-mapping>
219
```

web.xml, line 217 (Axis 2 Misconfiguration: Debug Information) — Medium

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Axis 2 Misconfiguration: Debug Information — Medium

Package: WebContent.WEB-INF

web.xml, line 217 (Axis 2 Misconfiguration: Debug Information) — Medium

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: web.xml:217

```
214
215 <servlet-mapping>
216 <servlet-name>SOAPMonitorService</servlet-name>
217 <url-pattern>/SOAPMonitor</url-pattern>
218 </servlet-mapping>
219
220 <!-- uncomment this if you want the admin servlet -->
```

Code Correctness: Byte Array to String Conversion (2 issues)

Abstract

Converting a byte array into a `String` may lead to data loss.

Explanation

When data from a byte array is converted into a `String`, it is unspecified what will happen to any data that is outside of the applicable character set. This can lead to data being lost, or a decrease in the level of security when binary data is needed to ensure proper security measures are followed.

Example 1: The following code converts data into a String in order to create a hash.

```
...
FileInputStream fis = new FileInputStream(myFile);
byte[] byteArr = byte[BUFSIZE];
...
int count = fis.read(byteArr);
...
String fileString = new String(byteArr);
String fileSHA256Hex = DigestUtils.sha256Hex(fileString);
// use fileSHA256Hex to validate file
...
```

Assuming the size of the file is less than `BUFSIZE`, this works fine as long as the information in `myFile` is encoded the same as the default character set, however if it's using a different encoding, or is a binary file, it will lose information. This in turn will cause the resulting SHA hash to be less reliable, and could mean it's far easier to cause collisions, especially if any data outside of the default character set is represented by the same value, such as a question mark.

Recommendation

Generally speaking, a byte array potentially containing noncharacter data should never be converted into a `String` object as it may break functionality, but in some cases this can cause much larger security concerns. In a lot of cases there is no need to actually convert a byte array into a String, but if there is a specific reason to be able to create a `String` object from binary data, it must first be encoded in a way such that it will fit into the default character set.

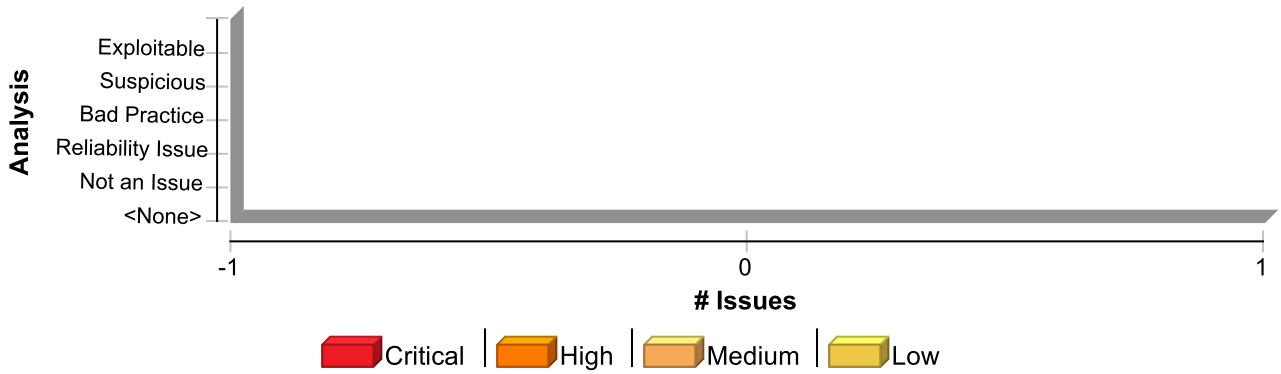
Example 2: The following uses a different variant of the API in Example 1 to prevent any validation problems.

```
...
FileInputStream fis = new FileInputStream(myFile);
byte[] byteArr = byte[BUFSIZE];
...
int count = fis.read(byteArr);
...
byte[] fileSHA256 = DigestUtils.sha256(byteArr);
// use fileSHA256 to validate file, comparing hash byte-by-byte.
...
```

In this case it is straightforward to rectify, since this API has overloaded variants including one that accepts a byte array, and this could be simplified even further by using another overloaded variant of `DigestUtils.sha256()` that accepts a `FileInputStream` object as its argument. Other scenarios may need careful consideration as to

whether it's possible that the byte array could contain data outside of the character set, and further refactoring may be required.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Byte Array to String Conversion	2	2	0	4
Total	**2**	**2**	**0**	**4**

Code Correctness: Byte Array to String Conversion — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 911 (Code Correctness: Byte Array to String Conversion) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: String()
Enclosing Method: unicodeEncode()
File: Encoding.java:911

```
908 Charset charset = Charset.forName( "ISO-8859-1" );
909 CharsetEncoder encoder = charset.newEncoder();
910 ByteBuffer bbuf = encoder.encode( CharBuffer.wrap( str ) );
911 return ( new String( bbuf.array() ) );
912 }
913 catch ( Exception e )
914 {
```

Encoding.java, line 266 (Code Correctness: Byte Array to String Conversion) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Semantic)

Code Correctness: Byte Array to String Conversion	Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 266 (Code Correctness: Byte Array to String Conversion)	Low

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: String()
Enclosing Method: base64Decode()
File: Encoding.java:266

```
263
264  byte[] b = decoder.decodeBuffer( str );
265
266  return ( new String( b ) );
267  }
268
269
```

Code Correctness: Constructor Invokes Overridable Function (50 issues)

Abstract

A constructor of the class calls a function that can be overridden.

Explanation

When a constructor calls an overridable function, it may allow an attacker to access the `this` reference prior to the object being fully initialized, which can in turn lead to a vulnerability.

Example 1: The following calls a method that can be overridden.

```
...
class User {
  private String username;
  private boolean valid;
  public User(String username, String password){
    this.username = username;
    this.valid = validateUser(username, password);
  }
  public boolean validateUser(String username, String password){
    //validate user is real and can authenticate
    ...
  }
  public final boolean isValid(){
    return valid;
  }
}
```

Since the function `validateUser` and the class are not `final`, it means that they can be overridden, and then initializing a variable to the subclass that overrides this function would allow bypassing of the `validateUser` functionality. For example:

```
...
class Attacker extends User{
  public Attacker(String username, String password){
    super(username, password);
  }
  public boolean validateUser(String username, String password){
    return true;
  }
}
...
class MainClass{
  public static void main(String[] args){
    User hacker = new Attacker("Evil", "Hacker");
    if (hacker.isValid()){
      System.out.println("Attack successful!");
    }else{
      System.out.println("Attack failed");
    }
  }
}
```

The above code prints "Attack successful!", since the `Attacker` class overrides the `validateUser()` function that is called from the constructor of the superclass `User`, and Java will first look in the subclass for functions called from the constructor.

Recommendation

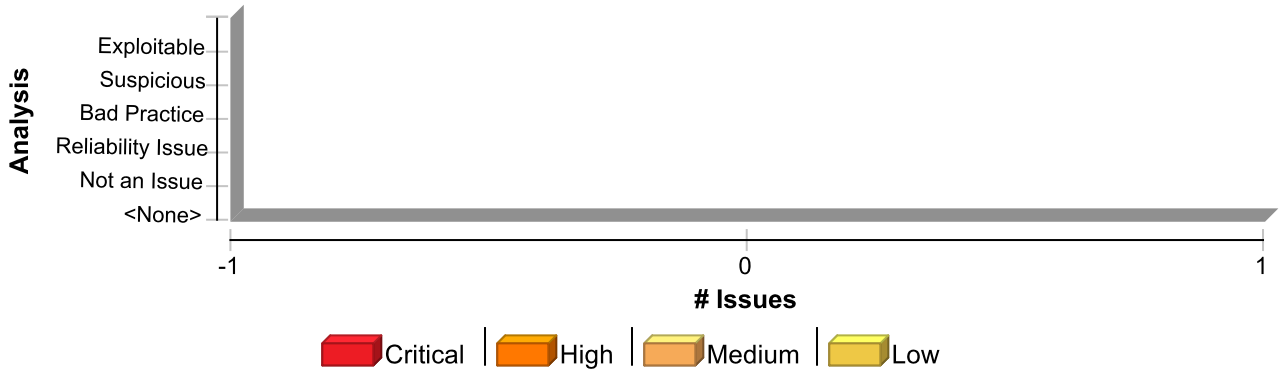
Constructors should not call functions that can be overridden, either by specifying them as `final`, or specifying the class as `final`. Alternatively if this code is only ever needed in the constructor, the `private` access specifier can be used, or the logic could be placed directly into the constructor of the superclass.

Example 2: The following makes the class `final` to prevent the function from being overridden elsewhere.

```
...
final class User {
  private String username;
  private boolean valid;
  public User(String username, String password){
    this.username = username;
    this.valid = validateUser(username, password);
  }
  private boolean validateUser(String username, String password){
    //validate user is real and can authenticate
    ...
  }
  public final boolean isValid(){
    return valid;
  }
}
```

This example specifies the class as `final`, so that it cannot be subclassed, and changes the `validateUser()` function to `private`, since it is not needed elsewhere in this application. This is programming defensively, since at a later date it may be decided that the `User` class needs to be subclassed, which would result in this vulnerability reappearing if the `validateUser()` function was not set to `private`.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Constructor Invokes Overridable Function	50	50	0	100
Total	**50**	**50**	**0**	**100**

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons

WelcomeScreen.java, line 56 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: setup
Enclosing Method: WelcomeScreen()
File: WelcomeScreen.java:56

```
53 */
54 public WelcomeScreen(WebSession s)
55 {
56 setup(s);
57 }
58
59
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:165

```
162
163 // These actions are special in that they chain to other actions.
164 registerAction(new Login(this, myClassName, LOGIN_ACTION,
165 getAction(LISTSTAFF_ACTION)));
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
166 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
167 getAction(LOGIN_ACTION)));
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
```

CrossSiteScripting.java, line 173 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:173

```
170 registerAction(new UpdateProfile(this, myClassName,
171 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
172 registerAction(new DeleteProfile(this, myClassName,
173 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
174 }
175
176
```

CrossSiteScripting.java, line 169 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:169

```
166 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
167 getAction(LOGIN_ACTION)));
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
169 getAction(VIEWPROFILE_ACTION)));
170 registerAction(new UpdateProfile(this, myClassName,
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 169 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
171 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
172 registerAction(new DeleteProfile(this, myClassName,
```

CrossSiteScripting.java, line 171 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:171

```
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
169 getAction(VIEWPROFILE_ACTION)));
170 registerAction(new UpdateProfile(this, myClassName,
171 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
172 registerAction(new DeleteProfile(this, myClassName,
173 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
174 }
```

CrossSiteScripting.java, line 167 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:167

```
164 registerAction(new Login(this, myClassName, LOGIN_ACTION,
165 getAction(LISTSTAFF_ACTION)));
166 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
167 getAction(LOGIN_ACTION)));
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
169 getAction(VIEWPROFILE_ACTION)));
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 167 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
170 registerAction(new UpdateProfile(this, myClassName,
```

CrossSiteScripting.java, line 157 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:157

```
154 public CrossSiteScripting()
155 {
156 String myClassName = parseClassName(this.getClass().getName());
157 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
158 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
159 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
160 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
```

CrossSiteScripting.java, line 158 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:158

```
155 {
156 String myClassName = parseClassName(this.getClass().getName());
157 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
158 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
159 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
160 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
161 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 158 (Code Correctness: Constructor Invokes Overridable Function)	Low

CrossSiteScripting.java, line 159 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:159

```
156 String myClassName = parseClassName(this.getClass().getName());
157 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
158 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
159 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
160 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
161 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
162
```

CrossSiteScripting.java, line 160 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:160

```
157 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
158 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
159 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
160 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
161 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
162
163 // These actions are special in that they chain to other actions.
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 161 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:161

```
158 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
159 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
160 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
161 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
162
163 // These actions are special in that they chain to other actions.
164 registerAction(new Login(this, myClassName, LOGIN_ACTION,
```

CrossSiteScripting.java, line 164 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:164

```
161 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
162
163 // These actions are special in that they chain to other actions.
164 registerAction(new Login(this, myClassName, LOGIN_ACTION,
165 getAction(LISTSTAFF_ACTION)));
166 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
167 getAction(LOGIN_ACTION)));
```

CrossSiteScripting.java, line 166 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 166 (Code Correctness: Constructor Invokes Overridable Function) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:166

```
163 // These actions are special in that they chain to other actions.
164 registerAction(new Login(this, myClassName, LOGIN_ACTION,
165 getAction(LISTSTAFF_ACTION)));
166 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
167 getAction(LOGIN_ACTION)));
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
169 getAction(VIEWPROFILE_ACTION)));
```

CrossSiteScripting.java, line 168 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:168

```
165 getAction(LISTSTAFF_ACTION)));
166 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
167 getAction(LOGIN_ACTION)));
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
169 getAction(VIEWPROFILE_ACTION)));
170 registerAction(new UpdateProfile(this, myClassName,
171 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
```

CrossSiteScripting.java, line 170 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 170 (Code Correctness: Constructor Invokes Overridable Function) — Low

Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:170

```
167 getAction(LOGIN_ACTION)));
168 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
169 getAction(VIEWPROFILE_ACTION)));
170 registerAction(new UpdateProfile(this, myClassName,
171 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
172 registerAction(new DeleteProfile(this, myClassName,
173 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
```

CrossSiteScripting.java, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: CrossSiteScripting()
File: CrossSiteScripting.java:172

```
169 getAction(VIEWPROFILE_ACTION)));
170 registerAction(new UpdateProfile(this, myClassName,
171 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
172 registerAction(new DeleteProfile(this, myClassName,
173 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
174 }
175
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:150

```
147 public RoleBasedAccessControl()
148 {
149 String myClassName = parseClassName(this.getClass().getName());
150 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
151 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
152 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
153 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
```

RoleBasedAccessControl.java, line 151 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:151

```
148 {
149 String myClassName = parseClassName(this.getClass().getName());
150 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
151 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
152 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
153 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
154 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
```

RoleBasedAccessControl.java, line 152 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 152 (Code Correctness: Constructor Invokes Overridable Function) — Low

Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:152

```
149 String myClassName = parseClassName(this.getClass().getName());
150 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
151 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
152 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
153 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
154 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
155
```

RoleBasedAccessControl.java, line 153 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:153

```
150 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
151 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
152 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
153 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
154 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
155
156 // These actions are special in that they chain to other actions.
```

RoleBasedAccessControl.java, line 154 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 154 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:154

```
151 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
152 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
153 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
154 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
155
156 // These actions are special in that they chain to other actions.
157 registerAction(new Login(this, myClassName, LOGIN_ACTION,
```

RoleBasedAccessControl.java, line 157 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:157

```
154 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
155
156 // These actions are special in that they chain to other actions.
157 registerAction(new Login(this, myClassName, LOGIN_ACTION,
158 getAction(LISTSTAFF_ACTION)));
159 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
160 getAction(LOGIN_ACTION)));
```

RoleBasedAccessControl.java, line 159 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 159 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:159

```
156 // These actions are special in that they chain to other actions.
157 registerAction(new Login(this, myClassName, LOGIN_ACTION,
158 getAction(LISTSTAFF_ACTION)));
159 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
160 getAction(LOGIN_ACTION)));
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
162 getAction(VIEWPROFILE_ACTION)));
```

RoleBasedAccessControl.java, line 161 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:161

```
158 getAction(LISTSTAFF_ACTION)));
159 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
160 getAction(LOGIN_ACTION)));
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
162 getAction(VIEWPROFILE_ACTION)));
163 registerAction(new UpdateProfile(this, myClassName,
164 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
```

RoleBasedAccessControl.java, line 163 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 163 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:163

```
160 getAction(LOGIN_ACTION)));
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
162 getAction(VIEWPROFILE_ACTION)));
163 registerAction(new UpdateProfile(this, myClassName,
164 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
165 registerAction(new DeleteProfile(this, myClassName,
166 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
```

RoleBasedAccessControl.java, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:165

```
162 getAction(VIEWPROFILE_ACTION)));
163 registerAction(new UpdateProfile(this, myClassName,
164 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
165 registerAction(new DeleteProfile(this, myClassName,
166 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
167 }
168
```

RoleBasedAccessControl.java, line 162 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 162 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:162

```
159 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
160 getAction(LOGIN_ACTION)));
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
162 getAction(VIEWPROFILE_ACTION)));
163 registerAction(new UpdateProfile(this, myClassName,
164 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
165 registerAction(new DeleteProfile(this, myClassName,
```

RoleBasedAccessControl.java, line 164 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:164

```
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
162 getAction(VIEWPROFILE_ACTION)));
163 registerAction(new UpdateProfile(this, myClassName,
164 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
165 registerAction(new DeleteProfile(this, myClassName,
166 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
167 }
```

RoleBasedAccessControl.java, line 158 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 158 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:158

```
155
156 // These actions are special in that they chain to other actions.
157 registerAction(new Login(this, myClassName, LOGIN_ACTION,
158 getAction(LISTSTAFF_ACTION)));
159 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
160 getAction(LOGIN_ACTION)));
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
```

RoleBasedAccessControl.java, line 166 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:166

```
163 registerAction(new UpdateProfile(this, myClassName,
164 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
165 registerAction(new DeleteProfile(this, myClassName,
166 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
167 }
168
169
```

RoleBasedAccessControl.java, line 160 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 160 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: RoleBasedAccessControl()
File: RoleBasedAccessControl.java:160

```
157 registerAction(new Login(this, myClassName, LOGIN_ACTION,
158 getAction(LISTSTAFF_ACTION)));
159 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
160 getAction(LOGIN_ACTION)));
161 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
162 getAction(VIEWPROFILE_ACTION)));
163 registerAction(new UpdateProfile(this, myClassName,
```

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:172

```
169 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
170 getAction(LOGIN_ACTION)));
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
172 getAction(VIEWPROFILE_ACTION)));
173 registerAction(new UpdateProfile(this, myClassName,
174 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
175 registerAction(new DeleteProfile(this, myClassName,
```

SQLInjection.java, line 174 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 174 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:174

```
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
172 getAction(VIEWPROFILE_ACTION)));
173 registerAction(new UpdateProfile(this, myClassName,
174 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
175 registerAction(new DeleteProfile(this, myClassName,
176 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
177 }
```

SQLInjection.java, line 168 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:168

```
165
166 // These actions are special in that they chain to other actions.
167 registerAction(new Login(this, myClassName, LOGIN_ACTION,
168 getAction(LISTSTAFF_ACTION)));
169 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
170 getAction(LOGIN_ACTION)));
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
```

SQLInjection.java, line 176 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 176 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:176

```
173 registerAction(new UpdateProfile(this, myClassName,
174 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
175 registerAction(new DeleteProfile(this, myClassName,
176 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
177 }
178
179
```

SQLInjection.java, line 170 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: getAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:170

```
167 registerAction(new Login(this, myClassName, LOGIN_ACTION,
168 getAction(LISTSTAFF_ACTION)));
169 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
170 getAction(LOGIN_ACTION)));
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
172 getAction(VIEWPROFILE_ACTION)));
173 registerAction(new UpdateProfile(this, myClassName,
```

SQLInjection.java, line 160 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 160 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:160

```
157 public SQLInjection()
158 {
159 String myClassName = parseClassName(this.getClass().getName());
160 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
161 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
162 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
163 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
```

SQLInjection.java, line 161 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:161

```
158 {
159 String myClassName = parseClassName(this.getClass().getName());
160 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
161 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
162 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
163 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
164 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
```

SQLInjection.java, line 162 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 162 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:162

```
159 String myClassName = parseClassName(this.getClass().getName());
160 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
161 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
162 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
163 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
164 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
165
```

SQLInjection.java, line 163 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:163

```
160 registerAction(new ListStaff(this, myClassName, LISTSTAFF_ACTION));
161 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
162 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
163 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
164 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
165
166 // These actions are special in that they chain to other actions.
```

SQLInjection.java, line 164 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 164 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:164

```
161 registerAction(new SearchStaff(this, myClassName, SEARCHSTAFF_ACTION));
162 registerAction(new ViewProfile(this, myClassName, VIEWPROFILE_ACTION));
163 registerAction(new EditProfile(this, myClassName, EDITPROFILE_ACTION));
164 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
165
166 // These actions are special in that they chain to other actions.
167 registerAction(new Login(this, myClassName, LOGIN_ACTION,
```

SQLInjection.java, line 167 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:167

```
164 registerAction(new EditProfile(this, myClassName, CREATEPROFILE_ACTION));
165
166 // These actions are special in that they chain to other actions.
167 registerAction(new Login(this, myClassName, LOGIN_ACTION,
168 getAction(LISTSTAFF_ACTION)));
169 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
170 getAction(LOGIN_ACTION)));
```

SQLInjection.java, line 169 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 169 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:169

```
166 // These actions are special in that they chain to other actions.
167 registerAction(new Login(this, myClassName, LOGIN_ACTION,
168 getAction(LISTSTAFF_ACTION)));
169 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
170 getAction(LOGIN_ACTION)));
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
172 getAction(VIEWPROFILE_ACTION)));
```

SQLInjection.java, line 171 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:171

```
168 getAction(LISTSTAFF_ACTION)));
169 registerAction(new Logout(this, myClassName, LOGOUT_ACTION,
170 getAction(LOGIN_ACTION)));
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
172 getAction(VIEWPROFILE_ACTION)));
173 registerAction(new UpdateProfile(this, myClassName,
174 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
```

SQLInjection.java, line 173 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 173 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:173

```
170 getAction(LOGIN_ACTION)));
171 registerAction(new FindProfile(this, myClassName, FINDPROFILE_ACTION,
172 getAction(VIEWPROFILE_ACTION)));
173 registerAction(new UpdateProfile(this, myClassName,
174 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
175 registerAction(new DeleteProfile(this, myClassName,
176 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
```

SQLInjection.java, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: registerAction
Enclosing Method: SQLInjection()
File: SQLInjection.java:175

```
172 getAction(VIEWPROFILE_ACTION)));
173 registerAction(new UpdateProfile(this, myClassName,
174 UPDATEPROFILE_ACTION, getAction(VIEWPROFILE_ACTION)));
175 registerAction(new DeleteProfile(this, myClassName,
176 DELETEPROFILE_ACTION, getAction(LISTSTAFF_ACTION)));
177 }
178
```

Package: org.owasp.webgoat.session

ErrorScreen.java, line 93 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.session

ErrorScreen.java, line 93 (Code Correctness: Constructor Invokes Overridable Function) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: setup
Enclosing Method: ErrorScreen()
File: ErrorScreen.java:93

```
90 {
91 this.message = msg;
92 fixCurrentScreen( s );
93 setup( s );
94 }
95
96
```

ErrorScreen.java, line 78 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: fixCurrentScreen
Enclosing Method: ErrorScreen()
File: ErrorScreen.java:78

```
75 public ErrorScreen( WebSession s, Throwable t )
76 {
77 this.error = t;
78 fixCurrentScreen( s );
79 setup( s );
80 }
81
```

ErrorScreen.java, line 79 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: org.owasp.webgoat.session

ErrorScreen.java, line 79 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink: FunctionCall: setup
Enclosing Method: ErrorScreen()
File: ErrorScreen.java:79

```
76 {
77 this.error = t;
78 fixCurrentScreen( s );
79 setup( s );
80 }
81
82
```

ErrorScreen.java, line 92 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: fixCurrentScreen
Enclosing Method: ErrorScreen()
File: ErrorScreen.java:92

```
89 public ErrorScreen( WebSession s, String msg )
90 {
91 this.message = msg;
92 fixCurrentScreen( s );
93 setup( s );
94 }
95
```

Code Correctness: Erroneous Class Compare (1 issue)

Abstract

Determining an object's type based on its class name can lead to unexpected behavior or allow an attacker to inject a malicious class.

Explanation

Attackers may deliberately duplicate class names in order to cause a program to execute malicious code. For this reason, class names are not good type identifiers and should not be used as the basis for granting trust to a given object.

Example 1: The following code opts to trust or distrust input from an `inputReader` object based on its class name. If an attacker is able to supply an implementation of `inputReader` that executes malicious commands, this code will be unable to differentiate the benign and malicious versions of the object.

```
if (inputReader.getClass().getName().equals("TrustedName"))
{
   input = inputReader.getInput();
   ...
}
```

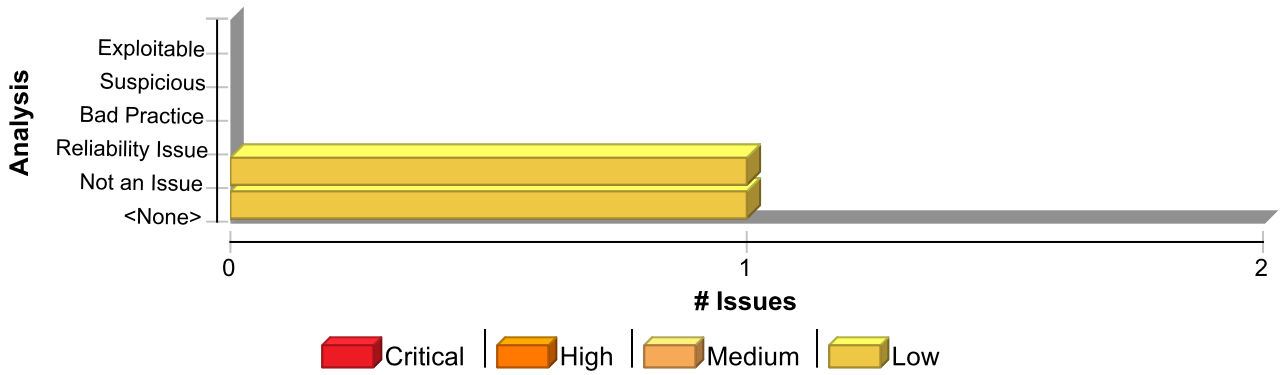
Recommendation

Always use a class-equivalence comparison to identify the type of an object. Do not rely on class names to convey type information.

Example 2: The following code has been rewritten to use a class-equivalency comparison to determine whether `inputReader` object has the expected type.

```
if (inputReader.getClass() == TrustedClass)
{
   input = inputReader.getInput();
   ...
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Erroneous Class Compare	1	1	0	2
Total	**1**	**1**	**0**	**2**

Code Correctness: Erroneous Class Compare — Low

Package: org.owasp.webgoat.session

WebSession.java, line 1197 (Code Correctness: Erroneous Class Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.lang.Class.getName()
From: org.owasp.webgoat.lessons.AbstractLesson.getName
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:223

```
220
221  public String getName()
222  {
223  String className = getClass().getName();
224  return className.substring(className.lastIndexOf('.') + 1);
225  }
226
```

Sink Details

Sink: java.lang.String.equals()
Enclosing Method: htmlEncode()
File: WebSession.java:1197
Taint Flags: CLASS_NAME

```
1194  public String htmlEncode(String s)
1195  {
1196  //System.out.println("Testing for stage 4 completion in lesson " + getCurrentLesson().getName());
1197  if (getCurrentLesson().getName().equals("CrossSiteScripting"))
1198  {
1199  if (getCurrentLesson().getStage(this) == 4 &&
1200  s.indexOf("<script>") > -1 && s.indexOf("alert") > -1 && s.indexOf("</script>") > -1)
```

Code Correctness: Erroneous String Compare (4 issues)

Abstract

Strings should be compared with the `equals()` method, not `==` or `!=`.

Explanation

This program uses `==` or `!=` to compare two strings for equality, which compares two objects for equality, not their values. Chances are good that the two references will never be equal.

Example 1: The following branch will never be taken.

```
if (args[0] == STRING_CONSTANT) {
    logger.info("miracle");
}
```

The `==` and `!=` operators will only behave as expected when they are used to compare strings contained in objects that are equal. The most common way for this to occur is for the strings to be interned, whereby the strings are added to a pool of objects maintained by the `String` class. Once a string is interned, all uses of that string will use the same object and equality operators will behave as expected. All string literals and string-valued constants are interned automatically. Other strings can be interned manually be calling `String.intern()`, which will return a canonical instance of the current string, creating one if necessary.

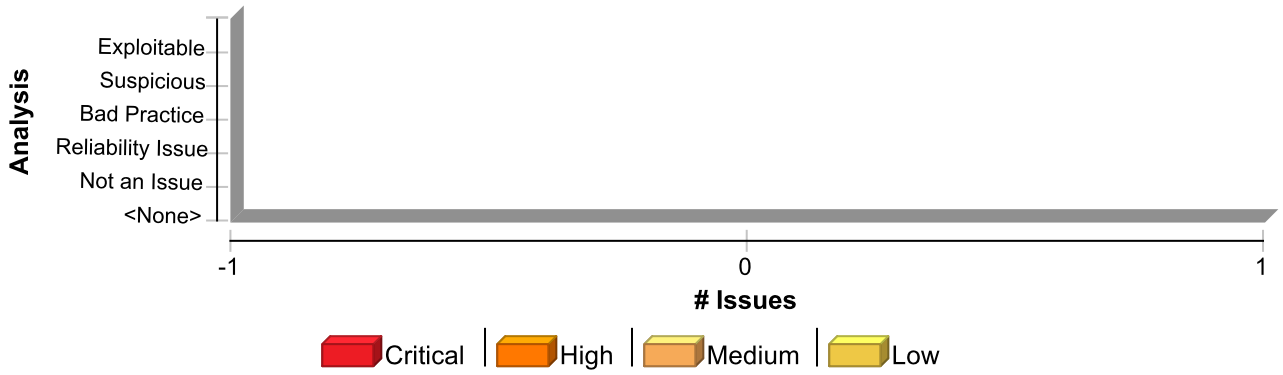
Recommendation

Use `equals()` to compare strings.

Example 2: The code in Example 1 could be rewritten in the following way:

```
if (STRING_CONSTANT.equals(args[0])) {
    logger.info("could happen");
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Erroneous String Compare	4	4	0	8
Total	**4**	**4**	**0**	**8**

Code Correctness: Erroneous String Compare — Low

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 142 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Operation
Enclosing Method: checkParams()
File: WeakAuthenticationCookie.java:142

```
139 loginID = encode("aspect12345");
140 }
141
142 if (loginID != "")
143 {
144 Cookie newCookie = new Cookie(AUTHCOOKIE, loginID);
145 s.setMessage("Your identity has been remembered");
```

XMLInjection.java, line 292 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Operation
Enclosing Method: createContent()
File: XMLInjection.java:292

```
289 for (int i = 1001; i < 1001 + rewardsMap.size(); i++)
290 {
291
292 if (s.getParser().getRawParameter("check" + i, "") != "")
293 {
294 shipment.append(((Reward) rewardsMap.get(i)).getName()
295 + "<br>");
```

XMLInjection.java, line 282 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Erroneous String Compare — Low

Package: org.owasp.webgoat.lessons

XMLInjection.java, line 282 (Code Correctness: Erroneous String Compare) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Operation
Enclosing Method: createContent()
File: XMLInjection.java:282

```
279
280 if (s.getParser().getRawParameter("SUBMIT", "") != "")
281 {
282 if (s.getParser().getRawParameter("check1004", "") != "")
283 {
284 makeSuccess(s);
285 }
```

XMLInjection.java, line 280 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Operation
Enclosing Method: createContent()
File: XMLInjection.java:280

```
277 b.setName("SUBMIT");
278 ec.addElement(b);
279
280 if (s.getParser().getRawParameter("SUBMIT", "") != "")
281 {
282 if (s.getParser().getRawParameter("check1004", "") != "")
283 {
```

Code Correctness: Multiple Stream Commits (3 issues)

Abstract

After a servlet's output stream has already been committed, it is erroneous to reset the stream buffer or perform any other action that recommits to the stream. Likewise, it is erroneous to call `getWriter()` after calling `getOutputStream` or vice versa.

Explanation

Forwarding an `HttpServletRequest`, redirecting an `HttpServletResponse`, or flushing the servlet's output stream buffer causes the associated stream to commit. Any subsequent buffer resets or stream commits, such as additional flushes or redirects, will result in `IllegalStateException`s.

Furthermore, Java servlets allow data to be written to the response stream using either `ServletOutputStream` or `PrintWriter`, but not both. Calling `getWriter()` after having called `getOutputStream()`, or vice versa, will also cause an `IllegalStateException`.

At runtime, an `IllegalStateException` prevents the response handler from running to completion, effectively dropping the response. This can cause server instability, which is a sign of an improperly implemented servlet.

Example 1: The following code redirects the servlet response after its output stream buffer has been flushed.

```
public class RedirectServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {
        ...
        OutputStream out = res.getOutputStream();
        ...
        // flushes, and thereby commits, the output stream
        out.flush();
        out.close();        // redirecting the response causes an
IllegalStateException
        res.sendRedirect("http://www.acme.com");
    }
}
```

Example 2: Conversely, the following code attempts to write to and flush the `PrintWriter`'s buffer after the request has been forwarded.

```
public class FlushServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {
        ...
        // forwards the request, implicitly committing the stream
        getServletConfig().getServletContext().getRequestDispatcher("/jsp/
boom.jsp").forward(req, res);
        ...

        // IllegalStateException; cannot redirect after forwarding
        res.sendRedirect("http://www.acme.com/jsp/boomboom.jsp");

        PrintWriter out = res.getWriter();

        // writing to an already-committed stream will not cause an exception,
        // but will not apply these changes to the final output, either
        out.print("Writing here does nothing");
```

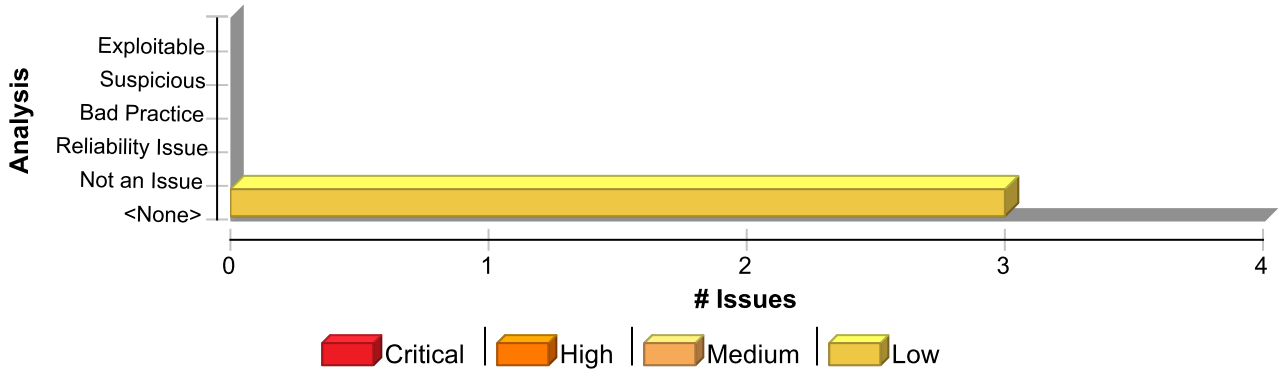
```
        // IllegalStateException; cannot flush a response's buffer after
forwarding the request
        out.flush();
        out.close();
    }
}
```

Recommendation

Ensure that no further changes to the stream are made after it is committed. At best, any further writes to the stream are ineffectual, and any additional commits can cause the servlet to throw an `IllegalStateException`. It is generally good practice to observe these guidelines when possible:

1. Return immediately after a `forward()` or `sendRedirect`. 2. Avoid calling `forward()` or `sendRedirect()` after calling `ServletResponse.getWriter()` or `ServletResponse.getOuputStream()`.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Multiple Stream Commits	3	3	0	6
Total	**3**	**3**	**0**	**6**

Code Correctness: Multiple Stream Commits	**Low**
Package: org.owasp.webgoat	
HammerHead.java, line 200 (Code Correctness: Multiple Stream Commits)	**Low**

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Sink Details

Sink: ?.writeScreen(?, response)
Enclosing Method: doPost()

Code Correctness: Multiple Stream Commits — Low

Package: org.owasp.webgoat

HammerHead.java, line 200 (Code Correctness: Multiple Stream Commits) — Low

File: HammerHead.java:200

```
197 {
198 try
199 {
200 this.writeScreen(screen, response);
201 }
202 catch (Throwable thr)
203 {
```

HammerHead.java, line 187 (Code Correctness: Multiple Stream Commits) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: forward(?, response)
Enclosing Method: doPost()
File: HammerHead.java:187

```
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
188 request, response);
189 }
190 catch (Throwable t)
```

LessonSource.java, line 100 (Code Correctness: Multiple Stream Commits) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: ?.writeSource(?, response)
Enclosing Method: doPost()
File: LessonSource.java:100

Code Correctness: Multiple Stream Commits	Low
Package: org.owasp.webgoat	
LessonSource.java, line 100 (Code Correctness: Multiple Stream Commits)	**Low**

```
97  {
98  try
99  {
100 this.writeSource(source, response);
101 }
102 catch (Throwable thr)
103 {
```

Command Injection (3 issues)

Abstract

Executing commands from an untrusted source or in an untrusted environment can cause an application to execute malicious commands on behalf of an attacker.

Explanation

Command injection vulnerabilities take two forms:

- An attacker can change the command that the program executes: the attacker explicitly controls what the command is.

- An attacker can change the environment in which the command executes: the attacker implicitly controls what the command means.

In this case we are primarily concerned with the first scenario, the possibility that an attacker may be able to control the command that is executed. Command injection vulnerabilities of this type occur when:

1. Data enters the application from an untrusted source.

2. The data is used as or as part of a string representing a command that is executed by the application.

3. By executing the command, the application gives an attacker a privilege or capability that the attacker would not otherwise have.

Example 1: The following code from a system utility uses the system property `APPHOME` to determine the directory in which it is installed and then executes an initialization script based on a relative path from the specified directory.

```
...
String home = System.getProperty("APPHOME");
String cmd = home + INITCMD;
java.lang.Runtime.getRuntime().exec(cmd);
...
```

The code in Example 1 allows an attacker to execute arbitrary commands with the elevated privilege of the application by modifying the system property `APPHOME` to point to a different path containing a malicious version of `INITCMD`. Because the program does not validate the value read from the environment, if an attacker can control the value of the system property `APPHOME`, then they can fool the application into running malicious code and take control of the system.

Example 2: The following code is from an administrative web application designed to allow users to kick off a backup of an Oracle database using a batch-file wrapper around the `rman` utility and then run a `cleanup.bat` script to delete some temporary files. The script `rmanDB.bat` accepts a single command line parameter, which specifies the type of backup to perform. Because access to the database is restricted, the application runs the backup as a privileged user.

```
...
String btype = request.getParameter("backuptype");
String cmd = new String("cmd.exe /K
\"c:\\util\\rmanDB.bat "+btype+"&&c:\\util\\cleanup.bat\"")
System.Runtime.getRuntime().exec(cmd);
...
```

The problem here is that the program does not do any validation on the `backuptype` parameter read from the user. Typically the `Runtime.exec()` function will not execute multiple commands, but in this case the program first runs the `cmd.exe` shell in order to run multiple commands with a single call to `Runtime.exec()`. Once the shell is invoked, it will allow for the execution of multiple commands separated by two ampersands. If an attacker passes a string of the form `"&& del c:\\dbms\\*.*"`, then the application will execute this command along with the others specified by the program. Because of the nature of the application, it runs with the privileges necessary to interact with the database, which means whatever command the attacker injects will run with those privileges as well.

Example 3: The following code is from a web application that allows users access to an interface through which they can update their password on the system. Part of the process for updating passwords in certain network environments is to run a `make` command in the `/var/yp` directory, the code for which is shown below.

```
...
System.Runtime.getRuntime().exec("make");
...
```

The problem here is that the program does not specify an absolute path for make and fails to clean its environment prior to executing the call to `Runtime.exec()`. If an attacker can modify the `$PATH` variable to point to a malicious binary called `make` and cause the program to be executed in their environment, then the malicious binary will be loaded instead of the one intended. Because of the nature of the application, it runs with the privileges necessary to perform system operations, which means the attacker's `make` will now be run with these privileges, possibly giving the attacker complete control of the system.

Some think that in the mobile world, classic vulnerabilities, such as command injection, do not make sense -- why would a user attack him or herself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 4: The following code reads commands to be executed from an Android intent.

```
...
        String[] cmds = this.getIntent().getStringArrayExtra("commands");
        Process p = Runtime.getRuntime().exec("su");
        DataOutputStream os = new DataOutputStream(p.getOutputStream());
        for (String cmd : cmds) {
                os.writeBytes(cmd+"\n");
        }
        os.writeBytes("exit\n");
        os.flush();
...
```

On a rooted device, a malicious application can force a victim application to execute arbitrary commands with super user privileges.

Recommendation

Do not allow users to have direct control over the commands executed by the program. In cases where user input must affect the command to be run, use the input only to make a selection from a predetermined set of safe commands. If the input appears to be malicious, the value passed to the command execution function should either default to some safe selection from this set or the program should decline to execute any command at all.

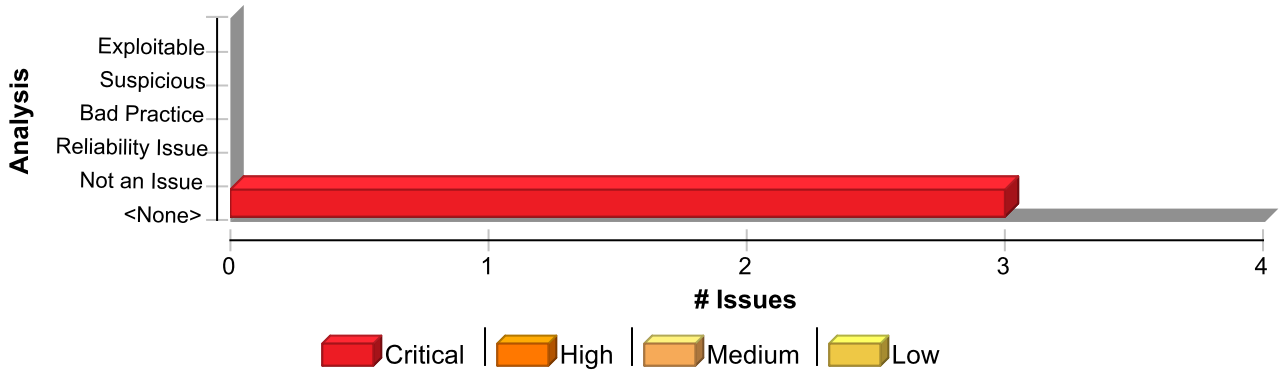
In cases where user input must be used as an argument to a command executed by the program, this approach often becomes impractical because the set of legitimate argument values is too large or too hard to keep track of. Developers often fall back on blacklisting in these situations. Blacklisting selectively rejects or escapes potentially dangerous characters before using the input. Any list of unsafe characters is likely to be incomplete and will be heavily dependent on the system where the commands are executed. A better approach is to create a whitelist of characters that are allowed to appear in the input and accept input composed exclusively of characters in the approved set.

An attacker may indirectly control commands executed by a program by modifying the environment in which they are executed. The environment should not be trusted and precautions should be taken to prevent an attacker from using some manipulation of the environment to perform an attack. Whenever possible, commands should be controlled by the application and executed using an absolute path. In cases where the path is not known at compile time, such as for cross-platform applications, an absolute path should be constructed from trusted values during execution. Command values and paths read from configuration files or the environment should be sanity-checked against a set of invariants that define valid values.

Other checks can sometimes be performed to detect if these sources may have been tampered with. For example, if a configuration file is world-writable, the program might refuse to run. In cases where information about the binary to be executed is known in advance, the program may perform checks to verify the identity of the binary. If a binary should always be owned by a particular user or have a particular set of access permissions assigned to it, these properties can be verified programmatically before the binary is executed.

Although it may be impossible to completely protect a program from an imaginative attacker bent on controlling the commands the program executes, be sure to apply the principle of least privilege wherever the program executes an external command: do not hold privileges that are not essential to the execution of the command.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Command Injection	3	3	0	6
Total	**3**	**3**	**0**	**6**

Command Injection	**Critical**
Package: org.owasp.webgoat.lessons	
WSDLScanning.java, line 143 (Command Injection)	**Critical**

Issue Details

Kingdom: Input Validation and Representation

Command Injection — Critical

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 143 (Command Injection) — Critical

Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getParameterValues
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:593

```
590  return (null);
591  }
592
593  return request.getParameterValues(name);
594  }
595
596
```

Sink Details

Sink: org.apache.axis.client.Call.setOperationName()
Enclosing Method: accessWGService()
File: WSDLScanning.java:143
Taint Flags: WEB, XSS

```
140  QName operationName = new QName(targetNamespace, proc);
141  Service service = new Service();
142  Call call = (Call) service.createCall();
143  call.setOperationName(operationName);
144  call.addParameter(parameterName, serviceName, ParameterMode.INOUT);
145  call.setReturnType(XMLType.XSD_STRING);
146  call.setUsername("guest");
```

Package: org.owasp.webgoat.util

Exec.java, line 103 (Command Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Command Injection — Critical

Package: org.owasp.webgoat.util

Exec.java, line 103 (Command Injection) — Critical

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.lang.Runtime.exec()
Enclosing Method: execOptions()
File: Exec.java:103
Taint Flags: TAINTED_PATH, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
100  try
101  {
102  // start the command
103  child = Runtime.getRuntime().exec(command);
104
105  // get the streams in and out of the command
106  InputStream processIn = child.getInputStream();
```

Exec.java, line 292 (Command Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
```

Command Injection	Critical

Package: org.owasp.webgoat.util

Exec.java, line 292 (Command Injection)	Critical

```
629  if (values == null)
630  {
```

Sink Details

Sink: java.lang.Runtime.exec()
Enclosing Method: execOptions()
File: Exec.java:292
Taint Flags: TAINTED_PATH, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
289  try
290  {
291  // start the command
292  child = Runtime.getRuntime().exec(command);
293
294  // get the streams in and out of the command
295  InputStream processIn = child.getInputStream();
```

Cookie Security: Cookie not Sent Over SSL (4 issues)

Abstract

A cookie is created without the `secure` flag set to `true`.

Explanation

Modern web browsers support a `secure` flag for each cookie. If the flag is set, the browser will only send the cookie over HTTPS. Sending cookies over an unencrypted channel can expose them to network sniffing attacks, so the secure flag helps keep a cookie's value confidential. This is especially important if the cookie contains private data or carries a session identifier.

Example 1: In the example below, a cookie added to the response without setting the `secure` flag.

```
Cookie cookie = new Cookie("emailCookie", email);
response.addCookie(cookie);
```

If your application uses both HTTPS and HTTP but does not set the `secure` flag, cookies sent during an HTTPS request will also be sent during subsequent HTTP requests. Sniffing network traffic over unencrypted wireless connections is a trivial task for attackers, so sending cookies (especially those with session IDs) over HTTP can result in application compromise.

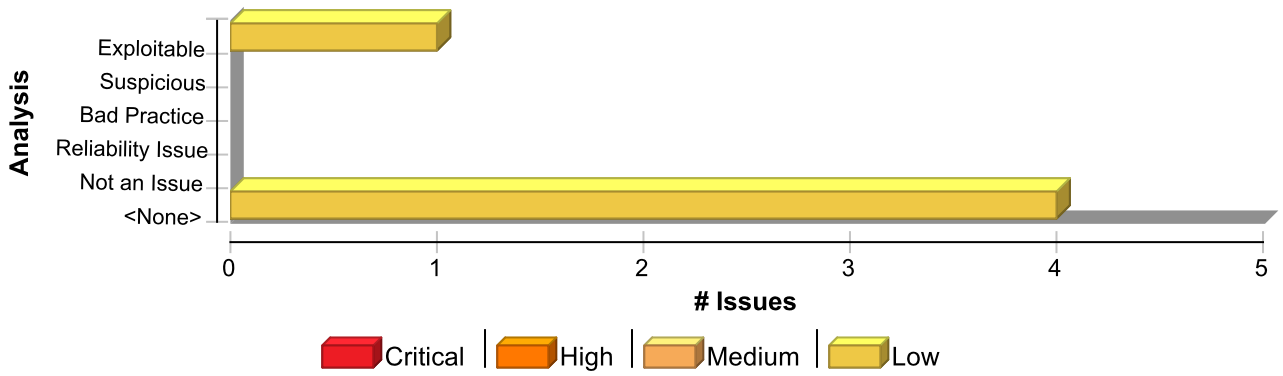
Recommendation

Set the `Secure` flag on all new cookies in order to instruct browsers not to send these cookies in the clear. This can be done by calling `setSecure(true)`.

Example 2:

```
Cookie cookie = new Cookie("emailCookie", email);
cookie.setSecure(true);
response.addCookie(cookie);
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Cookie Security: Cookie not Sent Over SSL	4	4	0	8
Total	**4**	**4**	**0**	**8**

Cookie Security: Cookie not Sent Over SSL — Low

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 209 (Cookie Security: Cookie not Sent Over SSL) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Sink Details

Sink: addCookie(cookie)
Enclosing Method: makeLogin()
File: WeakSessionID.java:209

```
206 {
207 weakid = newCookie();
208 Cookie cookie = new Cookie(SESSIONID, weakid);
209 s.getResponse().addCookie(cookie);
210 }
211
212 ec.addElement(new H1().addElement("Sign In "));
```

WeakAuthenticationCookie.java, line 146 (Cookie Security: Cookie not Sent Over SSL) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: addCookie(newCookie)
Enclosing Method: checkParams()
File: WeakAuthenticationCookie.java:146

```
143 {
144 Cookie newCookie = new Cookie(AUTHCOOKIE, loginID);
145 s.setMessage("Your identity has been remembered");
146 s.getResponse().addCookie(newCookie);
147
148 return (username);
```

Cookie Security: Cookie not Sent Over SSL — Low

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 146 (Cookie Security: Cookie not Sent Over SSL) — Low

```
149 }
```

Challenge2Screen.java, line 193 (Cookie Security: Cookie not Sent Over SSL) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: addCookie(newCookie)
Enclosing Method: doStage2()
File: Challenge2Screen.java:193

```
190 //<START_OMIT_SOURCE>
191
192 Cookie newCookie = new Cookie(USER_COOKIE, "White");
193 s.getResponse().addCookie(newCookie);
194
195 ElementContainer ec = new ElementContainer();
196 if (s.getParser().getStringParameter(Input.SUBMIT, "").equals(
```

Challenge2Screen.java, line 172 (Cookie Security: Cookie not Sent Over SSL) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: addCookie(newCookie)
Enclosing Method: doStage1()
File: Challenge2Screen.java:172

```
169 ec.addElement(input);
170
171 Cookie newCookie = new Cookie(USER_COOKIE, "White");
172 s.getResponse().addCookie(newCookie);
173 //<END_OMIT_SOURCE>
174
175 return (ec);
```

Cookie Security: HTTPOnly not Set (4 issues)

Abstract

The program creates a cookie, but fails to set the `HttpOnly` flag to `true`.

Explanation

All major browsers support the `HttpOnly` cookie property that prevents client-side scripts from accessing the cookie. Cross-site scripting attacks often access cookies in an attempt to steal session identifiers or authentication tokens. When `HttpOnly` is not enabled, attackers may more easily access user cookies.

Example 1: The code in the example below creates a cookie without setting the `HttpOnly` property.

```
javax.servlet.http.Cookie cookie = new
javax.servlet.http.Cookie("emailCookie", email);
// Missing a call to: cookie.setHttpOnly(true);
```

Recommendation

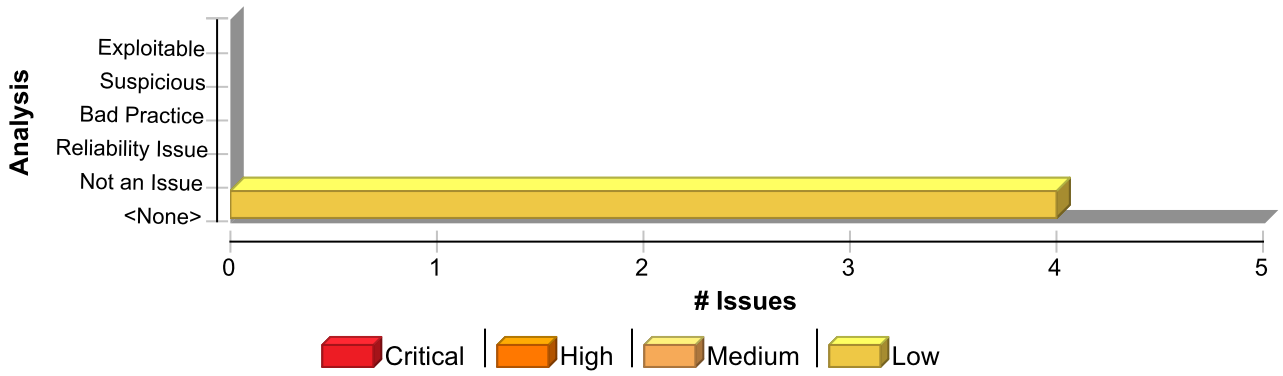
Enable the `HttpOnly` property when creating cookies. This can be done by calling, in the case of `javax.servlet.http.Cookie`, the `setHttpOnly(boolean)` method with argument `true`.

Example 2: The code in the example below creates the same cookie as the code in Example 1, but this time sets the `HttpOnly` parameter to `true`.

```
javax.servlet.http.Cookie cookie = new
javax.servlet.http.Cookie("emailCookie", email);
cookie.setHttpOnly(true);
```

Do not be lulled into a false sense of security by `HttpOnly`. As several mechanisms for bypassing it have been developed, it is not completely effective.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Cookie Security: HTTPOnly not Set	4	4	0	8
Total	**4**	**4**	**0**	**8**

Cookie Security: HTTPOnly not Set — Low

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 171 (Cookie Security: HTTPOnly not Set) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: newCookie = new Cookie(...)
Enclosing Method: doStage1()
File: Challenge2Screen.java:171

```
168 Input input = new Input(Input.HIDDEN, USER, "White");
169 ec.addElement(input);
170
171 Cookie newCookie = new Cookie(USER_COOKIE, "White");
172 s.getResponse().addCookie(newCookie);
173 //<END_OMIT_SOURCE>
174
```

WeakAuthenticationCookie.java, line 144 (Cookie Security: HTTPOnly not Set) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: newCookie = new Cookie(...)
Enclosing Method: checkParams()
File: WeakAuthenticationCookie.java:144

```
141
142 if (loginID != "")
143 {
144 Cookie newCookie = new Cookie(AUTHCOOKIE, loginID);
145 s.setMessage("Your identity has been remembered");
146 s.getResponse().addCookie(newCookie);
147
```

Challenge2Screen.java, line 192 (Cookie Security: HTTPOnly not Set) — Low

Issue Details

Cookie Security: HTTPOnly not Set — Low

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 192 (Cookie Security: HTTPOnly not Set) — Low

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: newCookie = new Cookie(...)
Enclosing Method: doStage2()
File: Challenge2Screen.java:192

```
189 {
190 //<START_OMIT_SOURCE>
191
192 Cookie newCookie = new Cookie(USER_COOKIE, "White");
193 s.getResponse().addCookie(newCookie);
194
195 ElementContainer ec = new ElementContainer();
```

WeakSessionID.java, line 208 (Cookie Security: HTTPOnly not Set) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: cookie = new Cookie(...)
Enclosing Method: makeLogin()
File: WeakSessionID.java:208

```
205 if (weakid == null)
206 {
207 weakid = newCookie();
208 Cookie cookie = new Cookie(SESSIONID, weakid);
209 s.getResponse().addCookie(cookie);
210 }
211
```

Cross-Site Request Forgery (27 issues)

Abstract

Form posts must contain a user-specific secret in order to prevent an attacker from making unauthorized requests.

Explanation

A cross-site request forgery (CSRF) vulnerability occurs when: 1. A Web application uses session cookies.

2. The application acts on an HTTP request without verifying that the request was made with the user's consent.

A nonce is a cryptographic random value that is sent with a message to prevent replay attacks. If the request does not contain a nonce that proves its provenance, the code that handles the request is vulnerable to a CSRF attack (unless it does not change the state of the application). This means a Web application that uses session cookies has to take special precautions in order to ensure that an attacker can't trick users into submitting bogus requests. Imagine a Web application that allows administrators to create new accounts by submitting this form:

```
<form method="POST" action="/new_user" >
  Name of new user: <input type="text" name="username">
  Password for new user: <input type="password" name="user_passwd">
    <input type="submit" name="action" value="Create User">
</form>
```

An attacker might set up a Web site with the following:

```
<form method="POST" action="http://www.example.com/new_user">
  <input type="hidden" name="username" value="hacker">
  <input type="hidden" name="user_passwd" value="hacked">
</form>
<script>
  document.usr_form.submit();
</script>
```

If an administrator for `example.com` visits the malicious page while she has an active session on the site, she will unwittingly create an account for the attacker. This is a CSRF attack. It is possible because the application does not have a way to determine the provenance of the request. Any request could be a legitimate action chosen by the user or a faked action set up by an attacker. The attacker does not get to see the Web page that the bogus request generates, so the attack technique is only useful for requests that alter the state of the application.

Applications that pass the session identifier in the URL rather than as a cookie do not have CSRF problems because there is no way for the attacker to access the session identifier and include it as part of the bogus request.

CSRF is entry number five on the 2007 OWASP Top 10 list.

Recommendation

Applications that use session cookies must include some piece of information in every form post that the back-end code can use to validate the provenance of the request. One way to do that is to include a random request identifier or nonce, like this:

```
  RequestBuilder rb = new RequestBuilder(RequestBuilder.POST, "/new_user");
  body = addToPost(body, new_username);
```

```
body = addToPost(body, new_passwd);
body = addToPost(body, request_id);
rb.sendRequest(body, new NewAccountCallback(callback));
```

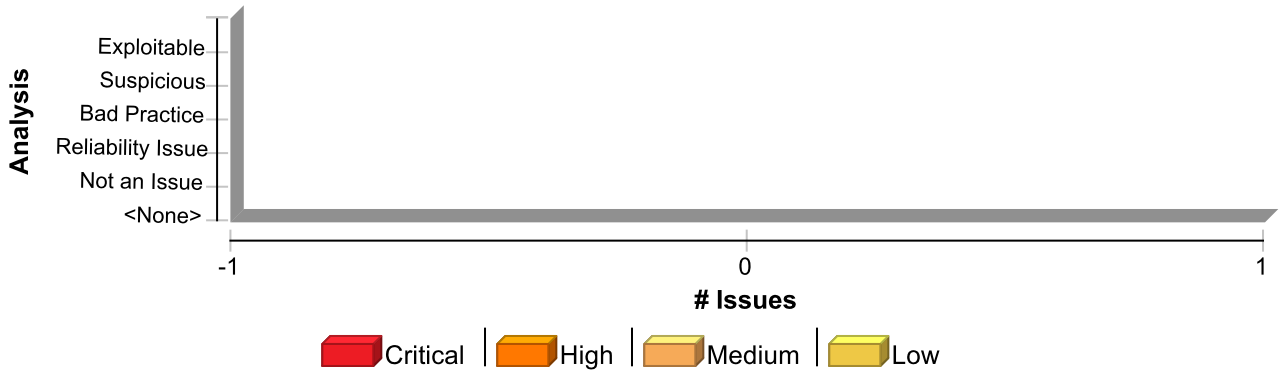
Then the back-end logic can validate the request identifier before processing the rest of the form data. When possible, the request identifier should be unique to each server request rather than shared across every request for a particular session. As with session identifiers, the harder it is for an attacker to guess the request identifier, the harder it is to conduct a successful CSRF attack. The token should not be easily guessed and it should be protected in the same way that session tokens are protected, such as using SSLv3.

Additional mitigation techniques include:

Framework protection: Most modern web application frameworks embed CSRF protection and they will automatically include and verify CSRF tokens. **Use a Challenge-Response control:** Forcing the customer to respond to a challenge sent by the server is a strong defense against CSRF. Some of the challenges that can be used for this purpose are: CAPTCHAs, password re-authentication and one-time tokens. **Check HTTP Referer/Origin headers:** An attacker won't be able to spoof these headers while performing a CSRF attack. This makes these headers a useful method to prevent CSRF attacks. **Double-submit Session Cookie:** Sending the session ID Cookie as a hidden form value in addition to the actual session ID Cookie is a good protection against CSRF attacks. The server will check both values and make sure they are identical before processing the rest of the form data. If an attacker submits a form in behalf of a user, he won't be able to modify the session ID cookie value as per the same-origin-policy. **Limit Session Lifetime:** When accessing protected resources using a CSRF attack, the attack will only be valid as long as the session ID sent as part of the attack is still valid on the server. Limiting the Session lifetime will reduce the probability of a successful attack.

The techniques described here can be defeated with XSS attacks. Effective CSRF mitigation includes XSS mitigation techniques.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Cross-Site Request Forgery	27	27	0	54
Total	**27**	**27**	**0**	**54**

Cross-Site Request Forgery — Low

Package: WebContent

webgoat.jsp, line 74 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: webgoat.jsp:74

```
71 <tr>
72 <td colspan = "2">
73 <div align="center" class="style2">
74 <form id="form" name="form" method="post" action="attack">
75 <input type="submit" name="start" value="Start WebGoat" />
76 </form>
77 </div>
```

webgoat_challenge.jsp, line 51 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: webgoat_challenge.jsp:51

```
48 <td><div align="center"><span class="style2_ch"></span></div></td>
49 </tr>
50 </table>
51 <form id="form" name="form" method="post" action="attack">
52 <div align="center">
53 <input type="submit" name="start" value="Start" />
54 </div>
```

Package: WebContent.lessons.CrossSiteScripting

EditProfile.jsp, line 10 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction	Not Predicted

Cross-Site Request Forgery — Low

Package: WebContent.lessons.CrossSiteScripting

EditProfile.jsp, line 10 (Cross-Site Request Forgery) — Low

Sink Details

File: EditProfile.jsp:10

```
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span></div>
9  <div class="lesson_text">
10  <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11  <Table>
12  <TR><TD>
13  First Name:
```

ListStaff.jsp, line 13 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ListStaff.jsp:13

```
10  <br>
11  <br>
12  <p>Select from the list below </p>
13  <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
14  <table width="60%" border="0" cellpadding="3">
15  <tr>
16  <td> <label>
```

ViewProfile.jsp, line 118 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:118

```
115  if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), CrossSiteScripting.EDITPROFILE_ACTION))
116  {
117  %>
118  <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
```

Cross-Site Request Forgery — Low

Package: WebContent.lessons.CrossSiteScripting

ViewProfile.jsp, line 118 (Cross-Site Request Forgery) — Low

```
119 <input type="hidden" name="<%=CrossSiteScripting.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
120 <input type="submit" name="action" value="<%=CrossSiteScripting.LISTSTAFF_ACTION%>"/>
121 </form></td>
```

ViewProfile.jsp, line 130 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:130

```
127 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), CrossSiteScripting.EDITPROFILE_ACTION))
128 {
129 %>
130 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
131 <input type="hidden" name="<%=CrossSiteScripting.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
132 <input type="submit" name="action" value="<%=CrossSiteScripting.EDITPROFILE_ACTION%>"/>
133 </form>
```

ViewProfile.jsp, line 143 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:143

```
140 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), CrossSiteScripting.DELETEPROFILE_ACTION))
141 {
142 %>
143 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
144 <input type="hidden" name="<%=CrossSiteScripting.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
145 <input type="submit" name="action" value="<%=CrossSiteScripting.DELETEPROFILE_ACTION%>"/>
146 </form>
```

ViewProfile.jsp, line 153 (Cross-Site Request Forgery) — Low

Issue Details

Cross-Site Request Forgery — Low

Package: WebContent.lessons.CrossSiteScripting

ViewProfile.jsp, line 153 (Cross-Site Request Forgery) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:153

```
150 </td>
151 <td width="190"> </td>
152 <td width="76">
153 <form method="POST">
154 <input type="submit" name="action" value="<%=CrossSiteScripting.LOGOUT_ACTION%>"/>
155 </form>
156 </td>
```

Login.jsp, line 9 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: Login.jsp:9

```
6 <%
7 WebSession webSession = ((WebSession)session.getAttribute("websession"));
8 %>
9 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
10 <label>
11 <select name="<%=CrossSiteScripting.EMPLOYEE_ID%>">
12 <%
```

SearchStaff.jsp, line 15 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

Cross-Site Request Forgery — Low

Package: WebContent.lessons.CrossSiteScripting

SearchStaff.jsp, line 15 (Cross-Site Request Forgery) — Low

File: SearchStaff.jsp:15

```
12 <%
13 }
14 %>
15 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
16 <label>Name
17 <input class="lesson_text_db" type="text" name="<%=CrossSiteScripting.SEARCHNAME%>"/>
18 </label>
```

Package: WebContent.lessons.RoleBasedAccessControl

EditProfile.jsp, line 10 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:10

```
7 %>
8 <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span> - Edit Profile Page</div>
9 <div class="lesson_text">
10 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11 <Table border="0" cellpadding="0" cellspacing="0">
12 <TR><TD width="110">
13 First Name:
```

ListStaff.jsp, line 14 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ListStaff.jsp:14

```
11 <br>
12 <p>Select from the list below </p>
13
```

Cross-Site Request Forgery — Low

Package: WebContent.lessons.RoleBasedAccessControl

ListStaff.jsp, line 14 (Cross-Site Request Forgery) — Low

```
14 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
15 <table width="60%" border="0" cellpadding="3">
16 <tr>
17 <td> <label>
```

ViewProfile.jsp, line 116 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:116

```
113 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), RoleBasedAccessControl.LISTSTAFF_ACTION))
114 {
115 %>
116 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
117 <input type="hidden" name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
118 <input type="submit" name="action" value="<%=RoleBasedAccessControl.LISTSTAFF_ACTION%>"/>
119 </form>
```

ViewProfile.jsp, line 128 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:128

```
125 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), RoleBasedAccessControl.EDITPROFILE_ACTION))
126 {
127 %>
128 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
129 <input type="hidden" name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
130 <input type="submit" name="action" value="<%=RoleBasedAccessControl.EDITPROFILE_ACTION%>"/>
131 </form>
```

ViewProfile.jsp, line 141 (Cross-Site Request Forgery) — Low

Issue Details

Cross-Site Request Forgery — Low

Package: WebContent.lessons.RoleBasedAccessControl

ViewProfile.jsp, line 141 (Cross-Site Request Forgery) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:141

```
138 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), RoleBasedAccessControl.DELETEPROFILE_ACTION))
139 {
140 %>
141 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
142 <input type="hidden" name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
143 <input type="submit" name="action" value="<%=RoleBasedAccessControl.DELETEPROFILE_ACTION%>"/>
144 </form>
```

ViewProfile.jsp, line 151 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:151

```
148 </td>
149 <td width="190"> </td>
150 <td width="76">
151 <form method="POST">
152 <input type="submit" name="action" value="<%=RoleBasedAccessControl.LOGOUT_ACTION%>"/>
153 </form>
154 </td>
```

Login.jsp, line 9 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

Cross-Site Request Forgery — Low

Package: WebContent.lessons.RoleBasedAccessControl

Login.jsp, line 9 (Cross-Site Request Forgery) — Low

File: Login.jsp:9

```
6  <%
7  WebSession webSession = ((WebSession)session.getAttribute("websession"));
8  %>
9  <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
10 <label>
11 <select name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>">
12 <%
```

error.jsp, line 10 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: error.jsp:10

```
7 %>
8 <br><br><br>An error has occurred.
9 <br><br><br>
10 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11
12 <input type="submit" name="action" value="<%=RoleBasedAccessControl.LOGIN_ACTION%>"/>
13 </form>
```

SearchStaff.jsp, line 15 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: SearchStaff.jsp:15

```
12 <%
13 }
14 %>
15 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
16 <label>Name
17 <input class="lesson_text_db" type="text" name="<%=RoleBasedAccessControl.SEARCHNAME%>"/>
```

Cross-Site Request Forgery — Low

Package: WebContent.lessons.RoleBasedAccessControl

SearchStaff.jsp, line 15 (Cross-Site Request Forgery) — Low

```
18 </label>
```

Package: WebContent.lessons.SQLInjection

EditProfile.jsp, line 10 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: EditProfile.jsp:10

```
7 %>
8 <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()%></span></div>
9 <div class="lesson_text">
10 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11 <Table>
12 <TR><TD>
13 First Name:
```

ListStaff.jsp, line 14 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: ListStaff.jsp:14

```
11 <br>
12 <p>Select from the list below </p>
13
14 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
15 <table width="60%" border="0" cellpadding="3">
16 <tr>
17 <td> <label>
```

ViewProfile.jsp, line 112 (Cross-Site Request Forgery) — Low

Issue Details

Cross-Site Request Forgery — Low

Package: WebContent.lessons.SQLInjection

ViewProfile.jsp, line 112 (Cross-Site Request Forgery) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:112

```
109 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), SQLInjection.LISTSTAFF_ACTION))
110 {
111 %>
112 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
113 <input type="hidden" name="<%=SQLInjection.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
114 <input type="submit" name="action" value="<%=SQLInjection.LISTSTAFF_ACTION%>"/>
115 </form>
```

ViewProfile.jsp, line 125 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:125

```
122 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), SQLInjection.EDITPROFILE_ACTION))
123 {
124 %>
125 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
126 <input type="hidden" name="<%=SQLInjection.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
127 <input type="submit" name="action" value="<%=SQLInjection.EDITPROFILE_ACTION%>"/>
128 </form>
```

ViewProfile.jsp, line 138 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

Cross-Site Request Forgery — Low

Package: WebContent.lessons.SQLInjection

ViewProfile.jsp, line 138 (Cross-Site Request Forgery) — Low

File: ViewProfile.jsp:138

```
135 if (webSession.isAuthorizedInLesson(webSession.getUserIdInLesson(), SQLInjection.DELETEPROFILE_ACTION))
136 {
137 %>
138 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
139 <input type="hidden" name="<%=SQLInjection.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
140 <input type="submit" name="action" value="<%=SQLInjection.DELETEPROFILE_ACTION%>"/>
141 </form>
```

ViewProfile.jsp, line 148 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:148

```
145 </td>
146 <td width="190"> </td>
147 <td width="76">
148 <form method="POST">
149 <input type="submit" name="action" value="<%=SQLInjection.LOGOUT_ACTION%>"/>
150 </form>
151 </td>
```

Login.jsp, line 9 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: Login.jsp:9

```
6 <%
7 WebSession webSession = ((WebSession)session.getAttribute("websession"));
8 %>
9 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
10 <label>
11 <select name="<%=SQLInjection.EMPLOYEE_ID%>">
```

Cross-Site Request Forgery — Low

Package: WebContent.lessons.SQLInjection

Login.jsp, line 9 (Cross-Site Request Forgery) — Low

```
12 <%
```

SearchStaff.jsp, line 15 (Cross-Site Request Forgery) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: SearchStaff.jsp:15

```
12 <%
13 }
14 %>
15 <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
16 <label>Name
17 <input class="lesson_text_db" type="text" name="<%=SQLInjection.SEARCHNAME%>"/>
18 </label>
```

Cross-Site Scripting: Persistent (38 issues)

Abstract

Sending unvalidated data to a web browser can result in the browser executing malicious code.

Explanation

Cross-site scripting (XSS) vulnerabilities occur when:

1. Data enters a web application through an untrusted source. In the case of Persistent (also known as Stored) XSS, the untrusted source is typically a database or other back-end data store, while in the case of Reflected XSS it is typically a web request.

2. The data is included in dynamic content that is sent to a web user without being validated.

The malicious content sent to the web browser often takes the form of a segment of JavaScript, but may also include HTML, Flash or any other type of code that the browser may execute. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site.

Example 1: The following JSP code segment queries a database for an employee with a given ID and prints the corresponding employee's name.

```
<%...
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery("select * from emp where id="+eid);
if (rs != null) {
   rs.next();
   String name = rs.getString("name");
}
%>

Employee Name: <%= name %>
```

This code functions correctly when the values of `name` are well-behaved, but it does nothing to prevent exploits if they are not. This code can appear less dangerous because the value of `name` is read from a database, whose contents are apparently managed by the application. However, if the value of `name` originates from user-supplied data, then the database can be a conduit for malicious content. Without proper input validation on all data stored in the database, an attacker may execute malicious commands in the user's web browser. This type of exploit, known as Persistent (or Stored) XSS, is particularly insidious because the indirection caused by the data store makes it more difficult to identify the threat and increases the possibility that the attack will affect multiple users. XSS got its start in this form with web sites that offered a "guestbook" to visitors. Attackers would include JavaScript in their guestbook entries, and all subsequent visitors to the guestbook page would execute the malicious code.

Example 2: The following JSP code segment reads an employee ID, `eid`, from an HTTP request and displays it to the user.

```
<% String eid = request.getParameter("eid"); %>
...
Employee ID: <%= eid %>
```

As in Example 1, this code operates correctly if `eid` contains only standard alphanumeric text. If `eid` has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response.

Initially this might not appear to be much of a vulnerability. After all, why would someone enter a URL that causes malicious code to run on their own computer? The real danger is that an attacker will create the malicious URL, then use e-mail or social engineering tricks to lure victims into visiting a link to the URL. When victims click the link, they unwittingly reflect the malicious content through the vulnerable web application back to their own computers. This mechanism of exploiting vulnerable web applications is known as Reflected XSS.

Some think that in the mobile world, classic web application vulnerabilities, such as cross-site scripting, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 3: The following code enables JavaScript in Android's WebView (by default, JavaScript is disabled) and loads a page based on the value received from an Android intent.

```
...
        WebView webview = (WebView) findViewById(R.id.webview);
        webview.getSettings().setJavaScriptEnabled(true);
        String url = this.getIntent().getExtras().getString("url");
        webview.loadUrl(url);
...
```

If the value of `url` starts with `javascript:`, JavaScript code that follows will execute within the context of the web page inside WebView.

As the examples demonstrate, XSS vulnerabilities are caused by code that includes unvalidated data in an HTTP response. There are three vectors by which an XSS attack can reach a victim:

- As in Example 1, the application stores dangerous data in a database or other trusted data store. The dangerous data is subsequently read back into the application and included in dynamic content. Persistent XSS exploits occur when an attacker injects dangerous content into a data store that is later read and included in dynamic content. From an attacker's perspective, the optimal place to inject malicious content is in an area that is displayed to either many users or particularly interesting users. Interesting users typically have elevated privileges in the application or interact with sensitive data that is valuable to the attacker. If one of these users executes malicious content, the attacker may be able to perform privileged operations on behalf of the user or gain access to sensitive data belonging to the user.

- As in Example 2, data is read directly from the HTTP request and reflected back in the HTTP response. Reflected XSS exploits occur when an attacker causes a user to supply dangerous content to a vulnerable web application, which is then reflected back to the user and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or e-mailed directly to victims. URLs constructed in this manner constitute the core of many phishing schemes, whereby an attacker convinces victims to visit a URL that refers to a vulnerable site. After the site reflects the attacker's content back to the user, the content is executed and proceeds to transfer private information, such as cookies that may include session information, from the user's machine to the attacker or perform other nefarious activities.

- As in Example 3, a source outside the application stores dangerous data in a database or other data store, and the dangerous data is subsequently read back into the application as trusted data and included in dynamic content.

A number of modern web frameworks provide mechanisms for performing validation of user input. Struts and Spring MVC are two examples. To highlight the unvalidated sources of input, the rulepacks dynamically re-prioritize the

issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

Recommendation

The solution to XSS is to ensure that validation occurs in the correct places and checks for the correct properties.

Since XSS vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating dynamic content, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for XSS.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for XSS is generally relatively easy. Despite its value, input validation for XSS does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent XSS vulnerabilities is to validate everything that enters the application and leaves the application destined for the user.

The most secure approach to validation for XSS is to create a whitelist of safe characters that are allowed to appear in HTTP content and accept input composed exclusively of characters in the approved set. For example, a valid username might only include alpha-numeric characters or a phone number might only include digits 0-9. However, this solution is often infeasible in web applications because many characters that have special meaning to the browser should still be considered valid input once they are encoded, such as a web design bulletin board that must accept HTML fragments from its users.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning for web browsers. Although the HTML standard defines what characters have special meaning, many web browsers try to correct common mistakes in HTML and may treat other characters as special in certain contexts, which is why we do not encourage the use of blacklists as a means to prevent XSS. The CERT(R) Coordination Center at the Software Engineering Institute at Carnegie Mellon University provides the following details about special characters in various contexts [1]:

In the content of a block-level element (in the middle of a paragraph of text):

- "<" is special because it introduces a tag.

- "&" is special because it introduces a character entity.

- ">" is special because some browsers treat it as special, on the assumption that the author of the page intended to include an opening "<", but omitted it in error.

The following principles apply to attribute values:

- In attribute values enclosed with double quotes, the double quotes are special because they mark the end of the attribute value.

- In attribute values enclosed with single quote, the single quotes are special because they mark the end of the attribute value.

- In attribute values without any quotes, white-space characters, such as space and tab, are special.

- "&" is special when used with certain attributes, because it introduces a character entity.

In URLs, for example, a search engine might provide a link within the results page that the user can click to re-run the search. This can be implemented by encoding the search query inside the URL, which introduces additional special characters:

- Space, tab, and new line are special because they mark the end of the URL.

- "&" is special because it either introduces a character entity or separates CGI parameters.

- Non-ASCII characters (that is, everything above 128 in the ISO-8859-1 encoding) are not allowed in URLs, so they are considered to be special in this context.

- The "%" symbol must be filtered from input anywhere parameters encoded with HTTP escape sequences are decoded by server-side code. For example, "%" must be filtered if input such as "%68%65%6C%6C%6F" becomes "hello" when it appears on the web page in question.

Within the body of a <SCRIPT> </SCRIPT>:

- Semicolons, parentheses, curly braces, and new line characters should be filtered out in situations where text could be inserted directly into a pre-existing script tag.

Server-side scripts:

- Server-side scripts that convert any exclamation characters (!) in input to double-quote characters (") on output might require additional filtering.

Other possibilities:

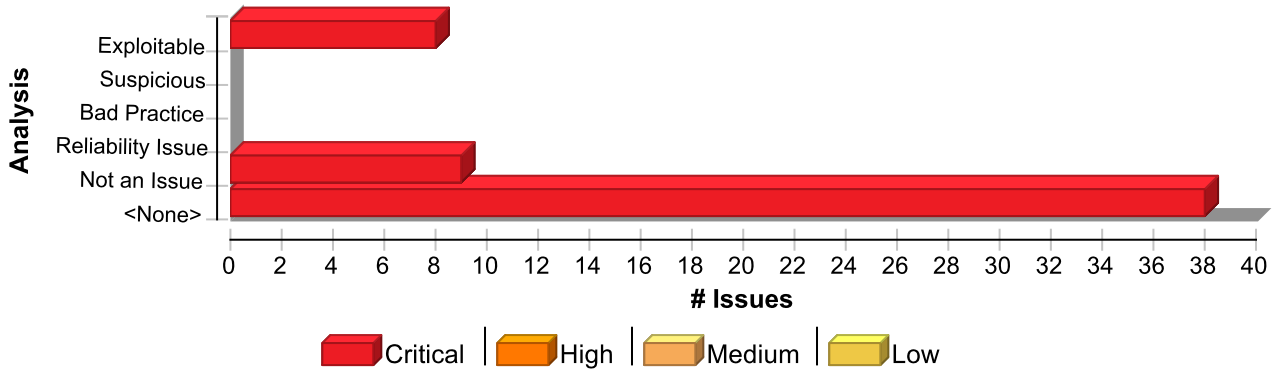
- If an attacker submits a request in UTF-7, the special character '<' appears as '+ADw-' and may bypass filtering. If the output is included in a page that does not explicitly specify an encoding format, then some browsers try to intelligently identify the encoding based on the content (in this case, UTF-7).

Once you identify the correct points in an application to perform validation for XSS attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. If special characters are not considered valid input to the application, then you can reject any input that contains special characters as invalid. A second option in this situation is to remove special characters with filtering. However, filtering has the side effect of changing any visual representation of the filtered content and may be unacceptable in circumstances where the integrity of the input must be preserved for display.

If input containing special characters must be accepted and displayed accurately, validation must encode any special characters to remove their significance. A complete list of ISO 8859-1 encoded values for special characters is provided as part of the official HTML specification [2].

Many application servers attempt to limit an application's exposure to cross-site scripting vulnerabilities by providing implementations for the functions responsible for setting certain specific HTTP response content that perform validation for the characters essential to a cross-site scripting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Cross-Site Scripting: Persistent	38	38	0	76
Total	**38**	**38**	**0**	**76**

Cross-Site Scripting: Persistent — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 11 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200 try
201 {
202 Statement answer_statement = WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
203 ResultSet answer_results = answer_statement.executeQuery( query );
204 if (answer_results.next())
205 name = answer_results.getString("first_name");
206 }
```

Cross-Site Scripting: Persistent — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 11 (Cross-Site Scripting: Persistent) — Critical

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:11
Taint Flags: DATABASE, XSS

```
8  Employee employee = (Employee) session.getAttribute("CrossSiteScripting." + CrossSiteScripting.EMPLOYEE_ATTRIBUTE_KEY);
9  // int myUserId = getIntSessionAttribute(webSession, "CrossSiteScripting." + CrossSiteScripting.USER_ID);
10 %>
11 <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span></div>
12 <div class="lesson_text">
13 <Table>
14 <TR><TD>
```

EditProfile.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Cross-Site Scripting: Persistent — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:8
Taint Flags: DATABASE, XSS

```
5  WebSession webSession = ((WebSession)session.getAttribute("websession"));
6  Employee employee = (Employee) session.getAttribute("CrossSiteScripting.Employee");
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span></div>
9  <div class="lesson_text">
10  <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11  <Table>
```

ListStaff.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()

Cross-Site Scripting: Persistent — Critical

Package: /lessons/CrossSiteScripting

ListStaff.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Enclosing Method: _jspService()
File: ListStaff.jsp:8
Taint Flags: DATABASE, XSS

```
5  WebSession webSession = ((WebSession)session.getAttribute("websession"));
6  int myUserId = webSession.getUserIdInLesson();
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span> - Staff Listing Page</div>
9  <br>
10  <br>
11  <br>
```

Package: /lessons/RoleBasedAccessControl

ListStaff.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Cross-Site Scripting: Persistent — Critical

Package: /lessons/RoleBasedAccessControl

ListStaff.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ListStaff.jsp:8
Taint Flags: DATABASE, XSS

```
5  WebSession webSession = ((WebSession)session.getAttribute("websession"));
6  int myUserId = webSession.getUserIdInLesson();
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span> - Staff Listing Page</div>
9  <br>
10  <br>
11  <br>
```

EditProfile.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()

Cross-Site Scripting: Persistent — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Enclosing Method: _jspService()
File: EditProfile.jsp:8
Taint Flags: DATABASE, XSS

```
5  WebSession webSession = ((WebSession)session.getAttribute("websession"));
6  Employee employee = (Employee) session.getAttribute("RoleBasedAccessControl.Employee");
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span> - Edit Profile Page</div>
9  <div class="lesson_text">
10  <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11 <Table border="0" cellpadding="0" cellspacing="0">
```

ViewProfile.jsp, line 9 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()

Cross-Site Scripting: Persistent — Critical

Package: /lessons/RoleBasedAccessControl

ViewProfile.jsp, line 9 (Cross-Site Scripting: Persistent) — Critical

File: ViewProfile.jsp:9
Taint Flags: DATABASE, XSS

```
6  WebSession webSession = ((WebSession)session.getAttribute("websession"));
7  // int myUserId = getIntSessionAttribute(webSession, "RoleBasedAccessControl." + RoleBasedAccessControl.USER_ID);
8  %>
9  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span> - View Profile Page</div>
10  <div class="lesson_text">
11  <Table>
12  <TR><TD>
```

Package: /lessons/SQLInjection

ListStaff.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()

Cross-Site Scripting: Persistent — Critical

Package: /lessons/SQLInjection

ListStaff.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Enclosing Method: _jspService()
File: ListStaff.jsp:8
Taint Flags: DATABASE, XSS

```
5  WebSession webSession = ((WebSession)session.getAttribute("websession"));
6  int myUserId = webSession.getUserIdInLesson();
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span> - Staff Listing Page</div>
9  <br>
10  <br>
11  <br>
```

ViewProfile.jsp, line 9 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()

Cross-Site Scripting: Persistent — Critical

Package: /lessons/SQLInjection

ViewProfile.jsp, line 9 (Cross-Site Scripting: Persistent) — Critical

File: ViewProfile.jsp:9
Taint Flags: DATABASE, XSS

```
6  Employee employee = (Employee) session.getAttribute("SQLInjection." + SQLInjection.EMPLOYEE_ATTRIBUTE_KEY);
7  // int myUserId = getIntSessionAttribute(webSession, "SQLInjection." + SQLInjection.USER_ID);
8  %>
9  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span></div>
10  <div class="lesson_text">
11  <Table>
12  <TR><TD>
```

EditProfile.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DefaultLessonAction.getUserName
File: JavaSource/org/owasp/webgoat/lessons/DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement =
WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
204  if (answer_results.next())
205  name = answer_results.getString("first_name");
206  }
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:8

Cross-Site Scripting: Persistent — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 8 (Cross-Site Scripting: Persistent) — Critical

Taint Flags: DATABASE, XSS

```
5  WebSession webSession = ((WebSession)session.getAttribute("websession"));
6  Employee employee = (Employee) session.getAttribute("SQLInjection.Employee");
7 %>
8  <div class="lesson_title_box"><strong>Welcome Back </strong><span class="lesson_text_db"><%=webSession.getUserNameInLesson()
%></span></div>
9  <div class="lesson_text">
10  <form id="form1" name="form1" method="post" action="attack?menu=<%=webSession.getCurrentMenu()%>">
11  <Table>
```

Package: org.owasp.webgoat.lessons

BackDoors.java, line 127 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.BackDoors.concept1
File: JavaSource/org/owasp/webgoat/lessons/BackDoors.java:113

```
110  .setMessage("You have succeeded in exploiting the vulnerable query and created another SQL statement. Now
move to stage 2 to learn how to create a backdoor or a DB worm");
111  }
112
113  ResultSet rs = statement.executeQuery(arrSQL[0]);
114  if (rs.next())
115  {
116  Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: concept1()
File: BackDoors.java:127

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 127 (Cross-Site Scripting: Persistent) — Critical

Taint Flags: DATABASE, XSS

```
124 tr = new TR();
125 tr.addElement(new TD(rs.getString("userid")));
126 tr.addElement(new TD(rs.getString("password")));
127 tr.addElement(new TD(rs.getString("ssn")));
128 tr.addElement(new TD(rs.getString("salary")));
129 t.addElement(tr);
130 ec.addElement(t);
```

BackDoors.java, line 125 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.BackDoors.concept1
File: JavaSource/org/owasp/webgoat/lessons/BackDoors.java:113

```
110 .setMessage("You have succeeded in exploiting the vulnerable query and created another SQL statement. Now move to stage 2 to learn how to create a backdoor or a DB worm");
111 }
112
113 ResultSet rs = statement.executeQuery(arrSQL[0]);
114 if (rs.next())
115 {
116 Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: concept1()
File: BackDoors.java:125
Taint Flags: DATABASE, XSS

```
122 tr.addElement(new TD("Salary"));
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 125 (Cross-Site Scripting: Persistent) — Critical

```
123  t.addElement(tr);
124  tr = new TR();
125  tr.addElement(new TD(rs.getString("userid")));
126  tr.addElement(new TD(rs.getString("password")));
127  tr.addElement(new TD(rs.getString("ssn")));
128  tr.addElement(new TD(rs.getString("salary")));
```

BackDoors.java, line 128 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.BackDoors.concept1
File: JavaSource/org/owasp/webgoat/lessons/BackDoors.java:113

```
110  .setMessage("You have succeeded in exploiting the vulnerable query and created another SQL statement. Now move to stage 2 to learn how to create a backdoor or a DB worm");
111  }
112
113  ResultSet rs = statement.executeQuery(arrSQL[0]);
114  if (rs.next())
115  {
116  Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: concept1()
File: BackDoors.java:128
Taint Flags: DATABASE, XSS

```
125  tr.addElement(new TD(rs.getString("userid")));
126  tr.addElement(new TD(rs.getString("password")));
127  tr.addElement(new TD(rs.getString("ssn")));
128  tr.addElement(new TD(rs.getString("salary")));
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 128 (Cross-Site Scripting: Persistent) — Critical

```
129 t.addElement(tr);
130 ec.addElement(t);
131 }
```

StoredXss.java, line 233 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.StoredXss.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/StoredXss.java:224

```
221 ResultSet.CONCUR_READ_ONLY);
222 statement.setString(1, getNameroot(s.getUserName()) + "%");
223 statement.setInt(2, messageNum);
224 ResultSet results = statement.executeQuery();
225
226 if ((results != null) && results.first())
227 {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeCurrent()
File: StoredXss.java:233
Taint Flags: DATABASE, XSS

```
230 Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
231 .setBorder(0);
232 TR row1 = new TR(new TD(new B(new StringElement("Title:"))));
233 row1.addElement(new TD(new StringElement(results
234 .getString(TITLE_COL))));
235 t.addElement(row1);
236
```

StoredXss.java, line 239 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

StoredXss.java, line 239 (Cross-Site Scripting: Persistent) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.StoredXss.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/StoredXss.java:224

```
221  ResultSet.CONCUR_READ_ONLY);
222  statement.setString(1, getNameroot(s.getUserName()) + "%");
223  statement.setInt(2, messageNum);
224  ResultSet results = statement.executeQuery();
225
226  if ((results != null) && results.first())
227  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeCurrent()
File: StoredXss.java:239
Taint Flags: DATABASE, XSS

```
236
237  String messageData = results.getString(MESSAGE_COL);
238  TR row2 = new TR(new TD(new B(new StringElement("Message:"))));
239  row2.addElement(new TD(new StringElement(messageData)));
240  t.addElement(row2);
241
242  // Edited by Chuck Willis - added display of the user who posted the message, so that
```

StoredXss.java, line 247 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.StoredXss.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/StoredXss.java:224

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

StoredXss.java, line 247 (Cross-Site Scripting: Persistent) — Critical

```
221  ResultSet.CONCUR_READ_ONLY);
222  statement.setString(1, getNameroot(s.getUserName()) + "%");
223  statement.setInt(2, messageNum);
224  ResultSet results = statement.executeQuery();
225
226  if ((results != null) && results.first())
227  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeCurrent()
File: StoredXss.java:247
Taint Flags: DATABASE, XSS

```
244  // they can see that the message is attributed to that user
245
246  TR row3 = new TR(new TD(new StringElement("Posted By:")));
247  row3.addElement(new TD(new StringElement(results
248  .getString(USER_COL))));
249  t.addElement(row3);
250
```

SqlNumericInjection.java, line 265 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SqlNumericInjection.getStations
File: JavaSource/org/owasp/webgoat/lessons/SqlNumericInjection.java:299

```
296  Statement statement = connection.createStatement(
297  ResultSet.TYPE_SCROLL_INSENSITIVE,
298  ResultSet.CONCUR_READ_ONLY);
299  ResultSet results = statement.executeQuery(query);
300
301  if ((results != null) && (results.first() == true))
302  {
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 265 (Cross-Site Scripting: Persistent) — Critical

Sink Details

Sink: org.apache.ecs.html.Option.addElement()
Enclosing Method: makeStationList()
File: SqlNumericInjection.java:265
Taint Flags: DATABASE, XSS

```
262 {
263 String key = (String) it.next();
264 select.addElement(new Option(key).addElement((String) stations
265 .get(key)));
266 }
267 ec.addElement(select);
268 ec.addElement(new P());
```

WSDLScanning.java, line 221 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: org.apache.axis.client.Call.invoke()
From: org.owasp.webgoat.lessons.WSDLScanning.accessWGService
File: JavaSource/org/owasp/webgoat/lessons/WSDLScanning.java:150

```
147 call.setPassword("guest");
148 call.setTargetEndpointAddress("http://localhost/WebGoat/services/"
149 + serv);
150 Object result = call.invoke(new Object[] { parameterValue });
151 return result;
152 }
153 catch (RemoteException e)
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: createContent()
File: WSDLScanning.java:221
Taint Flags: WEBSERVICE, XSS

```
218 {
219 header.addElement(new TD().addElement(fields[i]));
220 results.addElement(new TD()
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 221 (Cross-Site Scripting: Persistent) — Critical

```
221  .addElement((String) accessWGService("WSDLScanning",
222  fields[i], "acct_num", new Integer(id))));
223  }
224  if (fields.length == 0)
```

CSRF.java, line 193 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.CSRF.makeList
File: JavaSource/org/owasp/webgoat/lessons/CSRF.java:181

```
178
179  Statement statement = connection.createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
180
181  ResultSet results = statement.executeQuery( STANDARD_QUERY + " WHERE user_name LIKE '" +
getNameroot( s.getUserName() ) + "%'" );
182
183  if ( ( results != null ) && ( results.first() == true ) )
184  {
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: makeList()
File: CSRF.java:193
Taint Flags: DATABASE, XSS

```
190  "&Screen=" + String.valueOf(getScreenId()) +
191  "&menu=" + getDefaultCategory().getRanking().toString() +
192  "' style='cursor:hand'>" + results.getString( TITLE_COL ) + "</a>";
193  TD td = new TD().addElement( link );
194  TR tr = new TR().addElement( td );
195  t.addElement( tr );
196  }
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 264 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SqlNumericInjection.getStations
File: JavaSource/org/owasp/webgoat/lessons/SqlNumericInjection.java:299

```
296 Statement statement = connection.createStatement(
297 ResultSet.TYPE_SCROLL_INSENSITIVE,
298 ResultSet.CONCUR_READ_ONLY);
299 ResultSet results = statement.executeQuery(query);
300
301 if ((results != null) && (results.first() == true))
302 {
```

Sink Details

Sink: org.apache.ecs.html.Option.Option()
Enclosing Method: makeStationList()
File: SqlNumericInjection.java:264
Taint Flags: DATABASE, PRIMARY_KEY, XSS

```
261 while (it.hasNext())
262 {
263 String key = (String) it.next();
264 select.addElement(new Option(key).addElement((String) stations
265 .get(key)));
266 }
267 ec.addElement(select);
```

CSRF.java, line 248 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

CSRF.java, line 248 (Cross-Site Scripting: Persistent) — Critical

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.CSRF.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/CSRF.java:241

```
238  PreparedStatement statement = connection.prepareStatement( query,
ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
239  statement.setString(1, getNameroot( s.getUserName() ) + "%");
240  statement.setInt(2, messageNum);
241  ResultSet results = statement.executeQuery();
242
243  if ( ( results != null ) && results.first() )
244  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeCurrent()
File: CSRF.java:248
Taint Flags: DATABASE, XSS

```
245  ec.addElement( new H1( "Message Contents For: " + results.getString( TITLE_COL )) );
246  Table t = new Table( 0 ).setCellSpacing( 0 ).setCellPadding( 0 ).setBorder( 0 );
247  TR row1 = new TR( new TD( new B(new StringElement( "Title:" )) ) );
248  row1.addElement( new TD( new StringElement( results.getString( TITLE_COL ) ) ) );
249  t.addElement( row1 );
250
251  String messageData = results.getString( MESSAGE_COL );
```

CSRF.java, line 253 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.CSRF.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/CSRF.java:241

```
238  PreparedStatement statement = connection.prepareStatement( query,
ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
239  statement.setString(1, getNameroot( s.getUserName() ) + "%");
240  statement.setInt(2, messageNum);
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

CSRF.java, line 253 (Cross-Site Scripting: Persistent) — Critical

```
241  ResultSet results = statement.executeQuery();
242
243  if ( ( results != null ) && results.first() )
244  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeCurrent()
File: CSRF.java:253
Taint Flags: DATABASE, XSS

```
250
251  String messageData = results.getString( MESSAGE_COL );
252  TR row2 = new TR( new TD( new B(new StringElement( "Message:" )) ) );
253  row2.addElement( new TD( new StringElement( messageData ) ) );
254  t.addElement( row2 );
255
256  TR row3 = new TR( new TD( new StringElement( "Posted By:" ) ) );
```

CSRF.java, line 257 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.CSRF.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/CSRF.java:241

```
238  PreparedStatement statement = connection.prepareStatement( query,
ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
239  statement.setString(1, getNameroot( s.getUserName() ) + "%");
240  statement.setInt(2, messageNum);
241  ResultSet results = statement.executeQuery();
242
243  if ( ( results != null ) && results.first() )
244  {
```

Sink Details

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

CSRF.java, line 257 (Cross-Site Scripting: Persistent) — Critical

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeCurrent()
File: CSRF.java:257
Taint Flags: DATABASE, XSS

```
254 t.addElement( row2 );
255
256 TR row3 = new TR( new TD( new StringElement( "Posted By:" ) ) );
257 row3.addElement( new TD( new StringElement( results.getString( USER_COL ) ) ) );
258 t.addElement( row3 );
259
260 ec.addElement( t );
```

CSRF.java, line 245 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.CSRF.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/CSRF.java:241

```
238 PreparedStatement statement = connection.prepareStatement( query,
ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
239 statement.setString(1, getNameroot( s.getUserName() ) + "%");
240 statement.setInt(2, messageNum);
241 ResultSet results = statement.executeQuery();
242
243 if ( ( results != null ) && results.first() )
244 {
```

Sink Details

Sink: org.apache.ecs.html.H1.H1()
Enclosing Method: makeCurrent()
File: CSRF.java:245

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

CSRF.java, line 245 (Cross-Site Scripting: Persistent) — Critical

Taint Flags: DATABASE, XSS

```
242
243 if ( ( results != null ) && results.first() )
244 {
245 ec.addElement( new H1( "Message Contents For: " + results.getString( TITLE_COL )) );
246 Table t = new Table( 0 ).setCellSpacing( 0 ).setCellPadding( 0 ).setBorder( 0 );
247 TR row1 = new TR( new TD( new B(new StringElement( "Title:" )) ) );
248 row1.addElement( new TD( new StringElement( results.getString( TITLE_COL ) ) ) );
```

BackDoors.java, line 126 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.BackDoors.concept1
File: JavaSource/org/owasp/webgoat/lessons/BackDoors.java:113

```
110 .setMessage("You have succeeded in exploiting the vulnerable query and created another SQL statement. Now move to stage 2 to learn how to create a backdoor or a DB worm");
111 }
112
113 ResultSet rs = statement.executeQuery(arrSQL[0]);
114 if (rs.next())
115 {
116 Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: concept1()
File: BackDoors.java:126
Taint Flags: DATABASE, XSS

```
123 t.addElement(tr);
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 126 (Cross-Site Scripting: Persistent) — Critical

```
124  tr = new TR();
125  tr.addElement(new TD(rs.getString("userid")));
126  tr.addElement(new TD(rs.getString("password")));
127  tr.addElement(new TD(rs.getString("ssn")));
128  tr.addElement(new TD(rs.getString("salary")));
129  t.addElement(tr);
```

StoredXss.java, line 228 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.StoredXss.makeCurrent
File: JavaSource/org/owasp/webgoat/lessons/StoredXss.java:224

```
221  ResultSet.CONCUR_READ_ONLY);
222  statement.setString(1, getNameroot(s.getUserName()) + "%");
223  statement.setInt(2, messageNum);
224  ResultSet results = statement.executeQuery();
225
226  if ((results != null) && results.first())
227  {
```

Sink Details

Sink: org.apache.ecs.html.H1.H1()
Enclosing Method: makeCurrent()
File: StoredXss.java:228
Taint Flags: DATABASE, XSS

```
225
226  if ((results != null) && results.first())
227  {
228  ec.addElement(new H1("Message Contents For: "
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.lessons

StoredXss.java, line 228 (Cross-Site Scripting: Persistent) — Critical

```
229  + results.getString(TITLE_COL)));
230  Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
231  .setBorder(0);
```

Package: org.owasp.webgoat.session

ECSFactory.java, line 292 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.StoredXss.makeList
File: JavaSource/org/owasp/webgoat/lessons/StoredXss.java:343

```
340  // but not anyone elses. This allows users to try out XSS to grab another user's
341  // cookies, but not get confused by other users scripts
342
343  ResultSet results = statement.executeQuery(STANDARD_QUERY
344  + " WHERE user_name LIKE '" + getNameroot(s.getUserName())
345  + "%'");
346
```

Sink Details

Sink: org.apache.ecs.html.U.addElement()
Enclosing Method: makeLink()
File: ECSFactory.java:292
Taint Flags: DATABASE, XSS

```
289
290  A a = new A(href);
291
292  a.addElement(new U().addElement(text));
293
294  a.addAttribute("style", "cursor:hand");
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

ECSFactory.java, line 292 (Cross-Site Scripting: Persistent) — Critical

```
295
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SqlStringInjection.injectableQuery
File: JavaSource/org/owasp/webgoat/lessons/SqlStringInjection.java:112

```
109  Statement statement = connection.createStatement(
110  ResultSet.TYPE_SCROLL_INSENSITIVE,
111  ResultSet.CONCUR_READ_ONLY);
112  ResultSet results = statement.executeQuery(query);
113
114  if ((results != null) && (results.first() == true))
115  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152  for (int i = 1; i < (numColumns + 1); i++)
153  {
154  row.addElement(new TD(results.getString(i).replaceAll(" ",
155  " ")));
156  }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.ThreadSafetyProblem.createContent
File: JavaSource/org/owasp/webgoat/lessons/ThreadSafetyProblem.java:103

```
100  Statement statement = connection.createStatement(
101  ResultSet.TYPE_SCROLL_INSENSITIVE,
102  ResultSet.CONCUR_READ_ONLY);
103  ResultSet results = statement.executeQuery(query);
104
105  if ((results != null) && (results.first() == true))
106  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152  for (int i = 1; i < (numColumns + 1); i++)
153  {
154  row.addElement(new TD(results.getString(i).replaceAll(" ",
155  " ")));
156  }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.admin.ProductsAdminScreen.createContent
File: JavaSource/org/owasp/webgoat/lessons/admin/ProductsAdminScreen.java:75

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

```
72  Statement statement = connection.createStatement(
73  ResultSet.TYPE_SCROLL_INSENSITIVE,
74  ResultSet.CONCUR_READ_ONLY);
75  ResultSet results = statement.executeQuery(QUERY);
76
77  if (results != null)
78  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152  for (int i = 1; i < (numColumns + 1); i++)
153  {
154  row.addElement(new TD(results.getString(i).replaceAll(" ",
155  " ")));
156  }
157
```

ECSFactory.java, line 450 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.Challenge2Screen.doStage2
File: JavaSource/org/owasp/webgoat/lessons/Challenge2Screen.java:220

```
217  Vector<String> v = new Vector<String>();
218  try
219  {
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

ECSFactory.java, line 450 (Cross-Site Scripting: Persistent) — Critical

```
220  ResultSet results = statement3.executeQuery(query);
221
222  while (results.next())
223  {
```

Sink Details

Sink: org.apache.ecs.html.Select.addElement()
Enclosing Method: makePulldown()
File: ECSFactory.java:450
Taint Flags: DATABASE, PRIMARY_KEY, XSS

```
447
448  Select s = new Select(name);
449
450  s.addElement((String[]) options.toArray(new String[options.size()]));
451
452  return (s);
453  }
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.SqlNumericInjection.parameterizedQuery
File: JavaSource/org/owasp/webgoat/lessons/SqlNumericInjection.java:208

```
205  query, ResultSet.TYPE_SCROLL_INSENSITIVE,
206  ResultSet.CONCUR_READ_ONLY);
207  statement.setInt(1, Integer.parseInt(station));
208  ResultSet results = statement.executeQuery();
209
210  if ((results != null) && (results.first() == true))
211  {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152 for (int i = 1; i < (numColumns + 1); i++)
153 {
154 row.addElement(new TD(results.getString(i).replaceAll(" ",
155 " ")));
156 }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.Challenge2Screen.doStage2
File: JavaSource/org/owasp/webgoat/lessons/Challenge2Screen.java:220

```
217 Vector<String> v = new Vector<String>();
218 try
219 {
220 ResultSet results = statement3.executeQuery(query);
221
222 while (results.next())
223 {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152 for (int i = 1; i < (numColumns + 1); i++)
153 {
154 row.addElement(new TD(results.getString(i).replaceAll(" ",
155 " ")));
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

```
156 }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.WsSqlInjection.getResults
File: JavaSource/org/owasp/webgoat/lessons/WsSqlInjection.java:240

```
237 Statement statement = connection.createStatement(
238 ResultSet.TYPE_SCROLL_INSENSITIVE,
239 ResultSet.CONCUR_READ_ONLY);
240 ResultSet results = statement.executeQuery(query);
241 return results;
242 }
243 catch (SQLException sqle)
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152 for (int i = 1; i < (numColumns + 1); i++)
153 {
154 row.addElement(new TD(results.getString(i).replaceAll(" ",
155 " ")));
156 }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.DOS_Login.createContent
File: JavaSource/org/owasp/webgoat/lessons/DOS_Login.java:114

```
111  Statement statement = connection.createStatement(
112  ResultSet.TYPE_SCROLL_INSENSITIVE,
113  ResultSet.CONCUR_READ_ONLY);
114  ResultSet results = statement.executeQuery(query);
115  if ((results != null) && (results.first() == true))
116  {
117  ResultSetMetaData resultsMetaData = results.getMetaData();
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152  for (int i = 1; i < (numColumns + 1); i++)
153  {
154  row.addElement(new TD(results.getString(i).replaceAll(" ",
155  " ")));
156  }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.admin.ViewDatabase.createContent
File: JavaSource/org/owasp/webgoat/lessons/admin/ViewDatabase.java:89

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

```
86  Statement statement = connection.createStatement(
87  ResultSet.TYPE_SCROLL_INSENSITIVE,
88  ResultSet.CONCUR_READ_ONLY);
89  ResultSet results = statement.executeQuery(sqlStatement
90  .toString());
91
92  if ((results != null) && (results.first() == true))
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152  for (int i = 1; i < (numColumns + 1); i++)
153  {
154  row.addElement(new TD(results.getString(i).replaceAll(" ",
155  " ")));
156  }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SqlNumericInjection.injectableQuery
File: JavaSource/org/owasp/webgoat/lessons/SqlNumericInjection.java:130

```
127  Statement statement = connection.createStatement(
128  ResultSet.TYPE_SCROLL_INSENSITIVE,
129  ResultSet.CONCUR_READ_ONLY);
130  ResultSet results = statement.executeQuery(query);
131
132  if ((results != null) && (results.first() == true))
133  {
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152 for (int i = 1; i < (numColumns + 1); i++)
153 {
154 row.addElement(new TD(results.getString(i).replaceAll(" ",
155 " ")));
156 }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.admin.UserAdminScreen.createContent
File: JavaSource/org/owasp/webgoat/lessons/admin/UserAdminScreen.java:75

```
72 Statement statement = connection.createStatement(
73 ResultSet.TYPE_SCROLL_INSENSITIVE,
74 ResultSet.CONCUR_READ_ONLY);
75 ResultSet results = statement.executeQuery(QUERY);
76
77 if (results != null)
78 {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152 for (int i = 1; i < (numColumns + 1); i++)
153 {
```

Cross-Site Scripting: Persistent — Critical

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

```
154 row.addElement(new TD(results.getString(i).replaceAll(" ",
155 " ")));
156 }
157
```

DatabaseUtilities.java, line 154 (Cross-Site Scripting: Persistent) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.SqlStringInjection.parameterizedQuery
File: JavaSource/org/owasp/webgoat/lessons/SqlStringInjection.java:194

```
191 query, ResultSet.TYPE_SCROLL_INSENSITIVE,
192 ResultSet.CONCUR_READ_ONLY);
193 statement.setString(1, accountName);
194 ResultSet results = statement.executeQuery();
195
196 if ((results != null) && (results.first() == true))
197 {
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: writeTable()
File: DatabaseUtilities.java:154
Taint Flags: DATABASE, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, XSS

```
151
152 for (int i = 1; i < (numColumns + 1); i++)
153 {
154 row.addElement(new TD(results.getString(i).replaceAll(" ",
155 " ")));
156 }
157
```

Cross-Site Scripting: Reflected (70 issues)

Abstract

Sending unvalidated data to a web browser can result in the browser executing malicious code.

Explanation

Cross-site scripting (XSS) vulnerabilities occur when:

1. Data enters a web application through an untrusted source. In the case of Reflected XSS, the untrusted source is typically a web request, while in the case of Persisted (also known as Stored) XSS it is typically a database or other back-end data store.

2. The data is included in dynamic content that is sent to a web user without being validated.

The malicious content sent to the web browser often takes the form of a segment of JavaScript, but may also include HTML, Flash or any other type of code that the browser may execute. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site.

Example 1: The following JSP code segment reads an employee ID, `eid`, from an HTTP request and displays it to the user.

```
<% String eid = request.getParameter("eid"); %>
...
Employee ID: <%= eid %>
```

The code in this example operates correctly if `eid` contains only standard alphanumeric text. If `eid` has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response.

Initially this might not appear to be much of a vulnerability. After all, why would someone enter a URL that causes malicious code to run on their own computer? The real danger is that an attacker will create the malicious URL, then use e-mail or social engineering tricks to lure victims into visiting a link to the URL. When victims click the link, they unwittingly reflect the malicious content through the vulnerable web application back to their own computers. This mechanism of exploiting vulnerable web applications is known as Reflected XSS.

Example 2: The following JSP code segment queries a database for an employee with a given ID and prints the corresponding employee's name.

```
<%...
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery("select * from emp where id="+eid);
if (rs != null) {
   rs.next();
   String name = rs.getString("name");
}
%>

Employee Name: <%= name %>
```

As in Example 1, this code functions correctly when the values of `name` are well-behaved, but it does nothing to prevent exploits if they are not. Again, this code can appear less dangerous because the value of `name` is read from a database, whose contents are apparently managed by the application. However, if the value of `name` originates from user-supplied data, then the database can be a conduit for malicious content. Without proper input validation on all data stored in the database, an attacker may execute malicious commands in the user's web browser. This type of exploit, known as Persistent (or Stored) XSS, is particularly insidious because the indirection caused by the data store makes it more difficult to identify the threat and increases the possibility that the attack will affect multiple users. XSS got its start in this form with web sites that offered a "guestbook" to visitors. Attackers would include JavaScript in their guestbook entries, and all subsequent visitors to the guestbook page would execute the malicious code.

Some think that in the mobile world, classic web application vulnerabilities, such as cross-site scripting, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 3: The following code enables JavaScript in Android's WebView (by default, JavaScript is disabled) and loads a page based on the value received from an Android intent.

```
...
        WebView webview = (WebView) findViewById(R.id.webview);
        webview.getSettings().setJavaScriptEnabled(true);
        String url = this.getIntent().getExtras().getString("url");
        webview.loadUrl(url);
...
```

If the value of `url` starts with `javascript:`, JavaScript code that follows will execute within the context of the web page inside WebView.

As the examples demonstrate, XSS vulnerabilities are caused by code that includes unvalidated data in an HTTP response. There are three vectors by which an XSS attack can reach a victim:

- As in Example 1, data is read directly from the HTTP request and reflected back in the HTTP response. Reflected XSS exploits occur when an attacker causes a user to supply dangerous content to a vulnerable web application, which is then reflected back to the user and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or e-mailed directly to victims. URLs constructed in this manner constitute the core of many phishing schemes, whereby an attacker convinces victims to visit a URL that refers to a vulnerable site. After the site reflects the attacker's content back to the user, the content is executed and proceeds to transfer private information, such as cookies that may include session information, from the user's machine to the attacker or perform other nefarious activities.

- As in Example 2, the application stores dangerous data in a database or other trusted data store. The dangerous data is subsequently read back into the application and included in dynamic content. Persistent XSS exploits occur when an attacker injects dangerous content into a data store that is later read and included in dynamic content. From an attacker's perspective, the optimal place to inject malicious content is in an area that is displayed to either many users or particularly interesting users. Interesting users typically have elevated privileges in the application or interact with sensitive data that is valuable to the attacker. If one of these users executes malicious content, the attacker may be able to perform privileged operations on behalf of the user or gain access to sensitive data belonging to the user.

- As in Example 3, a source outside the application stores dangerous data in a database or other data store, and the dangerous data is subsequently read back into the application as trusted data and included in dynamic content.

A number of modern web frameworks provide mechanisms for performing validation of user input. Struts and Spring MVC are two examples. To highlight the unvalidated sources of input, the rulepacks dynamically re-prioritize the

issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

Recommendation

The solution to XSS is to ensure that validation occurs in the correct places and checks for the correct properties.

Since XSS vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating dynamic content, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for XSS.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for XSS is generally relatively easy. Despite its value, input validation for XSS does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent XSS vulnerabilities is to validate everything that enters the application and leaves the application destined for the user.

The most secure approach to validation for XSS is to create a whitelist of safe characters that are allowed to appear in HTTP content and accept input composed exclusively of characters in the approved set. For example, a valid username might only include alpha-numeric characters or a phone number might only include digits 0-9. However, this solution is often infeasible in web applications because many characters that have special meaning to the browser should still be considered valid input once they are encoded, such as a web design bulletin board that must accept HTML fragments from its users.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning for web browsers. Although the HTML standard defines what characters have special meaning, many web browsers try to correct common mistakes in HTML and may treat other characters as special in certain contexts, which is why we do not encourage the use of blacklists as a means to prevent XSS. The CERT(R) Coordination Center at the Software Engineering Institute at Carnegie Mellon University provides the following details about special characters in various contexts [1]:

In the content of a block-level element (in the middle of a paragraph of text):

- "<" is special because it introduces a tag.

- "&" is special because it introduces a character entity.

- ">" is special because some browsers treat it as special, on the assumption that the author of the page intended to include an opening "<", but omitted it in error.

The following principles apply to attribute values:

- In attribute values enclosed with double quotes, the double quotes are special because they mark the end of the attribute value.

- In attribute values enclosed with single quote, the single quotes are special because they mark the end of the attribute value.

- In attribute values without any quotes, white-space characters, such as space and tab, are special.

- "&" is special when used with certain attributes, because it introduces a character entity.

In URLs, for example, a search engine might provide a link within the results page that the user can click to re-run the search. This can be implemented by encoding the search query inside the URL, which introduces additional special characters:

- Space, tab, and new line are special because they mark the end of the URL.

- "&" is special because it either introduces a character entity or separates CGI parameters.

- Non-ASCII characters (that is, everything above 128 in the ISO-8859-1 encoding) are not allowed in URLs, so they are considered to be special in this context.

- The "%" symbol must be filtered from input anywhere parameters encoded with HTTP escape sequences are decoded by server-side code. For example, "%" must be filtered if input such as "%68%65%6C%6C%6F" becomes "hello" when it appears on the web page in question.

Within the body of a <SCRIPT> </SCRIPT>:

- Semicolons, parentheses, curly braces, and new line characters should be filtered out in situations where text could be inserted directly into a pre-existing script tag.

Server-side scripts:

- Server-side scripts that convert any exclamation characters (!) in input to double-quote characters (") on output might require additional filtering.

Other possibilities:

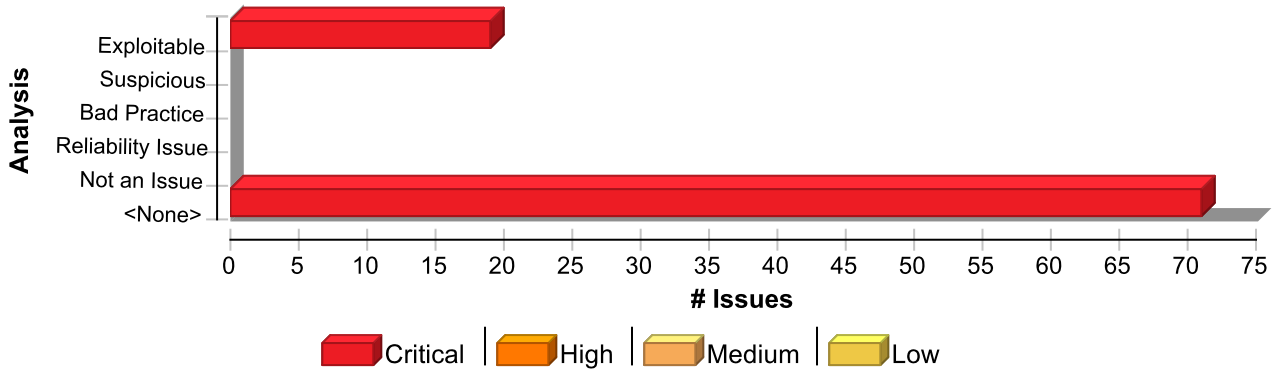
- If an attacker submits a request in UTF-7, the special character '<' appears as '+ADw-' and may bypass filtering. If the output is included in a page that does not explicitly specify an encoding format, then some browsers try to intelligently identify the encoding based on the content (in this case, UTF-7).

Once you identify the correct points in an application to perform validation for XSS attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. If special characters are not considered valid input to the application, then you can reject any input that contains special characters as invalid. A second option in this situation is to remove special characters with filtering. However, filtering has the side effect of changing any visual representation of the filtered content and may be unacceptable in circumstances where the integrity of the input must be preserved for display.

If input containing special characters must be accepted and displayed accurately, validation must encode any special characters to remove their significance. A complete list of ISO 8859-1 encoded values for special characters is provided as part of the official HTML specification [2].

Many application servers attempt to limit an application's exposure to cross-site scripting vulnerabilities by providing implementations for the functions responsible for setting certain specific HTTP response content that perform validation for the characters essential to a cross-site scripting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Cross-Site Scripting: Reflected	70	70	0	140
Total	**70**	**70**	**0**	**140**

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 163 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Indeterminate (Below Exploitable threshold)
AA_Training	Include

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.session.WebSession.getRestartLink
File: JavaSource/org/owasp/webgoat/session/WebSession.java:468

```
465  {
466  List<String> parameters = new ArrayList<String>();
467
468  String screenValue = request.getParameter(SCREEN);
469  if (screenValue != null)
470  parameters.add(SCREEN + "=" + screenValue);
471
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:163
Taint Flags: WEB, XSS

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 163 (Cross-Site Scripting: Reflected) — Critical

```
160 if (currentLesson != null)
161 {
162 %>
163 <div id="reset" class="info"><a href="<%=webSession.getRestartLink()%>">Restart this Lesson</a></div>
164 <%
165 }
166
```

main.jsp, line 163 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.session.WebSession.getRestartLink
File: JavaSource/org/owasp/webgoat/session/WebSession.java:472

```
469 if (screenValue != null)
470 parameters.add(SCREEN + "=" + screenValue);
471
472 String menuValue = request.getParameter(MENU);
473 if (menuValue != null)
474 parameters.add(MENU + "=" + menuValue);
475
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:163
Taint Flags: WEB, XSS

```
160 if (currentLesson != null)
161 {
162 %>
163 <div id="reset" class="info"><a href="<%=webSession.getRestartLink()%>">Restart this Lesson</a></div>
164 <%
165 }
166
```

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 180 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterNames()
From: org.owasp.webgoat.session.ParameterParser.getParameterNames
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:576

```
573  return (null);
574  }
575
576  return request.getParameterNames();
577  }
578
579
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.println()
Enclosing Method: _jspService()
File: main.jsp:180
Taint Flags: WEB, XSS

```
177  {
178  Parameter p = (Parameter) i.next();
179  printParameters = "<div id=\"parameter\" class=\"info\">" + p.getName() + "=" + p.getValue() + "</div><br>";
180  out.println(printParameters);
181  }
182  }
183
```

main.jsp, line 191 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 191 (Cross-Site Scripting: Reflected) — Critical

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: org.owasp.webgoat.session.WebSession.getCookies
File: JavaSource/org/owasp/webgoat/session/WebSession.java:596

```
593  List cookies = null;
594
595  if ( showCookies() )
596  cookies = Arrays.asList( request.getCookies() );
597
598  /*
599  * List cookies = new Vector();
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.println()
Enclosing Method: _jspService()
File: main.jsp:191
Taint Flags: WEB, XSS

```
188  {
189  Cookie c = (Cookie) i.next();
190  printCookies = "<div id=\"cookie\" class=\"info\">" + c.getName() + " <img src=\"images/icons/rightArrow.jpg\" alt=\"\"> " +
c.getValue() + "</div><br>";
191  out.println(printCookies);
192  }
193  }%>
194  <div id="lessonPlans" style="visibility:hidden; height:1px; position:absolute; left:260px; top:130px; width:425px; z-index:105;"><
%=currentLesson.getLessonPlan(webSession) %>
```

main.jsp, line 180 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getParameterValues
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:593

```
590  return (null);
591  }
592
```

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 180 (Cross-Site Scripting: Reflected) — Critical

```
593  return request.getParameterValues(name);
594  }
595
596
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.println()
Enclosing Method: _jspService()
File: main.jsp:180
Taint Flags: WEB, XSS

```
177  {
178  Parameter p = (Parameter) i.next();
179  printParameters = "<div id=\"parameter\" class=\"info\">" + p.getName() + "=" + p.getValue() + "</div><br>";
180  out.println(printParameters);
181  }
182  }
183
```

main.jsp, line 114 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /.main.jsp._jspService
File: WebContent/main.jsp:34

```
31 String printCookies = "";
32 String lessonComplete = "<img src=\"images/buttons/lessonComplete.jpg\">";
33 String m = "menu";
34 String menu = request.getParameter(m);
35
36 List categories = course.getCategories();
```

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 114 (Cross-Site Scripting: Reflected) — Critical

```
37
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:114
Taint Flags: WEB, XSS

```
111 if (webSession.isAuthorizedInLesson(webSession.getRole(), WebSession.SHOWHINTS))
112 {
113 %>
114 <a href="attack?show=PreviousHint&menu=<%=menu%>" target="_top" onclick="MM_nbGroup('down','group1','hintLeft','',1)"
115 onmouseover="MM_nbGroup('over','hintLeft','images/buttons/hintLeftOver.jpg','',1)"
116 onmouseout="MM_nbGroup('out')">
117 <img src="images/buttons/hintLeft.jpg" alt="Previous Hint" name="hintLeft" width="22" height="20" border="0" id="hintLeft"/>
```

main.jsp, line 119 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /.main.jsp._jspService
File: WebContent/main.jsp:34

```
31 String printCookies = "";
32 String lessonComplete = "<img src=\"images/buttons/lessonComplete.jpg\">";
33 String m = "menu";
34 String menu = request.getParameter(m);
35
36 List categories = course.getCategories();
37
```

Sink Details

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 119 (Cross-Site Scripting: Reflected) — Critical

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:119
Taint Flags: WEB, XSS

```
116 onmouseout="MM_nbGroup('out')">
117 <img src="images/buttons/hintLeft.jpg" alt="Previous Hint" name="hintLeft" width="22" height="20" border="0" id="hintLeft"/>
118 </a>
119 <a href="attack?show=NextHint&menu=<%=menu%>" target="_top" onclick="MM_nbGroup('down','group1','hint','',1)"
120 onmouseover="MM_nbGroup('over','hint','images/buttons/hintOver.jpg','',1)"
121 onmouseout="MM_nbGroup('out')">
122 <img src="images/buttons/hint.jpg" alt="Hints" name="hint" width="35" height="20" border="0" id="hint"/>
```

main.jsp, line 124 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /.main.jsp._jspService
File: WebContent/main.jsp:34

```
31 String printCookies = "";
32 String lessonComplete = "<img src=\"images/buttons/lessonComplete.jpg\">";
33 String m = "menu";
34 String menu = request.getParameter(m);
35
36 List categories = course.getCategories();
37
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:124
Taint Flags: WEB, XSS

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 124 (Cross-Site Scripting: Reflected) — Critical

```
121 onmouseout="MM_nbGroup('out')">
122 <img src="images/buttons/hint.jpg" alt="Hints" name="hint" width="35" height="20" border="0" id="hint"/>
123 </a>
124 <a href="attack?show=NextHint&menu=<%=menu%>" target="_top" onclick="MM_nbGroup('down','group1','hintRight','',1)"
125 onmouseover="MM_nbGroup('over','hintRight','images/buttons/hintRightOver.jpg','',1)"
126 onmouseout="MM_nbGroup('out')">
127 <img src="images/buttons/hintRight.jpg" alt="Next Hint" name="hintRight" width="20" height="20" border="0" id="hintRight"/>
```

main.jsp, line 130 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /.main.jsp._jspService
File: WebContent/main.jsp:34

```
31 String printCookies = "";
32 String lessonComplete = "<img src=\"images/buttons/lessonComplete.jpg\">";
33 String m = "menu";
34 String menu = request.getParameter(m);
35
36 List categories = course.getCategories();
37
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:130
Taint Flags: WEB, XSS

```
127 <img src="images/buttons/hintRight.jpg" alt="Next Hint" name="hintRight" width="20" height="20" border="0" id="hintRight"/>
128 </a>
129 <%}%>
```

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 130 (Cross-Site Scripting: Reflected) — Critical

```
130 <a href="attack?show=Params&menu=<%=menu%>" target="_top" onclick="MM_nbGroup('down','group1','params','',1)"
131 onmouseover="MM_nbGroup('over','params','images/buttons/paramsOver.jpg','',1)"
132 onmouseout="MM_nbGroup('out')">
133 <img src="images/buttons/params.jpg" alt="Show Params" name="attack?show=Params" width="92" height="20" border="0"
id="params"/>
```

main.jsp, line 135 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /.main.jsp._jspService
File: WebContent/main.jsp:34

```
31 String printCookies = "";
32 String lessonComplete = "<img src=\"images/buttons/lessonComplete.jpg\">";
33 String m = "menu";
34 String menu = request.getParameter(m);
35
36 List categories = course.getCategories();
37
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:135
Taint Flags: WEB, XSS

```
132 onmouseout="MM_nbGroup('out')">
133 <img src="images/buttons/params.jpg" alt="Show Params" name="attack?show=Params" width="92" height="20" border="0"
id="params"/>
134 </a>
135 <a href="attack?show=Cookies&menu=<%=menu%>" target="_top" onclick="MM_nbGroup('down','group1','cookies','',1)"
136 onmouseover="MM_nbGroup('over','cookies','images/buttons/cookiesOver.jpg','',1)"
```

Cross-Site Scripting: Reflected — Critical

Package: /

main.jsp, line 135 (Cross-Site Scripting: Reflected) — Critical

```
137  onmouseout="MM_nbGroup('out')">
138  <img src="images/buttons/cookies.jpg" alt="Show Cookies" name="cookies" width="104" height="20" border="0" id="cookies"/>
```

Package: /lessons/CrossSiteScripting

SearchStaff.jsp, line 11 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/CrossSiteScripting.SearchStaff.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/SearchStaff.jsp:7

```
4  <div id="lesson_search">
5  <%
6  WebSession webSession = ((WebSession)session.getAttribute("websession"));
7  String searchedName = request.getParameter(CrossSiteScripting.SEARCHNAME);
8  if (searchedName != null)
9  {
10  %>
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: SearchStaff.jsp:11
Taint Flags: WEB, XSS

```
8  if (searchedName != null)
9  {
10  %>
11  Employee <%=searchedName%> not found.
12  <%
13  }
14  %>
```

Cross-Site Scripting: Reflected — Critical

Package: /lessons/CrossSiteScripting

SearchStaff.jsp, line 11 (Cross-Site Scripting: Reflected) — Critical

ViewProfile.jsp, line 171 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.session.WebSession.getCurrentLink
File: JavaSource/org/owasp/webgoat/session/WebSession.java:521

```
518 {
519 thisLink += "&";
520 }
521 thisLink = thisLink + name + "=" + request.getParameter(name);
522 }
523
524 return thisLink;
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:171
Taint Flags: WEB, XSS

```
168 String thisPage = webSession.getCurrentLink();
169 //System.out.println("Redirecting to " + thisPage);
170 %>
171 <script language="javascript">location.href="<%=thisPage%>"</script>
172 <%
173 }
174 %>
```

ViewProfile.jsp, line 171 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Cross-Site Scripting: Reflected — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 171 (Cross-Site Scripting: Reflected) — Critical

AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterNames()
From: org.owasp.webgoat.session.WebSession.getCurrentLink
File: JavaSource/org/owasp/webgoat/session/WebSession.java:507

```
504 public String getCurrentLink()
505 {
506 String thisLink = "attack";
507 Enumeration e = request.getParameterNames();
508 boolean isFirstParameter = true;
509 while (e.hasMoreElements())
510 {
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:171
Taint Flags: WEB, XSS

```
168 String thisPage = webSession.getCurrentLink();
169 //System.out.println("Redirecting to " + thisPage);
170 %>
171 <script language="javascript">location.href="<%=thisPage%>"</script>
172 <%
173 }
174 %>
```

Package: /lessons/RoleBasedAccessControl

SearchStaff.jsp, line 11 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Cross-Site Scripting: Reflected — Critical

Package: /lessons/RoleBasedAccessControl

SearchStaff.jsp, line 11 (Cross-Site Scripting: Reflected) — Critical

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/RoleBasedAccessControl.SearchStaff.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/SearchStaff.jsp:7

```
4  <div id="lesson_search">
5  <%
6  WebSession webSession = ((WebSession)session.getAttribute("websession"));
7  String searchedName = request.getParameter(RoleBasedAccessControl.SEARCHNAME);
8  if (searchedName != null)
9  {
10  %>
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: SearchStaff.jsp:11
Taint Flags: WEB, XSS

```
8  if (searchedName != null)
9  {
10  %>
11  Employee <%=searchedName%> not found.
12  <%
13  }
14  %>
```

Package: /lessons/SQLInjection

SearchStaff.jsp, line 11 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameter()

Cross-Site Scripting: Reflected — Critical

Package: /lessons/SQLInjection

SearchStaff.jsp, line 11 (Cross-Site Scripting: Reflected) — Critical

From: /lessons/SQLInjection.SearchStaff.jsp._jspService
File: WebContent/lessons/SQLInjection/SearchStaff.jsp:7

```
4  <div id="lesson_search">
5  <%
6  WebSession webSession = ((WebSession)session.getAttribute("websession"));
7  String searchedName = request.getParameter(SQLInjection.SEARCHNAME);
8  if (searchedName != null)
9  {
10  %>
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: SearchStaff.jsp:11
Taint Flags: WEB, XSS

```
8  if (searchedName != null)
9  {
10  %>
11  Employee <%=searchedName%> not found.
12  <%
13  }
14  %>
```

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 156 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 156 (Cross-Site Scripting: Reflected) — Critical

```
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:156
Taint Flags: WEB, XSS

```
153  TextArea input3 = new TextArea("field3", 1, 25).addElement(param3);
154  TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
155  TextArea input5 = new TextArea("field5", 1, 25).addElement(param5);
156  TextArea input6 = new TextArea("field6", 1, 25).addElement(param6);
157  TextArea input7 = new TextArea("field7", 1, 25).addElement(param7);
158
159  Input b = new Input();
```

ReflectedXSS.java, line 165 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ReflectedXSS.java:165

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

ReflectedXSS.java, line 165 (Cross-Site Scripting: Reflected) — Critical

Taint Flags: WEB, XSS

```
162
163 tr.addElement(new TD().addElement(
164 new Input(Input.TEXT, "QTY4", s.getParser()
165 .getStringParameter("QTY4", "1")))
166 .setAlign("right"));
167 quantity = s.getParser().getFloatParameter("QTY4", 1.0f);
168 total = quantity * 299.99f;
```

WeakAuthenticationCookie.java, line 377 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.P.addElement()
Enclosing Method: makeUser()
File: WeakAuthenticationCookie.java:377
Taint Flags: WEB, XSS

```
374 throws Exception
375 {
376 ElementContainer ec = new ElementContainer();
377 ec.addElement(new P().addElement("Welcome, " + user));
378 ec.addElement(new P().addElement("You have been authenticated with "
379 + method));
380 ec.addElement(new P().addElement(ECSFactory.makeLink("Logout", LOGOUT,
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 377 (Cross-Site Scripting: Reflected) — Critical

JavaScriptValidation.java, line 155 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:155
Taint Flags: WEB, XSS

```
152  TextArea input2 = new TextArea("field2", 1, 25).addElement(param2);
153  TextArea input3 = new TextArea("field3", 1, 25).addElement(param3);
154  TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
155  TextArea input5 = new TextArea("field5", 1, 25).addElement(param5);
156  TextArea input6 = new TextArea("field6", 1, 25).addElement(param6);
157  TextArea input7 = new TextArea("field7", 1, 25).addElement(param7);
158
```

JavaScriptValidation.java, line 152 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 152 (Cross-Site Scripting: Reflected) — Critical

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:152
Taint Flags: WEB, XSS

```
149 "301-604-4882");
150 ec.addElement(new StringElement(script));
151 TextArea input1 = new TextArea("field1", 1, 25).addElement(param1);
152 TextArea input2 = new TextArea("field2", 1, 25).addElement(param2);
153 TextArea input3 = new TextArea("field3", 1, 25).addElement(param3);
154 TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
155 TextArea input5 = new TextArea("field5", 1, 25).addElement(param5);
```

ReflectedXSS.java, line 150 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

ReflectedXSS.java, line 150 (Cross-Site Scripting: Reflected) — Critical

```
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ReflectedXSS.java:150
Taint Flags: WEB, XSS

```
147  tr.addElement(new TD().addElement("1599.99").setAlign("right"));
148  tr.addElement(new TD().addElement(
149  new Input(Input.TEXT, "QTY3", s.getParser()
150  .getStringParameter("QTY3", "1")))
151  .setAlign("right"));
152  quantity = s.getParser().getFloatParameter("QTY3", 1.0f);
153  total = quantity * 1599.99f;
```

JavaScriptValidation.java, line 154 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 154 (Cross-Site Scripting: Reflected) — Critical

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:154
Taint Flags: WEB, XSS

```
151 TextArea input1 = new TextArea("field1", 1, 25).addElement(param1);
152 TextArea input2 = new TextArea("field2", 1, 25).addElement(param2);
153 TextArea input3 = new TextArea("field3", 1, 25).addElement(param3);
154 TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
155 TextArea input5 = new TextArea("field5", 1, 25).addElement(param5);
156 TextArea input6 = new TextArea("field6", 1, 25).addElement(param6);
157 TextArea input7 = new TextArea("field7", 1, 25).addElement(param7);
```

Encoding.java, line 369 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: Encoding.java:369
Taint Flags: PRIMARY_KEY, WEB, XSS

```
366
367 tr.addElement( new TD( "Enter a password (optional): " ) );
368
369 Input key = new Input( Input.TEXT, KEY, userKey );
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 369 (Cross-Site Scripting: Reflected) — Critical

```
370
371  tr.addElement( new TD().addElement( key ) );
372
```

ReflectedXSS.java, line 206 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ReflectedXSS.java:206
Taint Flags: WEB, XSS

```
203  tr.addElement(new TD()
204  .addElement("Enter your three digit access code:"));
205  tr.addElement(new TD().addElement(new Input(Input.TEXT, "field1",
206  param1)));
207  t.addElement(tr);
208
209  Element b = ECSFactory.makeButton("Purchase");
```

WeakSessionID.java, line 262 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 262 (Cross-Site Scripting: Reflected) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: org.owasp.webgoat.session.WebSession.getCookie
File: JavaSource/org/owasp/webgoat/session/WebSession.java:621

```
618 */
619 public String getCookie( String cookieName )
620 {
621 Cookie[] cookies = getRequest().getCookies();
622
623 for ( int i = 0; i < cookies.length; i++ )
624 {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: makeLogin()
File: WeakSessionID.java:262
Taint Flags: PRIMARY_KEY, WEB, XSS

```
259
260 Input input1 = new Input(Input.TEXT, USERNAME, "");
261 Input input2 = new Input(Input.PASSWORD, PASSWORD, "");
262 Input input3 = new Input(Input.HIDDEN, SESSIONID, weakid);
263 row1.addElement(new TD(input1));
264 row2.addElement(new TD(input2));
265 t.addElement(row1);
```

JavaScriptValidation.java, line 157 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 157 (Cross-Site Scripting: Reflected) — Critical

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:157
Taint Flags: WEB, XSS

```
154  TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
155  TextArea input5 = new TextArea("field5", 1, 25).addElement(param5);
156  TextArea input6 = new TextArea("field6", 1, 25).addElement(param6);
157  TextArea input7 = new TextArea("field7", 1, 25).addElement(param7);
158
159  Input b = new Input();
160  b.setType(Input.BUTTON);
```

TraceXSS.java, line 137 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 137 (Cross-Site Scripting: Reflected) — Critical

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: TraceXSS.java:137
Taint Flags: WEB, XSS

```
134 tr.addElement(new TD().addElement("27.99").setAlign("right"));
135 tr.addElement(new TD().addElement(
136 new Input(Input.TEXT, "QTY2", s.getParser()
137 .getStringParameter("QTY2", "1")))
138 .setAlign("right"));
139 quantity = s.getParser().getFloatParameter("QTY2", 1.0f);
140 total = quantity * 27.99f;
```

BlindSqlInjection.java, line 83 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

BlindSqlInjection.java, line 83 (Cross-Site Scripting: Reflected) — Critical

File: BlindSqlInjection.java:83
Taint Flags: PRIMARY_KEY, WEB, XSS

```
80  String accountNumber = s.getParser().getRawParameter(ACCT_NUM,
81  "101");
82  Input input = new Input(Input.TEXT, ACCT_NUM, accountNumber
83  .toString());
84  ec.addElement(input);
85
86  Element b = ECSFactory.makeButton("Go!");
```

AbstractLesson.java, line 920 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getReader()
From: org.owasp.webgoat.lessons.AbstractLesson.makeRequestDump_DELETEME
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:900

```
897
898  try
899  {
900  el = new StringElement(readFromFile(s.getRequest().getReader(),
901  false));
902  }
903  catch (Exception e)
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: makeRequestDump_DELETEME()
File: AbstractLesson.java:920
Taint Flags: NO_NEW_LINE, WEB, XSS

```
917  }
918
919  t.addElement(new TR().addElement(new TD().setVAlign("TOP").addElement(
920  ec)));
921
922  return (t);
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 920 (Cross-Site Scripting: Reflected) — Critical

```
923 }
```

ReflectedXSS.java, line 123 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ReflectedXSS.java:123
Taint Flags: WEB, XSS

```
120 tr.addElement(new TD().addElement("69.99").setAlign("right"));
121 tr.addElement(new TD().addElement(
122 new Input(Input.TEXT, "QTY1", s.getParser()
123 .getStringParameter("QTY1", "1")))
124 .setAlign("right"));
125 quantity = s.getParser().getFloatParameter("QTY1", 1.0f);
126 total = quantity * 69.99f;
```

BackDoors.java, line 235 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 235 (Cross-Site Scripting: Reflected) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.Div.Div()
Enclosing Method: makeUsername()
File: BackDoors.java:235
Taint Flags: WEB, XSS

```
232
233 String formattedInput = "<span class='myClass'>" + userInput
234 + "</span>";
235 ec.addElement(new Div(SELECT_ST + formattedInput));
236
237 Input b = new Input();
238
```

HttpOnly.java, line 233 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.session.WebSession.getHeader
File: JavaSource/org/owasp/webgoat/session/WebSession.java:1145

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

HttpOnly.java, line 233 (Cross-Site Scripting: Reflected) — Critical

```
1142  */
1143  public String getHeader( String header )
1144  {
1145  return getRequest().getHeader( header );
1146  }
1147
1148  public String getNextHint()
```

Sink Details

Sink: org.apache.ecs.html.TD.TD()
Enclosing Method: makeContent()
File: HttpOnly.java:233
Taint Flags: WEB, XSS

```
230
231  tr = new TR();
232
233  tr.addElement(new TD(new StringElement("Your browser appears to be: " + getBrowserType(s))));
234  t.addElement(tr);
235
236  tr = new TR();
```

TraceXSS.java, line 166 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 166 (Cross-Site Scripting: Reflected) — Critical

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: TraceXSS.java:166
Taint Flags: WEB, XSS

```
163
164 tr.addElement(new TD().addElement(
165 new Input(Input.TEXT, "QTY4", s.getParser()
166 .getStringParameter("QTY4", "1")))
167 .setAlign("right"));
168 quantity = s.getParser().getFloatParameter("QTY4", 1.0f);
169 total = quantity * 299.99f;
```

SqlNumericInjection.java, line 115 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.PRE.PRE()
Enclosing Method: injectableQuery()
File: SqlNumericInjection.java:115
Taint Flags: WEB, XSS

```
112 query = "SELECT * FROM weather_data WHERE station = " + station;
113 }
114
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 115 (Cross-Site Scripting: Reflected) — Critical

```
115 ec.addElement(new PRE(query));
116
117 if (station == null)
118 return ec;
```

JavaScriptValidation.java, line 151 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:151
Taint Flags: WEB, XSS

```
148 String param7 = s.getParser().getRawParameter("field7",
149 "301-604-4882");
150 ec.addElement(new StringElement(script));
151 TextArea input1 = new TextArea("field1", 1, 25).addElement(param1);
152 TextArea input2 = new TextArea("field2", 1, 25).addElement(param2);
153 TextArea input3 = new TextArea("field3", 1, 25).addElement(param3);
154 TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
```

TraceXSS.java, line 124 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 124 (Cross-Site Scripting: Reflected) — Critical

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: TraceXSS.java:124
Taint Flags: WEB, XSS

```
121 tr.addElement(new TD().addElement("69.99").setAlign("right"));
122 tr.addElement(new TD().addElement(
123 new Input(Input.TEXT, "QTY1", s.getParser()
124 .getStringParameter("QTY1", "1")))
125 .setAlign("right"));
126 quantity = s.getParser().getFloatParameter("QTY1", 1.0f);
127 total = quantity * 69.99f;
```

ReflectedXSS.java, line 200 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

ReflectedXSS.java, line 200 (Cross-Site Scripting: Reflected) — Critical

From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ReflectedXSS.java:200
Taint Flags: WEB, XSS

```
197  .addElement(new TD()
198  .addElement("Enter your credit card number:"));
199  tr.addElement(new TD().addElement(new Input(Input.TEXT, "field2",
200  param2)));
201  t.addElement(tr);
202  tr = new TR();
203  tr.addElement(new TD()
```

AbstractLesson.java, line 1086 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 1086 (Cross-Site Scripting: Reflected) — Critical

```
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.Form.addElement()
Enclosing Method: handleRequest()
File: AbstractLesson.java:1086
Taint Flags: CANONICAL_PATH, TAINTED_PATH, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
1083  Form form = new Form(getFormAction(), Form.POST).setName("form")
1084  .setEncType("");
1085
1086  form.addElement(createContent(s));
1087
1088  setContent(form);
1089  }
```

TraceXSS.java, line 207 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: TraceXSS.java:207

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 207 (Cross-Site Scripting: Reflected) — Critical

Taint Flags: WEB, XSS

```
204 tr.addElement(new TD()
205 .addElement("Enter your three digit access code:"));
206 tr.addElement(new TD().addElement(new Input(Input.TEXT, "field1",
207 param1)));
208 t.addElement(tr);
209
210 Element b = ECSFactory.makeButton("Purchase");
```

Encoding.java, line 359 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: Encoding.java:359
Taint Flags: WEB, XSS

```
356
357 tr.addElement( new TD( "Enter a string: " ) );
358
359 Input input = new Input( Input.TEXT, INPUT, userInput );
360
361 tr.addElement( new TD().addElement( input ) );
362
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 359 (Cross-Site Scripting: Reflected) — Critical

Encoding.java, line 794 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: makeRow()
File: Encoding.java:794
Taint Flags: WEAKCRYPTO, WEB, XSS

```
791
792  TD desc = new TD().addElement( description ).setBgColor( "#bbbbbb" );
793  TD val1 = new TD().addElement( value1 ).setBgColor( "#dddddd" );
794  TD val2 = new TD().addElement( value2 ).setBgColor( "#dddddd" );
795  TR tr = new TR();
796
797  tr.addElement( desc );
```

WSDLScanning.java, line 219 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 219 (Cross-Site Scripting: Reflected) — Critical

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getParameterValues
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:593

```
590  return (null);
591  }
592
593  return request.getParameterValues(name);
594  }
595
596
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: createContent()
File: WSDLScanning.java:219
Taint Flags: WEB, XSS

```
216  TR results = new TR();
217  for (int i = 0; i < fields.length; i++)
218  {
219  header.addElement(new TD().addElement(fields[i]));
220  results.addElement(new TD()
221  .addElement((String) accessWGService("WSDLScanning",
222  fields[i], "acct_num", new Integer(id))));
```

SilentTransactions.java, line 94 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

SilentTransactions.java, line 94 (Cross-Site Scripting: Reflected) — Critical

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.io.PrintWriter.print()
Enclosing Method: handleRequest()
File: SilentTransactions.java:94
Taint Flags: WEB, XSS

```
91  .append("Now you can send out a spam email containing this link and whoever clicks on it<br>");
92  result
93  .append(" and happens to be logged in the same time will loose their money !!");
94  out.print(result.toString());
95  out.flush();
96  out.close();
97  getLessonTracker(s).setCompleted(true);
```

HttpSplitting.java, line 180 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

HttpSplitting.java, line 180 (Cross-Site Scripting: Reflected) — Critical

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createAttackEnvironment()
File: HttpSplitting.java:180
Taint Flags: WEB, XSS

```
177 "UTF-8");
178
179 //add the search by field
180 Input input = new Input(Input.TEXT, LANGUAGE, lang.toString());
181 ec.addElement(input);
182
183 Element b = ECSFactory.makeButton("Search!");
```

UncheckedEmail.java, line 135 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

UncheckedEmail.java, line 135 (Cross-Site Scripting: Reflected) — Critical

```
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: UncheckedEmail.java:135
Taint Flags: VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
132  tr = new TR();
133  String message = s.getParser().getRawParameter(MESSAGE, "");
134  TextArea ta = new TextArea(MESSAGE, 5, 40);
135  ta.addElement(new StringElement(convertMetachars(message)));
136  tr.addElement(new TD().setAlign("LEFT").addElement(ta));
137  tr.addElement(new TD().setAlign("LEFT").setVAlign("MIDDLE")
138  .addElement(ECSFactory.makeButton("Send!")));
```

SqlStringInjection.java, line 235 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 235 (Cross-Site Scripting: Reflected) — Critical

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: makeAccountLine()
File: SqlStringInjection.java:235
Taint Flags: WEB, XSS

```
232 ec.addElement(new P().addElement("Enter your last name: "));
233
234 accountName = s.getParser().getRawParameter(ACCT_NAME, "Your Name");
235 Input input = new Input(Input.TEXT, ACCT_NAME, accountName.toString());
236 ec.addElement(input);
237
238 Element b = ECSFactory.makeButton("Go!");
```

HttpBasics.java, line 69 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: HttpBasics.java:69
Taint Flags: WEB, XSS

```
66 ""));
67 person.reverse();
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

HttpBasics.java, line 69 (Cross-Site Scripting: Reflected) — Critical

```
68
69  Input input = new Input(Input.TEXT, PERSON, person.toString());
70  ec.addElement(input);
71
72  Element b = ECSFactory.makeButton("Go!");
```

UncheckedEmail.java, line 158 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.B.addElement()
Enclosing Method: createContent()
File: UncheckedEmail.java:158
Taint Flags: WEB, XSS

```
155  ec.addElement(new HR());
156  ec
157  .addElement(new Center()
158  .addElement(new B()
159  .addElement("You sent the following message to: "
160  + to)));
161  ec.addElement(new BR());
```

HttpSplitting.java, line 112 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

HttpSplitting.java, line 112 (Cross-Site Scripting: Reflected) — Critical

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.io.PrintWriter.print()
Enclosing Method: doHTTPSplitting()
File: HttpSplitting.java:112
Taint Flags: WEB, XSS

```
109 PrintWriter out = new PrintWriter(res.getOutputStream());
110 String message = lang.substring(lang.indexOf("<html>"));
111
112 out.print(message);
113 out.flush();
114 out.close();
115
```

ReflectedXSS.java, line 136 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

ReflectedXSS.java, line 136 (Cross-Site Scripting: Reflected) — Critical

From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ReflectedXSS.java:136
Taint Flags: WEB, XSS

```
133  tr.addElement(new TD().addElement("27.99").setAlign("right"));
134  tr.addElement(new TD().addElement(
135  new Input(Input.TEXT, "QTY2", s.getParser()
136  .getStringParameter("QTY2", "1")))
137  .setAlign("right"));
138  quantity = s.getParser().getFloatParameter("QTY2", 1.0f);
139  total = quantity * 27.99f;
```

Encoding.java, line 793 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 793 (Cross-Site Scripting: Reflected) — Critical

```
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: makeRow()
File: Encoding.java:793
Taint Flags: WEB, XSS

```
790  {
791
792  TD desc = new TD().addElement( description ).setBgColor( "#bbbbbb" );
793  TD val1 = new TD().addElement( value1 ).setBgColor( "#dddddd" );
794  TD val2 = new TD().addElement( value2 ).setBgColor( "#dddddd" );
795  TR tr = new TR();
796
```

LogSpoofing.java, line 110 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

LogSpoofing.java, line 110 (Cross-Site Scripting: Reflected) — Critical

```
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.PRE.PRE()
Enclosing Method: createContent()
File: LogSpoofing.java:110
Taint Flags: WEB, XSS

```
107  .setBorder(0);
108  TR row4 = new TR();
109  row4.addElement(
110  new TD(new PRE("Login failed for username: "
111  + inputUsername))).setBgColor(HtmlColor.GRAY);
112
113  t2.addElement(row4);
```

SqlStringInjection.java, line 105 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.PRE.PRE()
Enclosing Method: injectableQuery()
File: SqlStringInjection.java:105

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 105 (Cross-Site Scripting: Reflected) — Critical

Taint Flags: WEB, XSS

```
102
103 String query = "SELECT * FROM user_data WHERE last_name = '"
104 + accountName + "'";
105 ec.addElement(new PRE(query));
106
107 try
108 {
```

Encoding.java, line 794 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: makeRow()
File: Encoding.java:794
Taint Flags: PRIMARY_KEY, WEB, XSS

```
791
792 TD desc = new TD().addElement( description ).setBgColor( "#bbbbbb" );
793 TD val1 = new TD().addElement( value1 ).setBgColor( "#dddddd" );
794 TD val2 = new TD().addElement( value2 ).setBgColor( "#dddddd" );
795 TR tr = new TR();
796
797 tr.addElement( desc );
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 794 (Cross-Site Scripting: Reflected) — Critical

WeakAuthenticationCookie.java, line 377 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.P.addElement()
Enclosing Method: makeUser()
File: WeakAuthenticationCookie.java:377
Taint Flags: WEB, XSS

```
374 throws Exception
375 {
376 ElementContainer ec = new ElementContainer();
377 ec.addElement(new P().addElement("Welcome, " + user));
378 ec.addElement(new P().addElement("You have been authenticated with "
379 + method));
380 ec.addElement(new P().addElement(ECSFactory.makeLink("Logout", LOGOUT,
```

BasicAuthentication.java, line 143 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

BasicAuthentication.java, line 143 (Cross-Site Scripting: Reflected) — Critical

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: doStage1()
File: BasicAuthentication.java:143
Taint Flags: WEB, XSS

```
140 "What is the decoded value of the authentication header: ")));
141
142 row1.addElement(new TD(new Input(Input.TEXT, HEADER_NAME,
143 headerName.toString())));
144 row2.addElement(new TD(new Input(Input.TEXT, HEADER_VALUE,
145 headerValue.toString())));
146
```

BasicAuthentication.java, line 145 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

BasicAuthentication.java, line 145 (Cross-Site Scripting: Reflected) — Critical

```
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: doStage1()
File: BasicAuthentication.java:145
Taint Flags: WEB, XSS

```
142  row1.addElement(new TD(new Input(Input.TEXT, HEADER_NAME,
143  headerName.toString())));
144  row2.addElement(new TD(new Input(Input.TEXT, HEADER_VALUE,
145  headerValue.toString())));
146
147  t.addElement(row1);
148  t.addElement(row2);
```

AbstractLesson.java, line 872 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterNames()
From: org.owasp.webgoat.session.ParameterParser.getParameterNames
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:576

```
573  return (null);
574  }
575
576  return request.getParameterNames();
577  }
578
579
```

Sink Details

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 872 (Cross-Site Scripting: Reflected) — Critical

Sink: org.apache.ecs.html.LI.LI()
Enclosing Method: makeParamDump_DELETEME()
File: AbstractLesson.java:872
Taint Flags: WEB, XSS

```
869 while (i.hasNext())
870 {
871 String str = (String) i.next();
872 list.addElement(new LI(str));
873 }
874
875 ElementContainer ec = new ElementContainer();
```

TraceXSS.java, line 151 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: TraceXSS.java:151
Taint Flags: WEB, XSS

```
148 tr.addElement(new TD().addElement("1599.99").setAlign("right"));
149 tr.addElement(new TD().addElement(
150 new Input(Input.TEXT, "QTY3", s.getParser()
151 .getStringParameter("QTY3", "1")))
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 151 (Cross-Site Scripting: Reflected) — Critical

```
152  .setAlign("right"));
153  quantity = s.getParser().getFloatParameter("QTY3", 1.0f);
154  total = quantity * 1599.99f;
```

JavaScriptValidation.java, line 153 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.TextArea.addElement()
Enclosing Method: createContent()
File: JavaScriptValidation.java:153
Taint Flags: WEB, XSS

```
150  ec.addElement(new StringElement(script));
151  TextArea input1 = new TextArea("field1", 1, 25).addElement(param1);
152  TextArea input2 = new TextArea("field2", 1, 25).addElement(param2);
153  TextArea input3 = new TextArea("field3", 1, 25).addElement(param3);
154  TextArea input4 = new TextArea("field4", 1, 25).addElement(param4);
155  TextArea input5 = new TextArea("field5", 1, 25).addElement(param5);
156  TextArea input6 = new TextArea("field6", 1, 25).addElement(param6);
```

ForgotPassword.java, line 191 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

ForgotPassword.java, line 191 (Cross-Site Scripting: Reflected) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: doStage3()
File: ForgotPassword.java:191
Taint Flags: WEB, XSS

```
188 t.addElement(tr);
189
190 tr = new TR();
191 tr.addElement(new TD().addElement(new StringElement("Username: " + USERNAME_RESPONSE)));
192 t.addElement(tr);
193
194 tr = new TR();
```

ForgotPassword.java, line 195 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

ForgotPassword.java, line 195 (Cross-Site Scripting: Reflected) — Critical

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: org.apache.ecs.html.TD.addElement()
Enclosing Method: doStage3()
File: ForgotPassword.java:195
Taint Flags: WEB, XSS

```
192  t.addElement(tr);
193
194  tr = new TR();
195  tr.addElement(new TD().addElement(new StringElement("Color: " + COLOR_RESPONSE)));
196  t.addElement(tr);
197
198  tr = new TR();
```

WsSqlInjection.java, line 175 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 175 (Cross-Site Scripting: Reflected) — Critical

Sink Details

Sink: org.apache.ecs.html.PRE.PRE()
Enclosing Method: createContent()
File: WsSqlInjection.java:175
Taint Flags: PRIMARY_KEY, WEB, XSS

```
172
173 String query = "SELECT * FROM user_data WHERE userid = "
174 + accountNumber;
175 ec.addElement(new PRE(query));
176 for (int i = 0; i < accountNumber.length(); i++)
177 {
178 char c = accountNumber.charAt(i);
```

AbstractLesson.java, line 872 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getParameterValues
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:593

```
590 return (null);
591 }
592
593 return request.getParameterValues(name);
594 }
595
596
```

Sink Details

Sink: org.apache.ecs.html.LI.LI()
Enclosing Method: makeParamDump_DELETEME()
File: AbstractLesson.java:872
Taint Flags: WEB, XSS

```
869 while (i.hasNext())
870 {
871 String str = (String) i.next();
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 872 (Cross-Site Scripting: Reflected) — Critical

```
872 list.addElement(new LI(str));
873 }
874
875 ElementContainer ec = new ElementContainer();
```

TraceXSS.java, line 201 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: TraceXSS.java:201
Taint Flags: WEB, XSS

```
198 .addElement(new TD()
199 .addElement("Enter your credit card number:"));
200 tr.addElement(new TD().addElement(new Input(Input.TEXT, "field2",
201 param2)));
202 t.addElement(tr);
203 tr = new TR();
204 tr.addElement(new TD()
```

WsSqlInjection.java, line 152 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 152 (Cross-Site Scripting: Reflected) — Critical

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: makeAccountLine()
File: WsSqlInjection.java:152
Taint Flags: PRIMARY_KEY, WEB, XSS

```
149  ec.addElement(new P().addElement("Enter your Account Number: "));
150
151  accountNumber = s.getParser().getRawParameter(ACCT_NUM, "101");
152  Input input = new Input(Input.TEXT, ACCT_NUM, accountNumber.toString());
153  ec.addElement(input);
154
155  Element b = ECSFactory.makeButton("Go!");
```

WsSAXInjection.java, line 155 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 155 (Cross-Site Scripting: Reflected) — Critical

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.apache.ecs.html.PRE.addElement()
Enclosing Method: createContent()
File: WsSAXInjection.java:155
Taint Flags: WEB, XSS

```
152 String xml = template1;
153 xml = xml + (password == null ? "[password]" : password);
154 xml = xml + template2;
155 pre.addElement(HtmlEncoder.encode(xml));
156 ec.addElement(pre);
157
158 if (password != null)
```

Package: org.owasp.webgoat.lessons.admin

ReportCardScreen.java, line 295 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons.admin

ReportCardScreen.java, line 295 (Cross-Site Scripting: Reflected) — Critical

Audit Comments

[AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterNames()
From: org.owasp.webgoat.session.ParameterParser.getParameterNames
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:576

```
573  return (null);
574  }
575
576  return request.getParameterNames();
577  }
578
579
```

Sink Details

Sink: org.apache.ecs.html.H2.addElement()
Enclosing Method: makeUser()
File: ReportCardScreen.java:295
Taint Flags: PRIMARY_KEY, WEB, XSS

```
292  // FIXME: The session is the current session, not the session of the user we are reporting.
293  //String type = s.isAdmin() ? " [Administrative User]" : s.isHackedAdmin() ? " [Normal User - Hacked Admin Access]" : " [Normal User]";
294  String type = "";
295  h2.addElement(new StringElement("Results for: " + user + type));
296  return h2;
297  }
298
```

ReportCardScreen.java, line 295 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
```

Cross-Site Scripting: Reflected — Critical

Package: org.owasp.webgoat.lessons.admin

ReportCardScreen.java, line 295 (Cross-Site Scripting: Reflected) — Critical

```
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.H2.addElement()
Enclosing Method: makeUser()
File: ReportCardScreen.java:295
Taint Flags: WEB, XSS

```
292  // FIXME: The session is the current session, not the session of the user we are reporting.
293  //String type = s.isAdmin() ? " [Administrative User]" : s.isHackedAdmin() ? " [Normal User - Hacked Admin Access]" : " [Normal User]";
294  String type = "";
295  h2.addElement(new StringElement("Results for: " + user + type));
296  return h2;
297  }
298
```

ViewDatabase.java, line 72 (Cross-Site Scripting: Reflected) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
```

Cross-Site Scripting: Reflected	Critical
Package: org.owasp.webgoat.lessons.admin	
ViewDatabase.java, line 72 (Cross-Site Scripting: Reflected)	Critical

```
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.apache.ecs.html.Input.Input()
Enclosing Method: createContent()
File: ViewDatabase.java:72
Taint Flags: WEB, XSS

```
69
70  StringBuffer sqlStatement = new StringBuffer(s.getParser()
71  .getRawParameter(SQL, ""));
72  Input input = new Input(Input.TEXT, SQL, sqlStatement.toString());
73  ec.addElement(input);
74
75  Element b = ECSFactory.makeButton("Go!");
```

Dead Code: Expression is Always false (3 issues)

Abstract

This expression (or part of it) will always evaluate to `false`.

Explanation

This expression (or part of it) will always evaluate to `false`; the program could be rewritten in a simpler form. The nearby code may be present for debugging purposes, or it may not have been maintained along with the rest of the program. The expression may also be indicative of a bug earlier in the method.

Example 1: The following method never sets the variable `secondCall` after initializing it to `false`. (The variable `firstCall` is mistakenly used twice.) The result is that the expression `firstCall && secondCall` will always evaluate to `false`, so `setUpDualCall()` will never be invoked.

```
public void setUpCalls() {
  boolean firstCall = false;
  boolean secondCall = false;

  if (fCall > 0) {
    setUpFCall();
    firstCall = true;
  }
  if (sCall > 0) {
    setUpSCall();
    firstCall = true;
  }

  if (firstCall && secondCall) {
    setUpDualCall();
  }
}
```

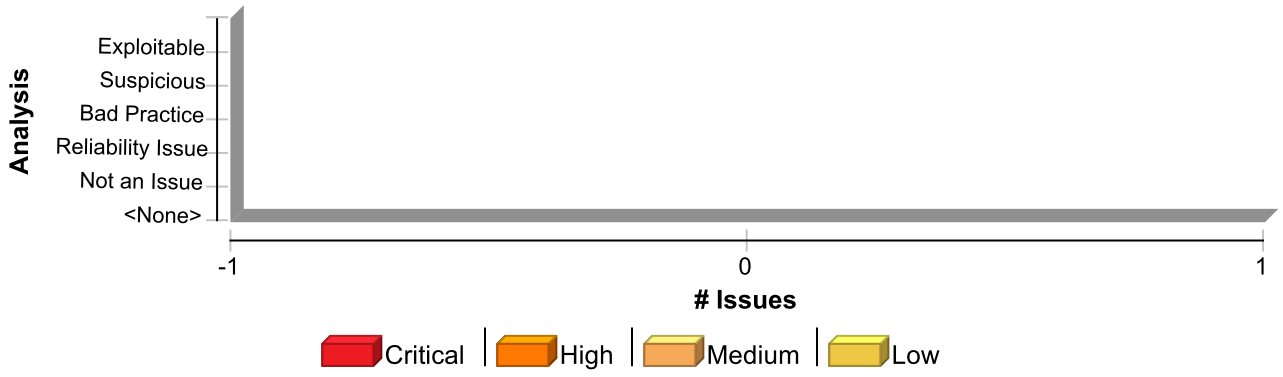
Example 2: The following method never sets the variable `firstCall` to `true`. (The variable `firstCall` is mistakenly set to `false` after the first conditional statement.) The result is that the first part of the expression `firstCall && secondCall` will always evaluate to `false`.

```
public void setUpCalls() {
  boolean firstCall = false;
  boolean secondCall = false;

  if (fCall > 0) {
    setUpFCall();
    firstCall = false;
  }
  if (sCall > 0) {
    setUpSCall();
    secondCall = true;
  }

  if (firstCall || secondCall) {
    setUpForCall();
  }
}
```

Recommendation

In general, you should repair or remove unused code. It causes additional complexity and maintenance burden without contributing to the functionality of the program.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Dead Code: Expression is Always false	3	3	0	6
Total	**3**	**3**	**0**	**6**

Dead Code: Expression is Always false — Low

Package: org.owasp.webgoat.util

Exec.java, line 156 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: IfStatement
Enclosing Method: execOptions()
File: Exec.java:156

```
153 // end of stream
154 output.write(c);
155
156 if (lazy && (processIn.available() < 1))
157 {
158 lazyQuit = true;
159 }
```

Exec.java, line 202 (Dead Code: Expression is Always false) — Low

Issue Details

Dead Code: Expression is Always false — Low

Package: org.owasp.webgoat.util

Exec.java, line 202 (Dead Code: Expression is Always false) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: IfStatement
Enclosing Method: execOptions()
File: Exec.java:202

```
199 // end of stream
200 output.write(c);
201
202 if (lazy && (processError.available() < 1))
203 {
204 lazyQuit = true;
205 }
```

Exec.java, line 111 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: IfStatement
Enclosing Method: execOptions()
File: Exec.java:111

```
108 OutputStream processOut = child.getOutputStream();
109
110 // start the clock running
111 if (timeout > 0)
112 {
113 watcher = new ThreadWatcher(child, interrupted, timeout);
114 new Thread(watcher).start();
```

Dead Code: Expression is Always true (1 issue)

Abstract

This expression (or part of it) will always evaluate to `true`.

Explanation

This expression (or part of it) will always evaluate to `true`; the program could be rewritten in a simpler form. The nearby code may be present for debugging purposes, or it may not have been maintained along with the rest of the program. The expression may also be indicative of a bug earlier in the method.

Example 1: The following method never sets the variable `secondCall` after initializing it to `true`. (The variable `firstCall` is mistakenly used twice.) The result is that the expression `firstCall || secondCall` will always evaluate to `true`, so `setUpForCall()` will always be invoked.

```
public void setUpCalls() {
  boolean firstCall = true;
  boolean secondCall = true;

  if (fCall < 0) {
    cancelFCall();
    firstCall = false;
  }
  if (sCall < 0) {
    cancelSCall();
    firstCall = false;
  }

  if (firstCall || secondCall) {
    setUpForCall();
  }
}
```

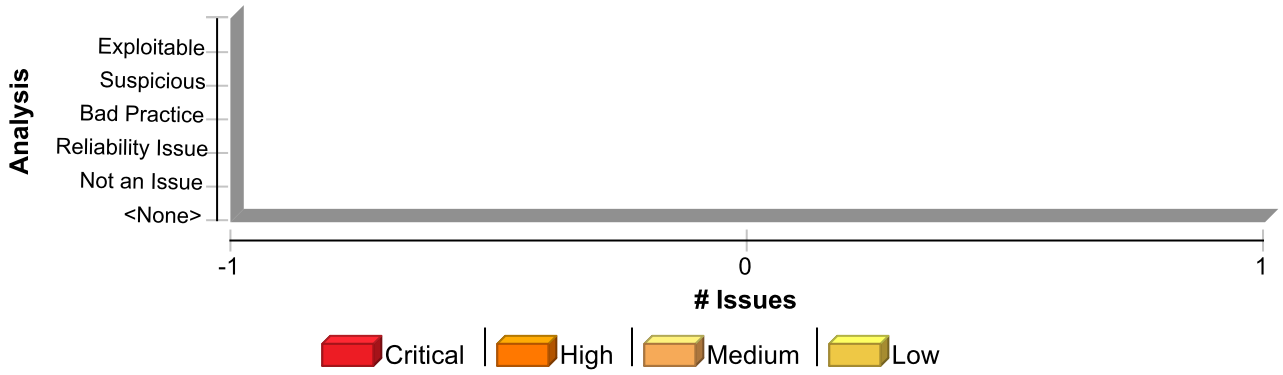
Example 2: The following method tries to check the variables `firstCall` and `secondCall`. (The variable `firstCall` is mistakenly set to `true` instead of being checked.) The result is that the first part of the expression `firstCall = true && secondCall == true` will always evaluate to `true`.

```
public void setUpCalls() {
  boolean firstCall = false;
  boolean secondCall = false;

  if (fCall > 0) {
    setUpFCall();
    firstCall = true;
  }
  if (sCall > 0) {
    setUpSCall();
    secondCall = true;
  }

  if (firstCall = true && secondCall == true) {
    setUpDualCall();
  }
}
```

Recommendation

In general, you should repair or remove unused code. It causes additional complexity and maintenance burden without contributing to the functionality of the program.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Dead Code: Expression is Always true	1	1	0	2
Total	**1**	**1**	**0**	**2**

Dead Code: Expression is Always true — Low

Package: org.owasp.webgoat.util

Exec.java, line 118 (Dead Code: Expression is Always true) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: IfStatement
Enclosing Method: execOptions()
File: Exec.java:118

```
115 }
116
117 // Write to the child process' input stream
118 if ((input != null) && !input.equals(""))
119 {
120 try
121 {
```

Dead Code: Unused Method (2 issues)

Abstract

This method is not reachable from any method outside the class.

Explanation

This method is never called or is only called from other dead code.

Example 1: In the following class, the method `doWork()` can never be called.

```
public class Dead {
  private void doWork() {
    System.out.println("doing work");
  }
  public static void main(String[] args) {
    System.out.println("running Dead");
  }
}
```

Example 2: In the following class, two private methods call each other, but since neither one is ever invoked from anywhere else, they are both dead code.

```
public class DoubleDead {
  private void doTweedledee() {
    doTweedledumb();
  }
  private void doTweedledumb() {
    doTweedledee();
  }
  public static void main(String[] args) {
    System.out.println("running DoubleDead");
  }
}
```

(In this case it is a good thing that the methods are dead: invoking either one would cause an infinite loop.)

Recommendation

A dead method may indicate a bug in dispatch code.

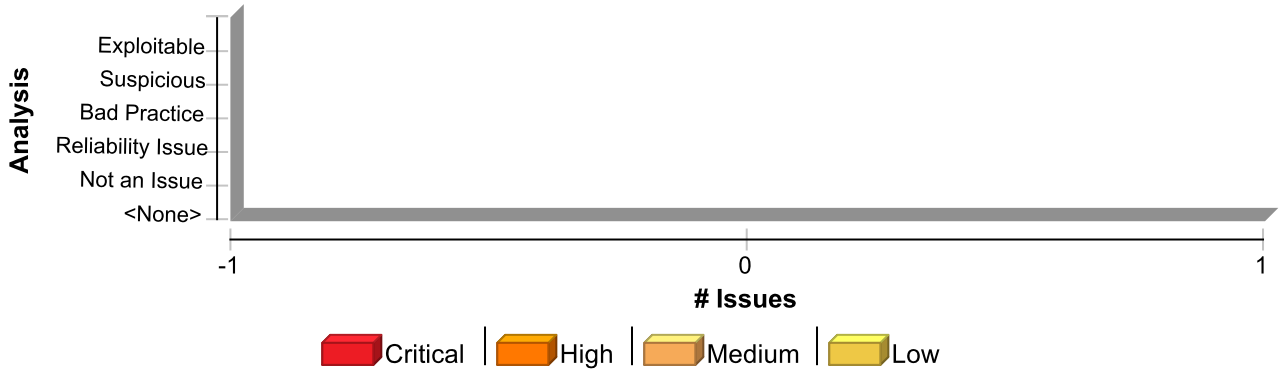
Example 3: If method is flagged as dead named `getWitch()` in a class that also contains the following dispatch method, it may be because of a copy-and-paste error. The 'w' case should return `getWitch()` not `getMummy()`.

```
public ScaryThing getScaryThing(char st) {
  switch(st) {
    case 'm':
      return getMummy();
    case 'w':
      return getMummy();
    default:
      return getBlob();
  }
```

```
}
```

In general, you should repair or remove dead code. To repair dead code, execute the dead code directly or indirectly through a public method. Dead code causes additional complexity and maintenance burden without contributing to the functionality of the program.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Dead Code: Unused Method	2	2	0	4
Total	**2**	**2**	**0**	**4**

Dead Code: Unused Method — Low

Package: org.owasp.webgoat

HammerHead.java, line 237 (Dead Code: Unused Method) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: dumpSession
Enclosing Method: dumpSession()
File: HammerHead.java:237

```
234 * @param session
235 * Description of the Parameter
236 */
237 private void dumpSession(HttpSession session)
238 {
239 Enumeration enumerator = session.getAttributeNames();
240
```

Dead Code: Unused Method — Low

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 304 (Dead Code: Unused Method) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: exec
Enclosing Method: exec()
File: CommandInjection.java:304

```
301 * @param s Description of the Parameter
302 * @return Description of the Return Value
303 */
304 private Element exec(WebSession s, String command, String args)
305 {
306 System.out.println("Executing OS command: '" + command
307 + "' with args: '" + args + "'");
```

Denial of Service (7 issues)

Abstract

An attacker could cause the program to crash or otherwise become unavailable to legitimate users.

Explanation

Attackers may be able to deny service to legitimate users by flooding the application with requests, but flooding attacks can often be defused at the network layer. More problematic are bugs that allow an attacker to overload the application using a small number of requests. Such bugs allow the attacker to specify the quantity of system resources their requests will consume or the duration for which they will use them.

Example 1: The following code allows a user to specify the amount of time for which a thread will sleep. By specifying a large number, an attacker may tie up the thread indefinitely. With a small number of requests, the attacker may deplete the application's thread pool.

```
  int usrSleepTime = Integer.parseInt(usrInput);
  Thread.sleep(usrSleepTime);
```

Example 2: The following code reads a String from a zip file. Because it uses the `readLine()` method, it will read an unbounded amount of input. An attacker may take advantage of this code to cause an `OutOfMemoryException` or to consume a large amount of memory so that the program spends more time performing garbage collection or runs out of memory during some subsequent operation.

```
  InputStream zipInput = zipFile.getInputStream(zipEntry);
  Reader zipReader = new InputStreamReader(zipInput);
  BufferedReader br = new BufferedReader(zipReader);
  String line = br.readLine();
```

Recommendation

Validate user input to ensure that it will not cause inappropriate resource utilization.

Example 3: The following code allows a user to specify the amount of time for which a thread will sleep just as in Example 1, but only if the value is within reasonable bounds.

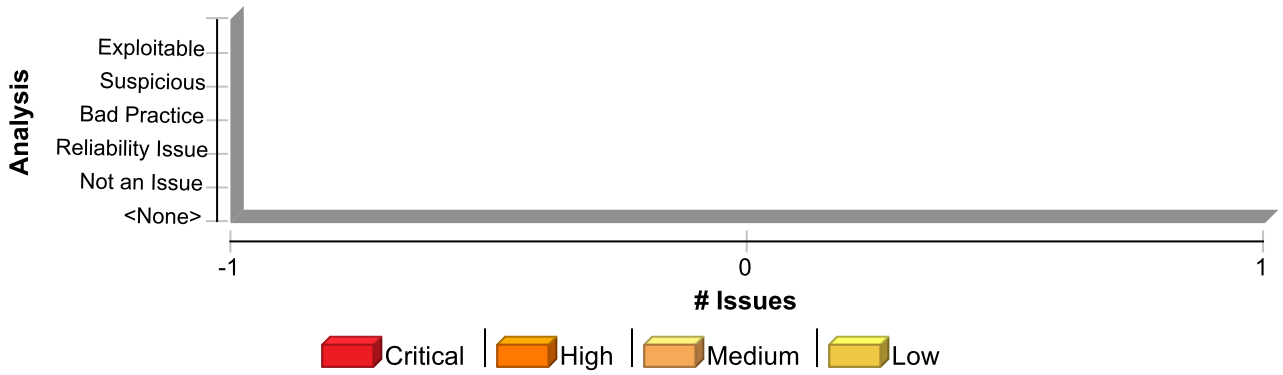
```
  int usrSleepTime = Integer.parseInt(usrInput);
  if (usrSleepTime >= SLEEP_MIN &&
      usrSleepTime <= SLEEP_MAX) {
    Thread.sleep(usrSleepTime);
  } else {
    throw new Exception("Invalid sleep duration");
  }
}
```

Example 4: The following code reads a String from a zip file just as in Example 2, but the maximum string length it will read is `MAX_STR_LEN` characters.

```
  InputStream zipInput = zipFile.getInputStream(zipEntry);
  Reader zipReader = new InputStreamReader(zipInput);
  BufferedReader br = new BufferedReader(zipReader);
  StringBuffer sb = new StringBuffer();
  int intC;
  while ((intC = br.read()) != -1) {
    char c = (char) intC;
```

```
    if (c == '\n') {
      break;
    }
    if (sb.length() >= MAX_STR_LEN) {
      throw new Exception("input too long");
    }
    sb.append(c);
  }
  String line = sb.toString();
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Denial of Service	7	7	0	14
Total	**7**	**7**	**0**	**14**

Denial of Service — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 288 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: readLine()
Enclosing Method: getInstructions()
File: LessonAdapter.java:288

```
285 BufferedReader in = new BufferedReader(new FileReader(fileName));
286 String line = null;
287 boolean startAppending = false;
288 while ((line = in.readLine()) != null)
289 {
290 if (line.indexOf("<!-- Start Instructions -->") != -1)
291 {
```

Denial of Service — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 288 (Denial of Service) — Low

AbstractLesson.java, line 1035 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: readLine()
Enclosing Method: readFromURL()
File: AbstractLesson.java:1035

```
1032 huc.getInputStream()));
1033 String line;
1034
1035 while ((line = reader.readLine()) != null)
1036 {
1037 ec.addElement(new StringElement(line));
1038 }
```

AbstractLesson.java, line 380 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: readLine()
Enclosing Method: getFileMethod()
File: AbstractLesson.java:380

```
377 {
378 String line;
379
380 while ((line = reader.readLine()) != null)
381 {
382 if ((line.indexOf(methodName) != -1)
383 && ((line.indexOf("public") != -1)
```

LessonAdapter.java, line 95 (Denial of Service) — Low

Issue Details

Denial of Service — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 95 (Denial of Service) — Low

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: readLine()
Enclosing Method: createContent()
File: LessonAdapter.java:95

```
92  PRE pre = new PRE();
93  BufferedReader in = new BufferedReader(new FileReader(fileName));
94  String line = null;
95  while ((line = in.readLine()) != null)
96  {
97  pre.addElement(line + "\n");
98  }
```

AbstractLesson.java, line 465 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: readLine()
Enclosing Method: getFileText()
File: AbstractLesson.java:465

```
462  {
463  String line;
464
465  while ((line = reader.readLine()) != null)
466  {
467  if (numbers)
468  {
```

Package: org.owasp.webgoat.util

Exec.java, line 418 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Denial of Service — Low

Package: org.owasp.webgoat.util

Exec.java, line 418 (Denial of Service) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: waitFor()
Enclosing Method: execOptions()
File: Exec.java:418

```
415 // wait for the return value of the child process.
416 if (!interrupted.get(0) && !lazyQuit)
417 {
418 int returnCode = child.waitFor();
419 results.setReturnCode(returnCode);
420
421 if (returnCode != successCode)
```

Exec.java, line 229 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: waitFor()
Enclosing Method: execOptions()
File: Exec.java:229

```
226 // wait for the return value of the child process.
227 if (!interrupted.get(0) && !lazyQuit)
228 {
229 int returnCode = child.waitFor();
230 results.setReturnCode(returnCode);
231
232 if (returnCode != successCode)
```

Denial of Service: Parse Double (1 issue)

Abstract

The program calls a method that parses doubles and can cause the thread to hang.

Explanation

There is a vulnerability in implementations of `java.lang.Double.parseDouble()` and related methods that can cause the thread to hang when parsing any number in the range `[2^(-1022) - 2^(-1075) : 2^(-1022) - 2^(-1076)]`. This defect can be used to execute a Denial of Service (DoS) attack.

Example 1: The following code uses a vulnerable method.

```
Double d = Double.parseDouble(request.getParameter("d"));
```

An attacker could send requests where the parameter `d` is a value in the range, such as `"0.0222507385850720119e-00306"`, to cause the program to hang while processing the request.

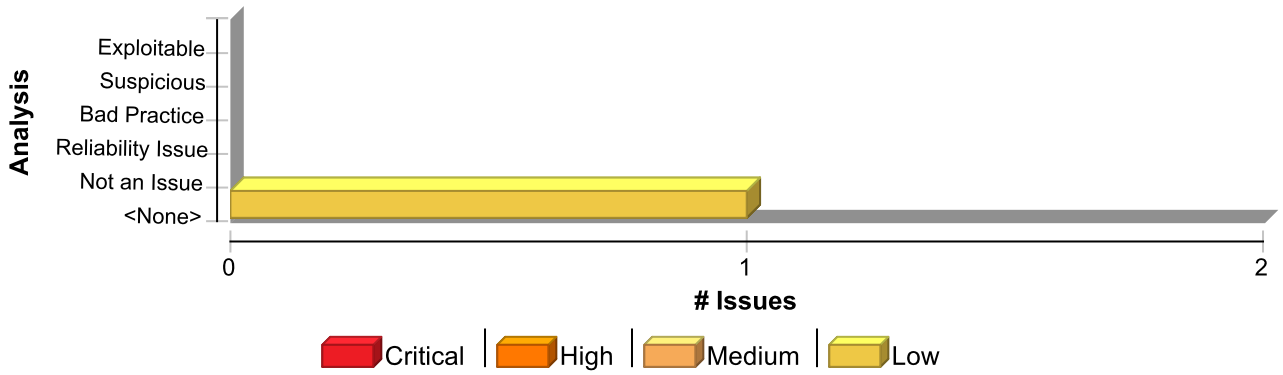
This vulnerability exists for Java version 6 Update 23 and earlier versions. It is not present for Java version 6 Update 24 and later.

Recommendation

If possible, apply the patch released by Oracle. Install patches for other vulnerable products, such as Apache Tomcat, as available. If this is not possible, be sure your installation of the Fortify Real-Time Analyzer is configured to protect against this attack.

If the vulnerable method call is within your program, you may be able to avoid using `Double`. However, this is not guaranteed to eliminate the issue. Other numeric classes, such as `BigInteger`, use the `parseDouble()` method in their implementation.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Denial of Service: Parse Double	1	1	0	2
Total	**1**	**1**	**0**	**2**

Denial of Service: Parse Double — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 280 (Denial of Service: Parse Double) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: java.lang.Double.doubleValue()
Enclosing Method: getDoubleParameter()
File: ParameterParser.java:280
Taint Flags: NUMBER, WEB

```
277 public double getDoubleParameter(String name)
278 throws ParameterNotFoundException, NumberFormatException
279 {
280 return new Double(getStringParameter(name)).doubleValue();
281 }
282
283
```

Denial of Service: Regular Expression (4 issues)

Abstract

Untrusted data is passed to the application and used as a regular expression. This can cause the thread to overconsume CPU resources.

Explanation

There is a vulnerability in implementations of regular expression evaluators and related methods that can cause the thread to hang when evaluating regular expressions that contain a grouping expression that is itself repeated. Additionally, any regular expression that contains alternate subexpressions that overlap one another can also be exploited. This defect can be used to execute a Denial of Service (DoS) attack. **Example:**

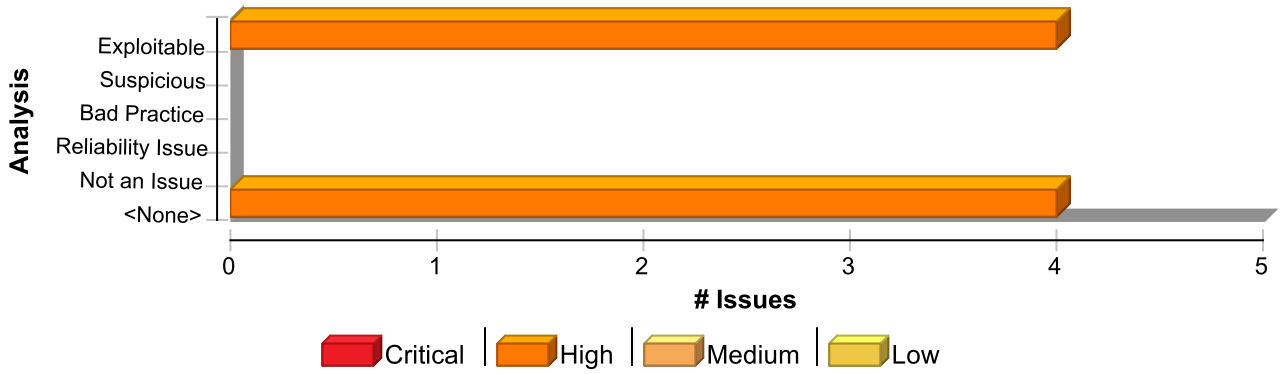
```
(e+)+
([a-zA-Z]+)*
(e|ee)+
```

There are no known regular expression implementations which are immune to this vulnerability. All platforms and languages are vulnerable to this attack.

Recommendation

Do not allow untrusted data to be used as regular expression patterns.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Denial of Service: Regular Expression	4	4	0	8
Total	**4**	**4**	**0**	**8**

Denial of Service: Regular Expression	**High**
Package: org.owasp.webgoat.lessons	
PathBasedAccessControl.java, line 199 (Denial of Service: Regular Expression)	**High**

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Denial of Service: Regular Expression — High

Package: org.owasp.webgoat.lessons

PathBasedAccessControl.java, line 199 (Denial of Service: Regular Expression) — High

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.PathBasedAccessControl.createContent
File: JavaSource/org/owasp/webgoat/lessons/PathBasedAccessControl.java:199

```
196  throw new Exception("File is binary");
197  }
198  ec.addElement(new StringElement(fileData.replaceAll(
199  System.getProperty("line.separator"), "<br>")
200  .replaceAll("(?s)<!DOCTYPE.*/head>", "")
201  .replaceAll("<br><br>", "<br>").replaceAll(
202  "<br>\\s<br>", "<br>").replaceAll("<\\?",
```

Sink Details

Sink: java.lang.String.replaceAll()
Enclosing Method: createContent()
File: PathBasedAccessControl.java:199
Taint Flags: PROPERTY, SYSTEMINFO

```
196 throw new Exception("File is binary");
197 }
198 ec.addElement(new StringElement(fileData.replaceAll(
199 System.getProperty("line.separator"), "<br>")
200 .replaceAll("(?s)<!DOCTYPE.*/head>", "")
201 .replaceAll("<br><br>", "<br>").replaceAll(
202 "<br>\\s<br>", "<br>").replaceAll("<\\?",
```

HttpSplitting.java, line 100 (Denial of Service: Regular Expression) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Denial of Service: Regular Expression — High

Package: org.owasp.webgoat.lessons

HttpSplitting.java, line 100 (Denial of Service: Regular Expression) — High

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.HttpSplitting.doHTTPSplitting
File: JavaSource/org/owasp/webgoat/lessons/HttpSplitting.java:98

```
95  if (lang.length() != 0 && fromRedirect.length() != 0)
96  {
97  //Split by the line separator line.separator is platform independant
98  String lineSep = System.getProperty("line.separator");
99  String[] arrTokens = lang.toString().toUpperCase().split(
100  lineSep);
101
```

Sink Details

Sink: java.lang.String.split()
Enclosing Method: doHTTPSplitting()
File: HttpSplitting.java:100
Taint Flags: PROPERTY, SYSTEMINFO

```
97  //Split by the line separator line.separator is platform independant
98  String lineSep = System.getProperty("line.separator");
99  String[] arrTokens = lang.toString().toUpperCase().split(
100  lineSep);
101
102  //Check if the user ended the first request and wrote the second malacious reply
103
```

CommandInjection.java, line 201 (Denial of Service: Regular Expression) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Denial of Service: Regular Expression — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 201 (Denial of Service: Regular Expression) — High

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.CommandInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/CommandInjection.java:201

```
198  ec.addElement(new BR());
199  ec.addElement(new HR().setWidth("90%"));
200  ec.addElement(new StringElement(fileData.replaceAll(
201  System.getProperty("line.separator"), "<br>")
202  .replaceAll("(?s)<!DOCTYPE.*/head>", "").replaceAll(
203  "<br><br>", "<br>").replaceAll("<br>\\s<br>",
204  "<br>")));
```

Sink Details

Sink: java.lang.String.replaceAll()
Enclosing Method: createContent()
File: CommandInjection.java:201
Taint Flags: PROPERTY, SYSTEMINFO

```
198  ec.addElement(new BR());
199  ec.addElement(new HR().setWidth("90%"));
200  ec.addElement(new StringElement(fileData.replaceAll(
201  System.getProperty("line.separator"), "<br>")
202  .replaceAll("(?s)<!DOCTYPE.*/head>", "").replaceAll(
203  "<br><br>", "<br>").replaceAll("<br>\\s<br>",
204  "<br>")));
```

Package: org.owasp.webgoat.session

Screen.java, line 331 (Denial of Service: Regular Expression) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: java.lang.System.getProperty()

Denial of Service: Regular Expression — High

Package: org.owasp.webgoat.session

Screen.java, line 331 (Denial of Service: Regular Expression) — High

From: org.owasp.webgoat.session.Screen.convertMetachars
File: JavaSource/org/owasp/webgoat/session/Screen.java:324

```
321  * screen size, say less than 80 characters.
322  */
323  String[] metaChar = { "&", "<", ">", "\"", "\t",
324  System.getProperty("line.separator") };
325
326  String[] htmlCode = { "&amp;", "&lt;", "&gt;", "&quot;", " ", "<br>" };
327
```

Sink Details

Sink: java.lang.String.replaceAll()
Enclosing Method: convertMetachars()
File: Screen.java:331
Taint Flags: PROPERTY, SYSTEMINFO

```
328  String replacedString = token;
329  for (; mci < metaChar.length; mci += 1)
330  {
331  replacedString = replacedString.replaceAll(metaChar[mci],
332  htmlCode[mci]);
333  }
334  return (replacedString);
```

Denial of Service: StringBuilder (32 issues)

Abstract

Appending untrusted data to `StringBuilder` instance initialized with the default backing-array size can cause the JVM to over-consume heap memory space.

Explanation

Appending user-controlled data to a `StringBuilder` instance initialized with the default backing character array size (16) can cause the application to consume large amounts of heap memory while resizing the underlying array to fit user's data. Everytime new data is appended to a `StringBuilder` instance, it will try to fit it on its backing character array. If data does not fit, a new array will be created doubling the previous size while the old array will remain in the heap until it is garbage collected. This defect can be used to execute a Denial of Service (DoS) attack.

Example 1: User-controlled data is appended to a `StringBuilder` instance initialized with the default constructor.

```
...
StringBuilder sb = new StringBuilder();
sb.append(request.getParameter("foo"));
...
```

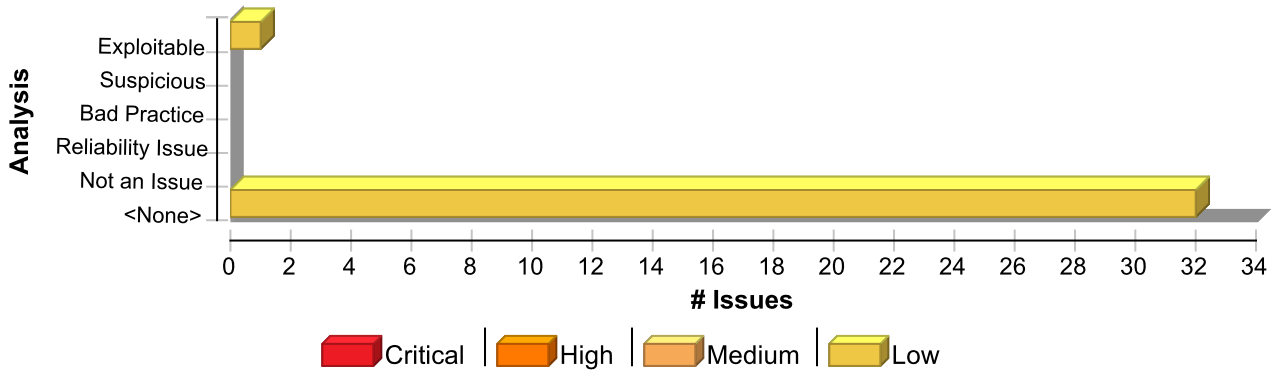
Recommendation

Initialize the `StringBuilder` with a size similar to the length of the expected data in order to reduce the number of times the backing array needs to be resized. Check the size of the data before appending it to a `StringBuilder` instance.

Example 2: User-controlled data is appended to a `StringBuilder` instance initialized with the default constructor.

```
...
private final int MAX_DATA = 128;
private final int EXPECTED_BUFFER_DATA = 1024;
StringBuilder sb = new StringBuilder(EXPECTED_BUFFER_DATA);
...
String data = request.getParameter("foo");
if (data.length() < MAX_DATA) sb.append(data);
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Denial of Service: StringBuilder	32	32	0	64
Total	**32**	**32**	**0**	**64**

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.JavaScriptValidation.createContent
File: JavaSource/org/owasp/webgoat/lessons/JavaScriptValidation.java:87

```
84 Pattern pattern5 = Pattern.compile(regex5);
85 Pattern pattern6 = Pattern.compile(regex6);
86 Pattern pattern7 = Pattern.compile(regex7);
87 String lineSep = System.getProperty("line.separator");
88 String script = "<SCRIPT>"
89 + lineSep
90 + "regex1=/"
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

Encoding.java, line 727 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 727 (Denial of Service: StringBuilder) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: main(0)
From: org.owasp.webgoat.lessons.Encoding.main
File: JavaSource/org/owasp/webgoat/lessons/Encoding.java:740

```
737  * @param args The command line arguments
738  */
739
740  public static void main( String[] args )
741  {
742  try
743  {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: hexEncode()
File: Encoding.java:727
Taint Flags: ARGS

```
724  for ( int i = 0; i < asciiString.length(); i++ )
725  {
726  hexBuff.append( "%" );
727  hexBuff.append( Integer.toHexString( ascii[i] ) );
728  }
729  return hexBuff.toString().toUpperCase();
730  }
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: CANONICAL_PATH, NO_NEW_LINE, TAINTED_PATH, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
326 value.append("ExecResults for \"" + myCommand + "\"" + sep);
327
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

Encoding.java, line 727 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: hexEncode()
File: Encoding.java:727
Taint Flags: WEB, XSS

```
724 for ( int i = 0; i < asciiString.length(); i++ )
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 727 (Denial of Service: StringBuilder) — Low

```
725 {
726 hexBuff.append( "%" );
727 hexBuff.append( Integer.toHexString( ascii[i] ) );
728 }
729 return hexBuff.toString().toUpperCase();
730 }
```

CommandInjection.java, line 230 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
326 value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: parseResults()
File: CommandInjection.java:230
Taint Flags: PROPERTY, SYSTEMINFO, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR

```
227
228 if(s.length() > 0 && s.endsWith(".help"))
229 {
230 modified.append(s + "\n");
231 }
232 }
233
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

SilentTransactions.java, line 87 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: handleRequest()
File: SilentTransactions.java:87
Taint Flags: WEB, XSS

```
84 if (!amount.equals(""))
85 {
86 result.append("You have just silently authorized ");
87 result.append(amount);
88 result.append("$ without the user interaction.<br>");
89 }
90 result
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.SilentTransactions.createContent
File: JavaSource/org/owasp/webgoat/lessons/SilentTransactions.java:139

```
136 protected Element createContent(WebSession s)
137 {
138 ElementContainer ec = new ElementContainer();
139 String lineSep = System.getProperty("line.separator");
140 String script = "<script>"
141 + lineSep
142 + "function processData(){"
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.AbstractLesson.getFileText
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:471

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

```
473
474 reader.close();
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.XMLInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/XMLInjection.java:151

```
148 {
149 isDone = true;
150 }
151 String lineSep = System.getProperty("line.separator");
152 String script = "<script>"
153 + lineSep
154 + "function getRewards() {"
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

LessonAdapter.java, line 302 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.io.BufferedReader.readLine()
From: org.owasp.webgoat.lessons.LessonAdapter.getInstructions
File: JavaSource/org/owasp/webgoat/lessons/LessonAdapter.java:288

```
285 BufferedReader in = new BufferedReader(new FileReader(fileName));
286 String line = null;
287 boolean startAppending = false;
288 while ((line = in.readLine()) != null)
289 {
290 if (line.indexOf("<!-- Start Instructions -->") != -1)
291 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getInstructions()
File: LessonAdapter.java:302
Taint Flags: NO_NEW_LINE, STREAM

```
299 }
300 if (startAppending)
301 {
302 buff.append(line + "\n");
303 }
304 }
305 }
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 302 (Denial of Service: StringBuilder) — Low

AbstractLesson.java, line 398 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.io.BufferedReader.readLine()
From: org.owasp.webgoat.lessons.AbstractLesson.getFileMethod
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:380

```
377  {
378  String line;
379
380  while ((line = reader.readLine()) != null)
381  {
382  if ((line.indexOf(methodName) != -1)
383  && ((line.indexOf("public") != -1)
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileMethod()
File: AbstractLesson.java:398
Taint Flags: NO_NEW_LINE, STREAM

```
395  sb.append(pad(++count) + " ");
396  }
397
398  sb.append(line + "\n");
399  }
400
401  if (echo && (line.indexOf("{") != -1))
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getReader()
From: org.owasp.webgoat.lessons.AbstractLesson.makeRequestDump_DELETEME
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:900

```
897
898  try
899  {
900  el = new StringElement(readFromFile(s.getRequest().getReader(),
901  false));
902  }
903  catch (Exception e)
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, WEB, XSS

```
468  {
469  sb.append(pad(++count) + " ");
470  }
471  sb.append(line + System.getProperty("line.separator"));
472  }
473
474  reader.close();
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.DOMInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/DOMInjection.java:95

```
92  e.printStackTrace();
93  }
94
95  String lineSep = System.getProperty("line.separator");
96  String script = "<script>" + lineSep + "function validate() {"
97  + lineSep + "var keyField = document.getElementById('key');"
98  + lineSep + "var url = '/WebGoat/attack?Screen="
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO

```
468  {
469  sb.append(pad(++count) + " ");
470  }
471  sb.append(line + System.getProperty("line.separator"));
472  }
473
474  reader.close();
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.io.BufferedReader.readLine()
From: org.owasp.webgoat.lessons.AbstractLesson.getFileText
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:465

```
462  {
463  String line;
464
465  while ((line = reader.readLine()) != null)
466  {
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

```
467  if (numbers)
468  {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, STREAM, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR

```
468  {
469  sb.append(pad(++count) + " ");
470  }
471  sb.append(line + System.getProperty("line.separator"));
472  }
473
474  reader.close();
```

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.JSONInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/JSONInjection.java:121

```
118  protected Element createContent(WebSession s)
119  {
120  ElementContainer ec = new ElementContainer();
121  String lineSep = System.getProperty("line.separator");
122  String script = "<script>"
123  + lineSep
124  + "function getFlights() {"
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: getFileText()
File: AbstractLesson.java:471
Taint Flags: NO_NEW_LINE, PROPERTY, SYSTEMINFO

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 471 (Denial of Service: StringBuilder) — Low

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

Package: org.owasp.webgoat.session

WebSession.java, line 868 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: setMessage()
File: WebSession.java:868
Taint Flags: WEB, XSS

```
865 */
866 public void setMessage( String text )
867 {
868 message.append( "<BR>" + " * " + text);
869 }
870
871 /**
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.session

WebSession.java, line 868 (Denial of Service: StringBuilder) — Low

WebSession.java, line 868 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: org.owasp.webgoat.session.WebSession.getCookie
File: JavaSource/org/owasp/webgoat/session/WebSession.java:621

```
618 */
619 public String getCookie( String cookieName )
620 {
621 Cookie[] cookies = getRequest().getCookies();
622
623 for ( int i = 0; i < cookies.length; i++ )
624 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: setMessage()
File: WebSession.java:868
Taint Flags: WEB, XSS

```
865 */
866 public void setMessage( String text )
867 {
868 message.append( "<BR>" + " * " + text);
869 }
870
871 /**
```

WebSession.java, line 868 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.session

WebSession.java, line 868 (Denial of Service: StringBuilder) — Low

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: setMessage()
File: WebSession.java:868
Taint Flags: WEB, XSS

```
865 */
866 public void setMessage( String text )
867 {
868 message.append( "<BR>" + " * " + text);
869 }
870
871 /**
```

Package: org.owasp.webgoat.util

HtmlEncoder.java, line 155 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: main(0)
From: org.owasp.webgoat.lessons.Encoding.main
File: JavaSource/org/owasp/webgoat/lessons/Encoding.java:740

```
737 * @param args The command line arguments
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

HtmlEncoder.java, line 155 (Denial of Service: StringBuilder) — Low

```
738  */
739
740  public static void main( String[] args )
741  {
742  try
743  {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: encode()
File: HtmlEncoder.java:155
Taint Flags: ARGS

```
152  {
153  if (((int) ch) > 128)
154  {
155  buf.append("&#" + ((int) ch) + ";");
156  }
157  else
158  {
```

HtmlEncoder.java, line 155 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

HtmlEncoder.java, line 155 (Denial of Service: StringBuilder) — Low

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: encode()
File: HtmlEncoder.java:155
Taint Flags: WEB, XSS

```
152 {
153 if (((int) ch) > 128)
154 {
155 buf.append("&#" + ((int) ch) + ";");
156 }
157 else
158 {
```

ExecResults.java, line 335 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
326 value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:335
Taint Flags: PROPERTY, SYSTEMINFO

```
332
333 if ((myOutput != null) && !myOutput.equals(""))
334 {
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 335 (Denial of Service: StringBuilder) — Low

```
335  value.append(sep + "Output..." + sep + myOutput + sep);
336  }
337
338  if ((myErrors != null) && !myErrors.equals(""))
```

ExecResults.java, line 347 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321  */
322  public String toString()
323  {
324  String sep = System.getProperty("line.separator");
325  StringBuffer value = new StringBuffer();
326  value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:347
Taint Flags: PROPERTY, SYSTEMINFO

```
344
345  if (myInterrupted)
346  {
347  value.append("Command timed out after " + (myTimeout / 1000)
348  + " seconds " + sep);
349  }
350
```

HtmlEncoder.java, line 209 (Denial of Service: StringBuilder) — Low

Issue Details

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

HtmlEncoder.java, line 209 (Denial of Service: StringBuilder) — Low

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: main(0)
From: org.owasp.webgoat.lessons.Encoding.main
File: JavaSource/org/owasp/webgoat/lessons/Encoding.java:740

```
737  * @param args The command line arguments
738  */
739
740  public static void main( String[] args )
741  {
742  try
743  {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: decode()
File: HtmlEncoder.java:209
Taint Flags: ARGS

```
206  }
207  if (iso == null)
208  {
209  buf.append("&" + entity + ";");
210  }
211  else
212  {
```

ExecResults.java, line 340 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 340 (Denial of Service: StringBuilder) — Low

From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
326 value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:340
Taint Flags: PROPERTY, SYSTEMINFO

```
337
338 if ((myErrors != null) && !myErrors.equals(""))
339 {
340 value.append(sep + "Errors..." + sep + myErrors + sep);
341 }
342
343 value.append(sep);
```

ExecResults.java, line 343 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 343 (Denial of Service: StringBuilder) — Low

```
326  value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:343
Taint Flags: PROPERTY, SYSTEMINFO

```
340  value.append(sep + "Errors..." + sep + myErrors + sep);
341  }
342
343  value.append(sep);
344
345  if (myInterrupted)
346  {
```

ExecResults.java, line 355 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321  */
322  public String toString()
323  {
324  String sep = System.getProperty("line.separator");
325  StringBuffer value = new StringBuffer();
326  value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:355
Taint Flags: PROPERTY, SYSTEMINFO

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 355 (Denial of Service: StringBuilder) — Low

```
352
353 if (myError)
354 {
355 value.append(getErrorMessage() + sep);
356 }
357
358 return (value.toString());
```

HtmlEncoder.java, line 209 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: decode()
File: HtmlEncoder.java:209
Taint Flags: WEB, XSS

```
206 }
207 if (iso == null)
208 {
209 buf.append("&" + entity + ";");
210 }
211 else
212 {
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 326 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321  */
322  public String toString()
323  {
324  String sep = System.getProperty("line.separator");
325  StringBuffer value = new StringBuffer();
326  value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:326
Taint Flags: PROPERTY, SYSTEMINFO

```
323  {
324  String sep = System.getProperty("line.separator");
325  StringBuffer value = new StringBuffer();
326  value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
328  if ((myInput != null) && !myInput.equals(""))
329  {
```

ExecResults.java, line 330 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 330 (Denial of Service: StringBuilder) — Low

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
326 value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:330
Taint Flags: PROPERTY, SYSTEMINFO

```
327
328 if ((myInput != null) && !myInput.equals(""))
329 {
330 value.append(sep + "Input..." + sep + myInput + sep);
331 }
332
333 if ((myOutput != null) && !myOutput.equals(""))
```

ExecResults.java, line 351 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.ExecResults.toString
File: JavaSource/org/owasp/webgoat/util/ExecResults.java:324

```
321 */
322 public String toString()
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
```

Denial of Service: StringBuilder — Low

Package: org.owasp.webgoat.util

ExecResults.java, line 351 (Denial of Service: StringBuilder) — Low

```
326  value.append("ExecResults for \"" + myCommand + "\"" + sep);
327
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:351
Taint Flags: PROPERTY, SYSTEMINFO

```
348  + " seconds " + sep);
349  }
350
351  value.append("Returncode: " + myReturnCode + sep);
352
353  if (myError)
354  {
```

ExecResults.java, line 326 (Denial of Service: StringBuilder) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.lang.StringBuffer.append()
Enclosing Method: toString()
File: ExecResults.java:326
Taint Flags: TAINTED_PATH, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

Denial of Service: StringBuilder	Low
Package: org.owasp.webgoat.util	
ExecResults.java, line 326 (Denial of Service: StringBuilder)	**Low**

```
323 {
324 String sep = System.getProperty("line.separator");
325 StringBuffer value = new StringBuffer();
326 value.append("ExecResults for \'" + myCommand + "\'" + sep);
327
328 if ((myInput != null) && !myInput.equals(""))
329 {
```

Dynamic Code Evaluation: Code Injection (1 issue)

Abstract

Interpreting user-controlled instructions at run-time can allow attackers to execute malicious code.

Explanation

Many modern programming languages allow dynamic interpretation of source instructions. This capability allows programmers to perform dynamic instructions based on input received from the user. Code injection vulnerabilities occur when the programmer incorrectly assumes that instructions supplied directly from the user will perform only innocent operations, such as performing simple calculations on active user objects or otherwise modifying the user's state. However, without proper validation, a user might specify operations the programmer does not intend.

Example: In this classic code injection example, the application implements a basic calculator that allows the user to specify commands for execution.

```
...
    userOp = form.operation.value;
    calcResult = eval(userOp);
...
```

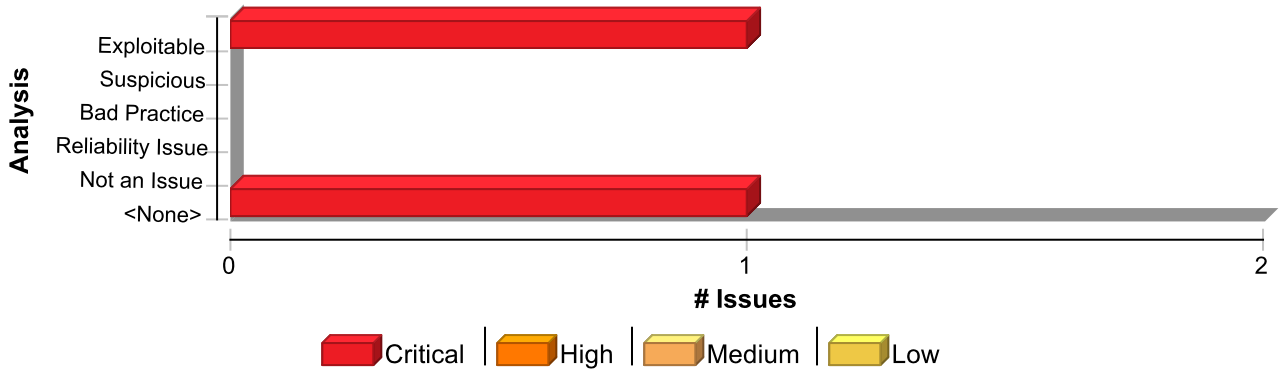
The program behaves correctly when the `operation` parameter is a benign value, such as "8 + 7 * 2", in which case the `calcResult` variable is assigned a value of 22. However, if an attacker specifies languages operations that are both valid and malicious, those operations would be executed with the full privilege of the parent process. Such attacks are even more dangerous when the underlying language provides access to system resources or allows execution of system commands. In the case of JavaScript, the attacker may utilize this vulnerability to perform a cross-site scripting attack.

Recommendation

Avoid dynamic code interpretation whenever possible. If your program's functionality requires code to be interpreted dynamically, the likelihood of attack can be minimized by constraining the code your program will execute dynamically as much as possible, limiting it to an application- and context-specific subset of the base programming language.

If dynamic code execution is required, unvalidated user input should never be directly executed and interpreted by the application. Instead, use a level of indirection: create a list of legitimate operations and data objects that users are allowed to specify, and only allow users to select from the list. With this approach, input provided by users is never executed directly.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Dynamic Code Evaluation: Code Injection	1	1	0	2
Total	**1**	**1**	**0**	**2**

Dynamic Code Evaluation: Code Injection — Critical

Package: WebContent.javascript

menu_system.js, line 137 (Dynamic Code Evaluation: Code Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read document.URL
From: trigMM1url
File: WebContent/javascript/menu_system.js:132

```
129
130 function trigMM1url(param,opt){
131  var ur,x,i,nv,mn,pr=new Array();
132  ur=document.URL;x=ur.indexOf("?");
133  if(x>1){pr=ur.substring(x+1,ur.length).split("&");
134  for(i=0;i<pr.length;i++){nv=pr[i].split("=");
135  if(nv.length>0){if(unescape(nv[0])==param){
```

Sink Details

Dynamic Code Evaluation: Code Injection — Critical

Package: WebContent.javascript

menu_system.js, line 137 (Dynamic Code Evaluation: Code Injection) — Critical

Sink: eval()
Enclosing Method: trigMM1url()
File: menu_system.js:137
Taint Flags: VALIDATED_OPEN_REDIRECT, WEB, XSS

```
134 for(i=0;i<pr.length;i++){nv=pr[i].split("=");
135 if(nv.length>0){if(unescape(nv[0])==param){
136 mn="menu"+unescape(nv[1]);
137 eval("trigMenuMagic1('"+mn+"',"+opt+")");}}}}
138 }
139
140 document.mm1Q=true;
```

File Disclosure: J2EE (1 issue)

Abstract

Constructing a server-side redirect path with user input could allow an attacker to download application binaries (including application classes or jar files) or view arbitrary files within protected directories.

Explanation

A file disclosure occur when: 1. Data enters a program from an untrusted source.

2. The data is used to dynamically construct a path.

Example 1: The following code takes untrusted data and uses it to build a path which is used in a server side forward.

```
...
String returnURL = request.getParameter("returnURL");
    RequestDispatcher rd = request.getRequestDispatcher(returnURL);
    rd.forward();
...
```

Example 2: The following code takes untrusted data and uses it to build a path which is used in a server side forward.

```
...
    <% String returnURL = request.getParameter("returnURL"); %>
    <jsp:include page="<%=returnURL%>" />
    ...
```

If an attacker provided a URL with the request parameter matching a sensitive file location, they would be able to view that file. For example, "http://www.yourcorp.com/webApp/logic?returnURL=WEB-INF/applicationContext.xml" would allow them to view the applicationContext.xml of the application. Once the attacker had the applicationContext.xml, they could locate and download other configuration files referenced in the applicationContext.xml or even class or jar files. This would allow attackers to gain sensitive infomation about an application and target it for other types of attack.

Recommendation

Do not use untrusted data to direct requests to server-side resources. Instead, use a level of indirection between locations and paths. Instead of the following:

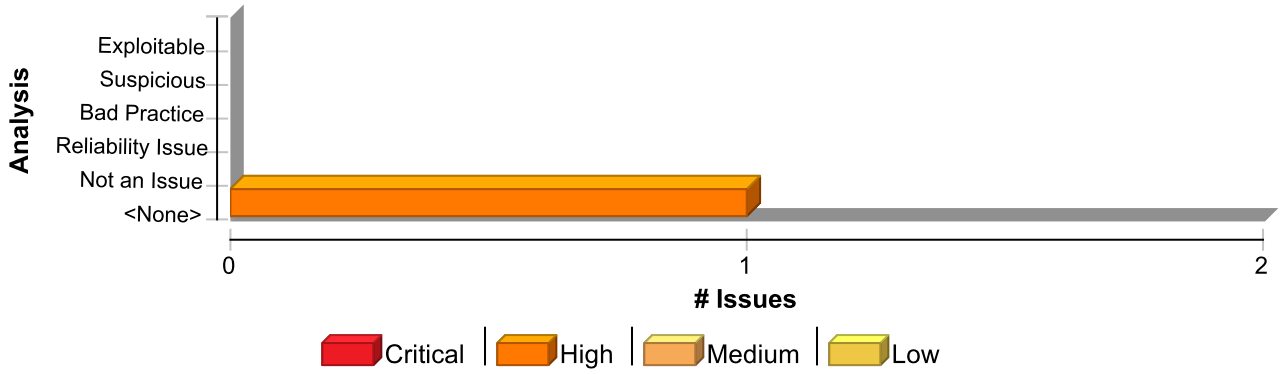
```
< a href="http://www.yourcorp.com/webApp/logic?nextPage=WEB-INF/
signup.jsp">New Customer</a>
```

Use the following:

```
< a href="http://www.yourcorp.com/webApp/logic?nextPage=newCustomer">New
Customer</a>
```

The server-side logic would have a map keyed by logical names to server-side paths such that the path stored under the key "newCustomer" would be "/WEB-INF/signup.jsp".

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
File Disclosure: J2EE	1	1	0	2
Total	**1**	**1**	**0**	**2**

File Disclosure: J2EE — High

Package: org.owasp.webgoat.util

Interceptor.java, line 135 (File Disclosure: J2EE) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getRequestURL()
From: org.owasp.webgoat.util.Interceptor.doFilter
File: JavaSource/org/owasp/webgoat/util/Interceptor.java:133

```
130  }
131  }
132
133  String url = req.getRequestURL().toString();
134
135  RequestDispatcher disp = req.getRequestDispatcher(url.substring(url
136  .lastIndexOf("WebGoat/")
```

Sink Details

Sink: javax.servlet.ServletRequest.getRequestDispatcher()
Enclosing Method: doFilter()
File: Interceptor.java:135
Taint Flags: POORVALIDATION, URL_ENCODE, VALIDATED_CROSS_SITE_SCRIPTING_DOM, VALIDATED_CROSS_SITE_SCRIPTING_INTER_COMPONENT_COMMUNICATION, VALIDATED_CROSS_SITE_SCRIPTING_PERSISTENT, VALIDATED_CROSS_SITE_SCRIPTING_REFLECTED,

File Disclosure: J2EE — High

Package: org.owasp.webgoat.util

Interceptor.java, line 135 (File Disclosure: J2EE) — High

WEB, XSS

```
132
133 String url = req.getRequestURL().toString();
134
135 RequestDispatcher disp = req.getRequestDispatcher(url.substring(url
136 .lastIndexOf("WebGoat/")
137 + "WebGoat".length()));
138
```

Header Manipulation (2 issues)

Abstract

Including unvalidated data in an HTTP response header can enable cache-poisoning, cross-site scripting, cross-user defacement, page hijacking, cookie manipulation or open redirect.

Explanation

Header Manipulation vulnerabilities occur when:

1. Data enters a web application through an untrusted source, most frequently an HTTP request.

2. The data is included in an HTTP response header sent to a web user without being validated.

As with many software security vulnerabilities, Header Manipulation is a means to an end, not an end in itself. At its root, the vulnerability is straightforward: an attacker passes malicious data to a vulnerable application, and the application includes the data in an HTTP response header.

One of the most common Header Manipulation attacks is HTTP Response Splitting. To mount a successful HTTP Response Splitting exploit, the application must allow input that contains CR (carriage return, also given by %0d or \r) and LF (line feed, also given by %0a or \n)characters into the header. These characters not only give attackers control of the remaining headers and body of the response the application intends to send, but also allows them to create additional responses entirely under their control.

Many of today's modern application servers will prevent the injection of malicious characters into HTTP headers. For example, recent versions of Apache Tomcat will throw an `IllegalArgumentException` if you attempt to set a header with prohibited characters. If your application server prevents setting headers with new line characters, then your application is not vulnerable to HTTP Response Splitting. However, solely filtering for new line characters can leave an application vulnerable to Cookie Manipulation or Open Redirects, so care must still be taken when setting HTTP headers with user input.

Example: The following code segment reads the name of the author of a weblog entry, `author`, from an HTTP request and sets it in a cookie header of an HTTP response.

```
String author = request.getParameter(AUTHOR_PARAM);
...
Cookie cookie = new Cookie("author", author);
     cookie.setMaxAge(cookieExpiration);
     response.addCookie(cookie);
```

Assuming a string consisting of standard alpha-numeric characters, such as "Jane Smith", is submitted in the request the HTTP response including this cookie might take the following form:

```
HTTP/1.1 200 OK
...
Set-Cookie: author=Jane Smith
...
```

However, because the value of the cookie is formed of unvalidated user input the response will only maintain this form if the value submitted for `AUTHOR_PARAM` does not contain any CR and LF characters. If an attacker submits a malicious string, such as "Wiley Hacker\r\nHTTP/1.1 200 OK\r\n...", then the HTTP response would be split into two responses of the following form:

```
HTTP/1.1 200 OK
...
Set-Cookie: author=Wiley Hacker

HTTP/1.1 200 OK
...
```

Clearly, the second response is completely controlled by the attacker and can be constructed with any header and body content desired. The ability of attacker to construct arbitrary HTTP responses permits a variety of resulting attacks, including: cross-user defacement, web and browser cache poisoning, cross-site scripting and page hijacking.

Cross-User Defacement: An attacker will be able to make a single request to a vulnerable server that will cause the server to create two responses, the second of which may be misinterpreted as a response to a different request, possibly one made by another user sharing the same TCP connection with the server. This can be accomplished by convincing the user to submit the malicious request themselves, or remotely in situations where the attacker and the user share a common TCP connection to the server, such as a shared proxy server. In the best case, an attacker may leverage this ability to convince users that the application has been hacked, causing users to lose confidence in the security of the application. In the worst case, an attacker may provide specially crafted content designed to mimic the behavior of the application but redirect private information, such as account numbers and passwords, back to the attacker.

Cache Poisoning: The impact of a maliciously constructed response can be magnified if it is cached either by a web cache used by multiple users or even the browser cache of a single user. If a response is cached in a shared web cache, such as those commonly found in proxy servers, then all users of that cache will continue receive the malicious content until the cache entry is purged. Similarly, if the response is cached in the browser of an individual user, then that user will continue to receive the malicious content until the cache entry is purged, although only the user of the local browser instance will be affected.

Cross-Site Scripting: Once attackers have control of the responses sent by an application, they have a choice of a variety of malicious content to provide users. Cross-site scripting is common form of attack where malicious JavaScript or other code included in a response is executed in the user's browser. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site. The most common and dangerous attack vector against users of a vulnerable application uses JavaScript to transmit session and authentication information back to the attacker who can then take complete control of the victim's account.

Page Hijacking: In addition to using a vulnerable application to send malicious content to a user, the same root vulnerability can also be leveraged to redirect sensitive content generated by the server and intended for the user to the attacker instead. By submitting a request that results in two responses, the intended response from the server and the response generated by the attacker, an attacker may cause an intermediate node, such as a shared proxy server, to misdirect a response generated by the server for the user to the attacker. Because the request made by the attacker generates two responses, the first is interpreted as a response to the attacker's request, while the second remains in limbo. When the user makes a legitimate request through the same TCP connection, the attacker's request is already waiting and is interpreted as a response to the victim's request. The attacker then sends a second request to the server, to which the proxy server responds with the server generated request intended for the victim, thereby compromising any sensitive information in the headers or body of the response intended for the victim.

Cookie Manipulation: When combined with attacks like Cross-Site Request Forgery, attackers may change, add to, or even overwrite a legitimate user's cookies.

Open Redirect: Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Recommendation

The solution to Header Manipulation is to ensure that input validation occurs in the correct places and checks for the correct properties.

Since Header Manipulation vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating responses dynamically, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for Header Manipulation.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for Header Manipulation is generally relatively easy. Despite its value, input validation for Header Manipulation does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent Header Manipulation vulnerabilities is to validate everything that enters the application or leaves the application destined for the user.

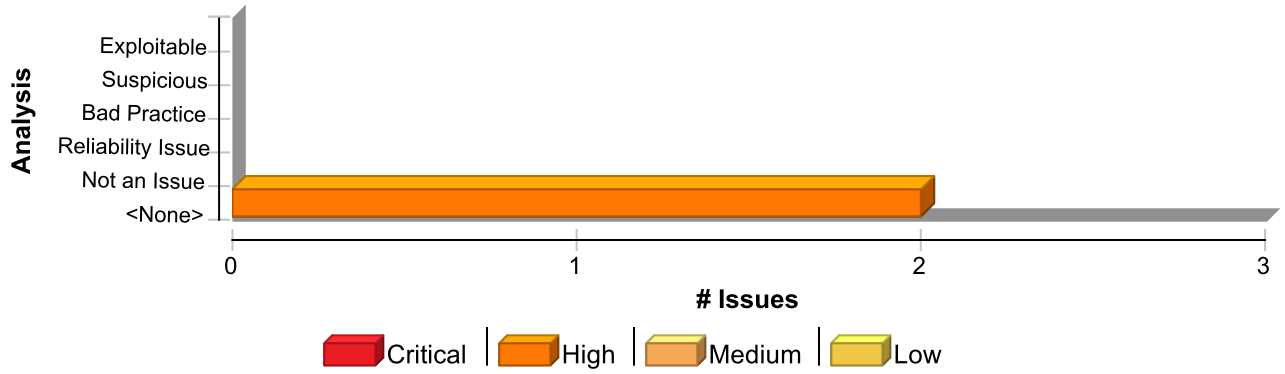
The most secure approach to validation for Header Manipulation is to create a whitelist of safe characters that are allowed to appear in HTTP response headers and accept input composed exclusively of characters in the approved set. For example, a valid name might only include alpha-numeric characters or an account number might only include digits 0-9.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning in HTTP response headers. Although the CR and LF characters are at the heart of an HTTP response splitting attack, other characters, such as ':' (colon) and '=' (equal), have special meaning in response headers as well.

Once you identify the correct points in an application to perform validation for Header Manipulation attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. The application should reject any input destined to be included in HTTP response headers that contains special characters, particularly CR and LF, as invalid.

Many application servers attempt to limit an application's exposure to HTTP response splitting vulnerabilities by providing implementations for the functions responsible for setting HTTP headers and cookies that perform validation for the characters essential to an HTTP response splitting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Header Manipulation	2	2	0	4
Total	**2**	**2**	**0**	**4**

Header Manipulation — High

Package: org.owasp.webgoat.lessons

HttpOnly.java, line 212 (Header Manipulation) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: org.owasp.webgoat.session.WebSession.getCookie
File: JavaSource/org/owasp/webgoat/session/WebSession.java:621

```
618 */
619 public String getCookie( String cookieName )
620 {
621 Cookie[] cookies = getRequest().getCookies();
622
623 for ( int i = 0; i < cookies.length; i++ )
624 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.setHeader()
Enclosing Method: removeHttpOnly()
File: HttpOnly.java:212
Taint Flags: WEB, XSS

```
209 response.setHeader("Set-Cookie", UNIQUE2U + "=" + value + ";");
210 original = value;
211 } else {
212 response.setHeader("Set-Cookie", UNIQUE2U + "=" + cookie + ";");
213 original = cookie;
214 }
215 }
```

HttpOnly.java, line 198 (Header Manipulation) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Header Manipulation — High

Package: org.owasp.webgoat.lessons

HttpOnly.java, line 198 (Header Manipulation) — High

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: org.owasp.webgoat.session.WebSession.getCookie
File: JavaSource/org/owasp/webgoat/session/WebSession.java:621

```
618 */
619 public String getCookie( String cookieName )
620 {
621 Cookie[] cookies = getRequest().getCookies();
622
623 for ( int i = 0; i < cookies.length; i++ )
624 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.setHeader()
Enclosing Method: setHttpOnly()
File: HttpOnly.java:198
Taint Flags: WEB, XSS

```
195 response.setHeader("Set-Cookie", UNIQUE2U + "=" + value + "; HttpOnly");
196 original = value;
197 } else {
198 response.setHeader("Set-Cookie", UNIQUE2U + "=" + cookie + "; HttpOnly");
199 original = cookie;
200 }
201 }
```

Hidden Field (15 issues)

Abstract

A hidden form field is used.

Explanation

Programmers often trust the contents of hidden fields, expecting that users will not be able to view them or manipulate their contents. Attackers will violate these assumptions. They will examine the values written to hidden fields and alter them or replace the contents with attack data.

Example: An `<input>` tag of type `hidden` indicates the use of a hidden field.

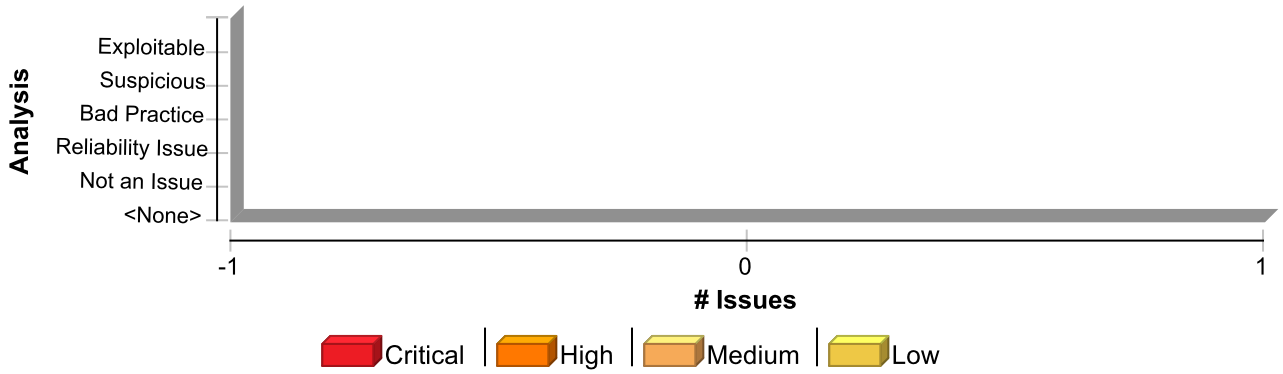
```
<input type="hidden">
```

If hidden fields carry sensitive information, this information will be cached the same way the rest of the page is cached. This can lead to sensitive information being tucked away in the browser cache without the user's knowledge.

Recommendation

Expect that attackers will study and decode all uses of hidden fields in the application. Treat hidden fields as untrusted input. Don't store information in hidden fields if the information should not be cached along with the rest of the page.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Hidden Field	15	15	0	30
Total	**15**	**15**	**0**	**30**

Hidden Field	**Low**
Package: WebContent.lessons.CrossSiteScripting	
ViewProfile.jsp, line 119 (Hidden Field)	**Low**

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Hidden Field — Low

Package: WebContent.lessons.CrossSiteScripting

ViewProfile.jsp, line 119 (Hidden Field) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:119

```
116 {
117 %>
118 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
119 <input type="hidden" name="<%=CrossSiteScripting.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
120 <input type="submit" name="action" value="<%=CrossSiteScripting.LISTSTAFF_ACTION%>"/>
121 </form></td>
122 <%
```

ViewProfile.jsp, line 131 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:131

```
128 {
129 %>
130 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
131 <input type="hidden" name="<%=CrossSiteScripting.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
132 <input type="submit" name="action" value="<%=CrossSiteScripting.EDITPROFILE_ACTION%>"/>
133 </form>
134 <%
```

ViewProfile.jsp, line 144 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:144

```
141 {
142 %>
```

Hidden Field — Low

Package: WebContent.lessons.CrossSiteScripting

ViewProfile.jsp, line 144 (Hidden Field) — Low

```
143 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
144 <input type="hidden" name="<%=CrossSiteScripting.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
145 <input type="submit" name="action" value="<%=CrossSiteScripting.DELETEPROFILE_ACTION%>"/>
146 </form>
147 <%
```

EditProfile.jsp, line 122 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:122

```
119 </td>
120
121 <td width="81">
122 <input name="<%=CrossSiteScripting.EMPLOYEE_ID%>" type="hidden" value="<%=employee.getId()%>">
123 <input name="<%=CrossSiteScripting.TITLE%>" type="hidden" value="<%=employee.getTitle()%>">
124 <input type="submit" name="action" value="<%=CrossSiteScripting.UPDATEPROFILE_ACTION%>"/>
125 </td>
```

EditProfile.jsp, line 123 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:123

```
120
121 <td width="81">
122 <input name="<%=CrossSiteScripting.EMPLOYEE_ID%>" type="hidden" value="<%=employee.getId()%>">
123 <input name="<%=CrossSiteScripting.TITLE%>" type="hidden" value="<%=employee.getTitle()%>">
124 <input type="submit" name="action" value="<%=CrossSiteScripting.UPDATEPROFILE_ACTION%>"/>
125 </td>
126 <td width="211"></td>
```

Hidden Field — Low

Package: WebContent.lessons.RoleBasedAccessControl

ViewProfile.jsp, line 117 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:117

```
114 {
115 %>
116 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
117 <input type="hidden" name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
118 <input type="submit" name="action" value="<%=RoleBasedAccessControl.LISTSTAFF_ACTION%>"/>
119 </form>
120 <%
```

ViewProfile.jsp, line 129 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:129

```
126 {
127 %>
128 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
129 <input type="hidden" name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
130 <input type="submit" name="action" value="<%=RoleBasedAccessControl.EDITPROFILE_ACTION%>"/>
131 </form>
132 <%
```

ViewProfile.jsp, line 142 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

Hidden Field — Low

Package: WebContent.lessons.RoleBasedAccessControl

ViewProfile.jsp, line 142 (Hidden Field) — Low

File: ViewProfile.jsp:142

```
139 {
140 %>
141 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
142 <input type="hidden" name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
143 <input type="submit" name="action" value="<%=RoleBasedAccessControl.DELETEPROFILE_ACTION%>"/>
144 </form>
145 <%
```

EditProfile.jsp, line 125 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:125

```
122 </td>
123
124 <td width="81">
125 <input name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" type="hidden" value="<%=employee.getId()%>">
126 <input name="<%=RoleBasedAccessControl.TITLE%>" type="hidden" value="<%=employee.getTitle()%>">
127 <input type="submit" name="action" value="<%=RoleBasedAccessControl.UPDATEPROFILE_ACTION%>"/>
128 </td>
```

EditProfile.jsp, line 126 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:126

```
123
124 <td width="81">
125 <input name="<%=RoleBasedAccessControl.EMPLOYEE_ID%>" type="hidden" value="<%=employee.getId()%>">
126 <input name="<%=RoleBasedAccessControl.TITLE%>" type="hidden" value="<%=employee.getTitle()%>">
127 <input type="submit" name="action" value="<%=RoleBasedAccessControl.UPDATEPROFILE_ACTION%>"/>
128 </td>
```

Hidden Field — Low

Package: WebContent.lessons.RoleBasedAccessControl

EditProfile.jsp, line 126 (Hidden Field) — Low

```
129 <td width="211"></td>
```

Package: WebContent.lessons.SQLInjection

ViewProfile.jsp, line 113 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:113

```
110 {
111 %>
112 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
113 <input type="hidden" name="<%=SQLInjection.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
114 <input type="submit" name="action" value="<%=SQLInjection.LISTSTAFF_ACTION%>"/>
115 </form>
116 <%
```

ViewProfile.jsp, line 126 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:126

```
123 {
124 %>
125 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
126 <input type="hidden" name="<%=SQLInjection.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
127 <input type="submit" name="action" value="<%=SQLInjection.EDITPROFILE_ACTION%>"/>
128 </form>
129 <%
```

ViewProfile.jsp, line 139 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation

Hidden Field — Low

Package: WebContent.lessons.SQLInjection

ViewProfile.jsp, line 139 (Hidden Field) — Low

Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: ViewProfile.jsp:139

```
136 {
137 %>
138 <form method="POST" action="attack?menu=<%=webSession.getCurrentMenu()%>">
139 <input type="hidden" name="<%=SQLInjection.EMPLOYEE_ID%>" value="<%=employee.getId()%>">
140 <input type="submit" name="action" value="<%=SQLInjection.DELETEPROFILE_ACTION%>"/>
141 </form>
142 <%
```

EditProfile.jsp, line 122 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:122

```
119 </td>
120
121 <td width="81">
122 <input name="<%=SQLInjection.EMPLOYEE_ID%>" type="hidden" value="<%=employee.getId()%>">
123 <input name="<%=SQLInjection.TITLE%>" type="hidden" value="<%=employee.getTitle()%>">
124 <input type="submit" name="action" value="<%=SQLInjection.UPDATEPROFILE_ACTION%>"/>
125 </td>
```

EditProfile.jsp, line 123 (Hidden Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Content)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: EditProfile.jsp:123

Hidden Field	Low

Package: WebContent.lessons.SQLInjection

EditProfile.jsp, line 123 (Hidden Field)	Low

```
120
121 <td width="81">
122 <input name="<%=SQLInjection.EMPLOYEE_ID%>" type="hidden" value="<%=employee.getId()%>">
123 <input name="<%=SQLInjection.TITLE%>" type="hidden" value="<%=employee.getTitle()%>">
124 <input type="submit" name="action" value="<%=SQLInjection.UPDATEPROFILE_ACTION%>"/>
125 </td>
126 <td width="211"></td>
```

Insecure Randomness (1 issue)

Abstract

Standard pseudorandom number generators cannot withstand cryptographic attacks.

Explanation

Insecure randomness errors occur when a function that can produce predictable values is used as a source of randomness in a security-sensitive context.

Computers are deterministic machines, and as such are unable to produce true randomness. Pseudorandom Number Generators (PRNGs) approximate randomness algorithmically, starting with a seed from which subsequent values are calculated.

There are two types of PRNGs: statistical and cryptographic. Statistical PRNGs provide useful statistical properties, but their output is highly predictable and form an easy to reproduce numeric stream that is unsuitable for use in cases where security depends on generated values being unpredictable. Cryptographic PRNGs address this problem by generating output that is more difficult to predict. For a value to be cryptographically secure, it must be impossible or highly improbable for an attacker to distinguish between the generated random value and a truly random value. In general, if a PRNG algorithm is not advertised as being cryptographically secure, then it is probably a statistical PRNG and should not be used in security-sensitive contexts, where its use can lead to serious vulnerabilities such as easy-to-guess temporary passwords, predictable cryptographic keys, session hijacking, and DNS spoofing.

Example: The following code uses a statistical PRNG to create a URL for a receipt that remains active for some period of time after a purchase.

```
String GenerateReceiptURL(String baseUrl) {
    Random ranGen = new Random();
    ranGen.setSeed((new Date()).getTime());
    return (baseUrl + ranGen.nextInt(400000000) + ".html");
}
```

This code uses the `Random.nextInt()` function to generate "unique" identifiers for the receipt pages it generates. Since `Random.nextInt()` is a statistical PRNG, it is easy for an attacker to guess the strings it generates. Although the underlying design of the receipt system is also faulty, it would be more secure if it used a random number generator that did not produce predictable receipt identifiers, such as a cryptographic PRNG.

Recommendation

When unpredictability is critical, as is the case with most security-sensitive uses of randomness, use a cryptographic PRNG. Regardless of the PRNG you choose, always use a value with sufficient entropy to seed the algorithm. (Values such as the current time offer only negligible entropy and should not be used.)

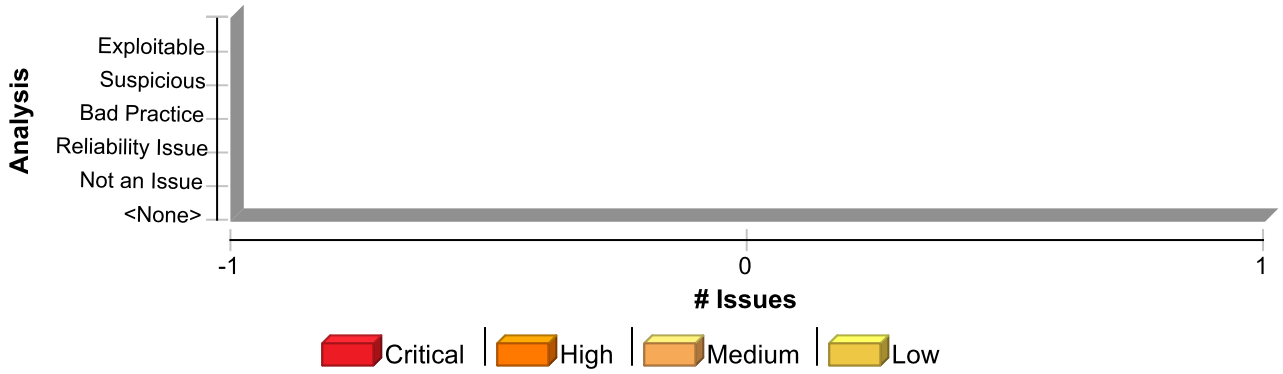
The Java language provides a cryptographic PRNG in `java.security.SecureRandom`. As is the case with other algorithm-based classes in `java.security`, `SecureRandom` provides an implementation-independent wrapper around a particular set of algorithms. When you request an instance of a `SecureRandom` object using `SecureRandom.getInstance()`, you can request a specific implementation of the algorithm. If the algorithm is available, then it is given as a `SecureRandom` object. If it is unavailable or if you do not specify a particular implementation, then you are given a `SecureRandom` implementation selected by the system.

Sun provides a single `SecureRandom` implementation with the Java distribution named `SHA1PRNG`, which Sun describes as computing:

"The SHA-1 hash over a true-random seed value concatenated with a 64-bit counter which is incremented by 1 for each operation. From the 160-bit SHA-1 output, only 64 bits are used [1]."

However, the specifics of the Sun implementation of the `SHA1PRNG` algorithm are poorly documented, and it is unclear what sources of entropy the implementation uses and therefore what amount of true randomness exists in its output. Although there is speculation on the Web about the Sun implementation, there is no evidence to contradict the claim that the algorithm is cryptographically strong and can be used safely in security-sensitive contexts.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Insecure Randomness	1	1	0	2
Total	**1**	**1**	**0**	**2**

Insecure Randomness — High

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 77 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: random()
Enclosing Method: ()
File: WeakSessionID.java:77

```
74
75  protected static List<String> sessionList = new ArrayList<String>();
76
77  protected static long seq = Math.round(Math.random() * 10240) + 10000;
78
79  protected static long lastTime = System.currentTimeMillis();
80
```

J2EE Bad Practices: Leftover Debug Code (4 issues)

Abstract

Debug code can create unintended entry points in a deployed web application.

Explanation

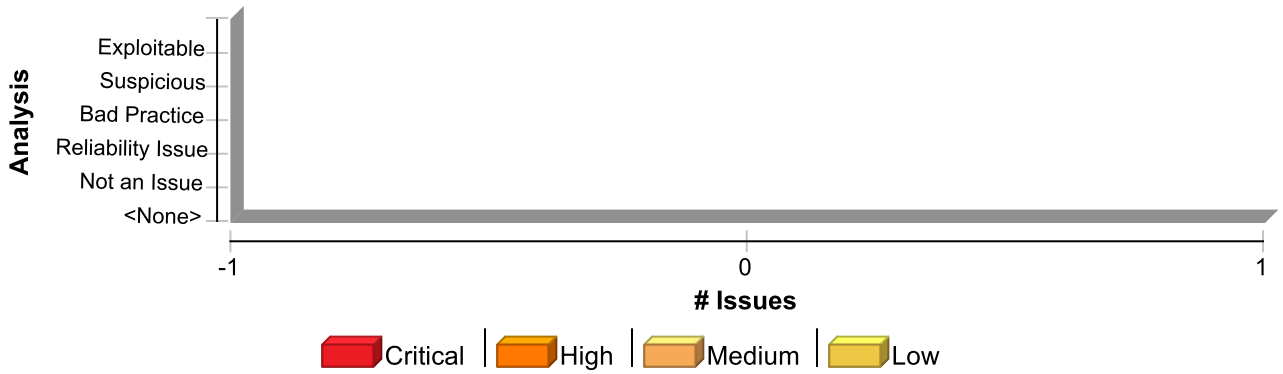
A common development practice is to add "back door" code specifically designed for debugging or testing purposes that is not intended to be shipped or deployed with the application. When this sort of debug code is accidentally left in the application, the application is open to unintended modes of interaction. These back door entry points create security risks because they are not considered during design or testing and fall outside of the expected operating conditions of the application.

The most common example of forgotten debug code is a `main()` method appearing in a web application. Although this is an acceptable practice during product development, classes that are part of a production J2EE application should not define a `main()`.

Recommendation

Remove debug code before deploying a production version of an application. Regardless of whether a direct security threat can be articulated, it is unlikely that there is a legitimate reason for such code to remain in the application after the early stages of development.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: Leftover Debug Code	4	4	0	8
Total	**4**	**4**	**0**	**8**

J2EE Bad Practices: Leftover Debug Code	**Low**
Package: org.owasp.webgoat.lessons	
Encoding.java, line 740 (J2EE Bad Practices: Leftover Debug Code)	**Low**

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

J2EE Bad Practices: Leftover Debug Code — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 740 (J2EE Bad Practices: Leftover Debug Code) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: main
Enclosing Method: main()
File: Encoding.java:740

```
737 * @param args The command line arguments
738 */
739
740 public static void main( String[] args )
741 {
742 try
743 {
```

Package: org.owasp.webgoat.session

CreateDB.java, line 50 (J2EE Bad Practices: Leftover Debug Code) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: main
Enclosing Method: main()
File: CreateDB.java:50

```
47 *
48 * @param args The command line arguments
49 */
50 public static void main(String[] args)
51 {
52
53 CreateDB db = new CreateDB();
```

WebgoatProperties.java, line 112 (J2EE Bad Practices: Leftover Debug Code) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

J2EE Bad Practices: Leftover Debug Code — Low

Package: org.owasp.webgoat.session

WebgoatProperties.java, line 112 (J2EE Bad Practices: Leftover Debug Code) — Low

Sink Details

Sink: Function: main
Enclosing Method: main()
File: WebgoatProperties.java:112

```
109 }
110
111
112 public static void main(String[] args)
113 {
114 WebgoatProperties properties = null;
115 try
```

Package: org.owasp.webgoat.util

Exec.java, line 502 (J2EE Bad Practices: Leftover Debug Code) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: main
Enclosing Method: main()
File: Exec.java:502

```
499 *
500 * @param args The command line arguments
501 */
502 public static void main(String[] args)
503 {
504 ExecResults results;
505 String sep = System.getProperty("line.separator");
```

J2EE Bad Practices: Non-Serializable Object Stored in Session (3 issues)

Abstract

Storing a non-serializable object as an `HttpSession` attribute can damage application reliability.

Explanation

A J2EE application can make use of multiple JVMs in order to improve application reliability and performance. In order to make the multiple JVMs appear as a single application to the end user, the J2EE container can replicate an `HttpSession` object across multiple JVMs so that if one JVM becomes unavailable another can step in and take its place without disrupting the flow of the application.

In order for session replication to work, the values the application stores as attributes in the session must implement the `Serializable` interface.

Example 1: The following class adds itself to the session, but because it is not serializable, the session can no longer be replicated.

```
public class DataGlob {
   String globName;
   String globValue;

   public void addToSession(HttpSession session) {
     session.setAttribute("glob", this);
   }
}
```

Recommendation

In many cases, the easiest way to fix this problem is simply to have the offending object implement the `Serializable` interface.

Example 2: The code in Example 1 could be rewritten in the following way:

```
public class DataGlob implements java.io.Serializable {
   String globName;
   String globValue;

   public void addToSession(HttpSession session) {
     session.setAttribute("glob", this);
   }
}
```

Note that for complex objects, the transitive closure of the objects stored in the session must be serializable. If object A references object B and object A is stored in the session, then both A and B must implement `Serializable`.

While implementing the `Serializable` interface is often easy (since the interface does not force the class to define any methods), some types of objects will cause complications. Watch out for objects that hold references to external resources. For example, both streams and JNI are likely to cause complications.

Example 3: Use type checking to require serializable objects. Instead of this:

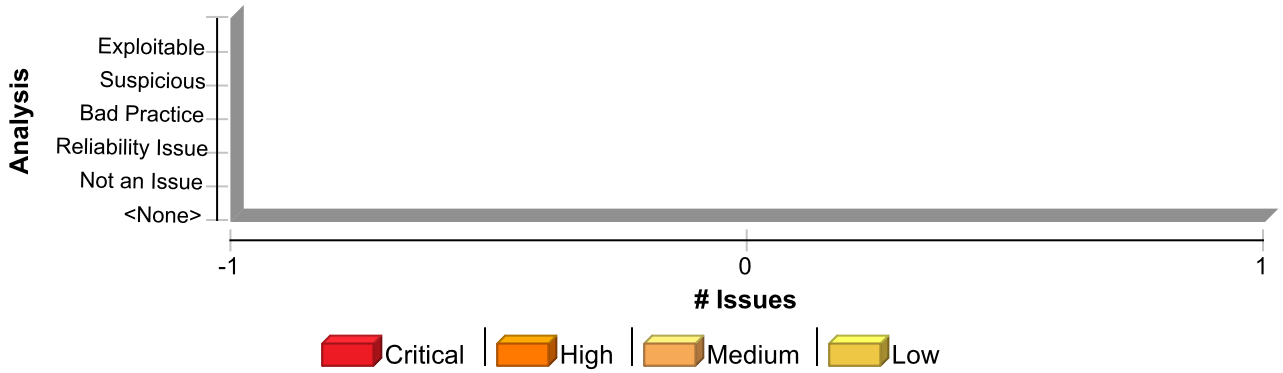
```
public static void addToSession(HttpServletRequest req,
```

```
                    String attrib, Object obj)
{
  HttpSession sess = req.getSession(true);
  sess.setAttribute(attrib, obj);
}
```

write this:

```
public static void addToSession(HttpServletRequest req,
                    String attrib, Serializable ser) {
  HttpSession sess = req.getSession(true);
  sess.setAttribute(attrib, ser);
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: Non-Serializable Object Stored in Session	3	3	0	6
Total	**3**	**3**	**0**	**6**

J2EE Bad Practices: Non-Serializable Object Stored in Session	High
Package: org.owasp.webgoat	
HammerHead.java, line 495 (J2EE Bad Practices: Non-Serializable Object Stored in Session)	**High**

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: setAttribute
Enclosing Method: updateSession()
File: HammerHead.java:495

J2EE Bad Practices: Non-Serializable Object Stored in Session — High

Package: org.owasp.webgoat

HammerHead.java, line 495 (J2EE Bad Practices: Non-Serializable Object Stored in Session) — High

```
492  // Create new custom session and save it in the HTTP session
493  // System.out.println( "HH Creating new WebSession: " );
494  session = new WebSession(this, context);
495  hs.setAttribute(WebSession.SESSION, session);
496  // reset timeout
497  hs.setMaxInactiveInterval(sessionTimeoutSeconds);
498
```

HammerHead.java, line 185 (J2EE Bad Practices: Non-Serializable Object Stored in Session) — High

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: setAttribute
Enclosing Method: doPost()
File: HammerHead.java:185

```
182  }
183  request.setAttribute("client.browser", clientBrowser);
184  request.getSession().setAttribute("websession", mySession);
185  request.getSession().setAttribute("course", mySession.getCourse());
186
187  request.getRequestDispatcher(getViewPage(mySession)).forward(
188  request, response);
```

HammerHead.java, line 184 (J2EE Bad Practices: Non-Serializable Object Stored in Session) — High

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: setAttribute
Enclosing Method: doPost()
File: HammerHead.java:184

```
181  clientBrowser = userAgent;
```

J2EE Bad Practices: Non-Serializable Object Stored in Session	High
Package: org.owasp.webgoat	
HammerHead.java, line 184 (J2EE Bad Practices: Non-Serializable Object Stored in Session)	**High**

```
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
```

J2EE Bad Practices: Sockets (1 issue)

Abstract

Socket-based communication in web applications is prone to error.

Explanation

The J2EE standard permits the use of sockets only for the purpose of communication with legacy systems when no higher-level protocol is available. Authoring your own communication protocol requires wrestling with difficult security issues, including:

- In-band versus out-of-band signaling

- Compatibility between protocol versions

- Channel security

- Error handling

- Network constraints (firewalls)

- Session management

Without significant scrutiny by a security expert, chances are good that a custom communication protocol will suffer from security problems.

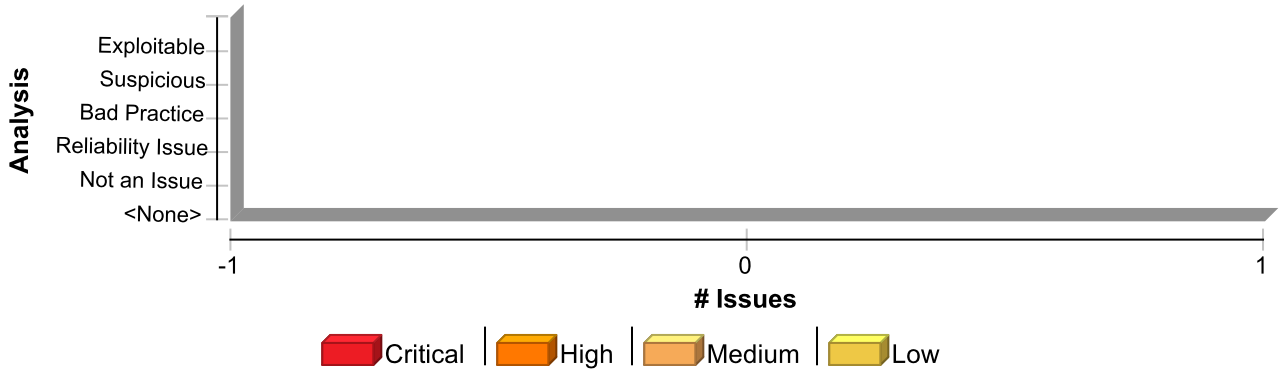
Many of the same issues apply to a custom implementation of a standard protocol. While there are usually more resources available that address security concerns related to implementing a standard protocol, these resources are also available to attackers.

Recommendation

Replace a custom communication protocol with an industry standard protocol or framework. Consider whether you can use a protocol such as HTTP, FTP, SMTP, CORBA, RMI/IIOP, EJB, or SOAP.

Consider the security track record of the protocol implementation you choose.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: Sockets	1	1	0	2
Total	**1**	**1**	**0**	**2**

J2EE Bad Practices: Sockets — Low

Package: org.owasp.webgoat.util

Interceptor.java, line 93 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Socket()
Enclosing Method: doFilter()
File: Interceptor.java:93

```
90 if (osgServerName != null && osgServerName.length() != 0
91 && osgServerPort != null && osgServerPort.length() != 0)
92 {
93 osgSocket = new Socket(osgServerName, Integer
94 .parseInt(osgServerPort));
95 if (osgSocket != null)
96 {
```

J2EE Bad Practices: Threads (6 issues)

Abstract

Thread management in a web application is forbidden in some circumstances and is always highly error prone.

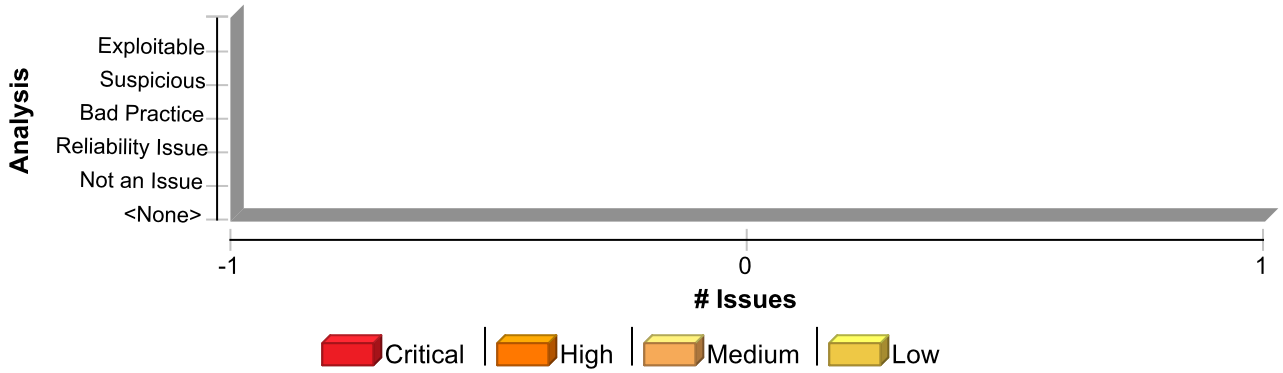
Explanation

Thread management in a web application is forbidden by the J2EE standard in some circumstances and is always highly error prone. Managing threads is difficult and is likely to interfere in unpredictable ways with the behavior of the application container. Even without interfering with the container, thread management usually leads to bugs that are hard to detect and diagnose like deadlock, race conditions, and other synchronization errors.

Recommendation

Avoid managing threads directly from within the web application. Instead use standards such as message driven beans and the EJB timer service that are provided by the application container.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: Threads	6	6	0	12
Total	**6**	**6**	**0**	**12**

J2EE Bad Practices: Threads	Low

Package: org.owasp.webgoat.lessons

ThreadSafetyProblem.java, line 95 (J2EE Bad Practices: Threads)	Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: sleep()
Enclosing Method: createContent()

J2EE Bad Practices: Threads — Low

Package: org.owasp.webgoat.lessons

ThreadSafetyProblem.java, line 95 (J2EE Bad Practices: Threads) — Low

File: ThreadSafetyProblem.java:95

```
92
93  if (!"".equals(currentUser))
94  {
95  Thread.sleep(1500);
96
97  // Get the users info from the DB
98  String query = "SELECT * FROM user_system_data WHERE user_name = '"
```

Package: org.owasp.webgoat.util

ThreadWatcher.java, line 108 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: sleep()
Enclosing Method: run()
File: ThreadWatcher.java:108

```
105  {
106  try
107  {
108  Thread.sleep(myTimeout);
109  }
110  catch (InterruptedException e)
111  {
```

Exec.java, line 303 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: start()
Enclosing Method: execOptions()
File: Exec.java:303

```
300  if (timeout > 0)
```

J2EE Bad Practices: Threads — Low

Package: org.owasp.webgoat.util

Exec.java, line 303 (J2EE Bad Practices: Threads) — Low

```
301 {
302 watcher = new ThreadWatcher(child, interrupted, timeout);
303 new Thread(watcher).start();
304 }
305
306 // Write to the child process' input stream
```

Exec.java, line 114 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: start()
Enclosing Method: execOptions()
File: Exec.java:114

```
111 if (timeout > 0)
112 {
113 watcher = new ThreadWatcher(child, interrupted, timeout);
114 new Thread(watcher).start();
115 }
116
117 // Write to the child process' input stream
```

Exec.java, line 303 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: run()
Enclosing Method: execOptions()
File: Exec.java:303

```
300 if (timeout > 0)
301 {
302 watcher = new ThreadWatcher(child, interrupted, timeout);
303 new Thread(watcher).start();
304 }
```

J2EE Bad Practices: Threads — Low

Package: org.owasp.webgoat.util

Exec.java, line 303 (J2EE Bad Practices: Threads) — Low

```
305
306 // Write to the child process' input stream
```

Exec.java, line 114 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: run()
Enclosing Method: execOptions()
File: Exec.java:114

```
111 if (timeout > 0)
112 {
113 watcher = new ThreadWatcher(child, interrupted, timeout);
114 new Thread(watcher).start();
115 }
116
117 // Write to the child process' input stream
```

J2EE Bad Practices: getConnection() (5 issues)

Abstract

The J2EE standard forbids the direct management of connections.

Explanation

The J2EE standard requires that applications use the container's resource management facilities to obtain connections to resources.

For example, a J2EE application should obtain a database connection as follows:

```
ctx = new InitialContext();
datasource = (DataSource)ctx.lookup(DB_DATASRC_REF);
conn = datasource.getConnection();
```

and should avoid obtaining a connection in this way:

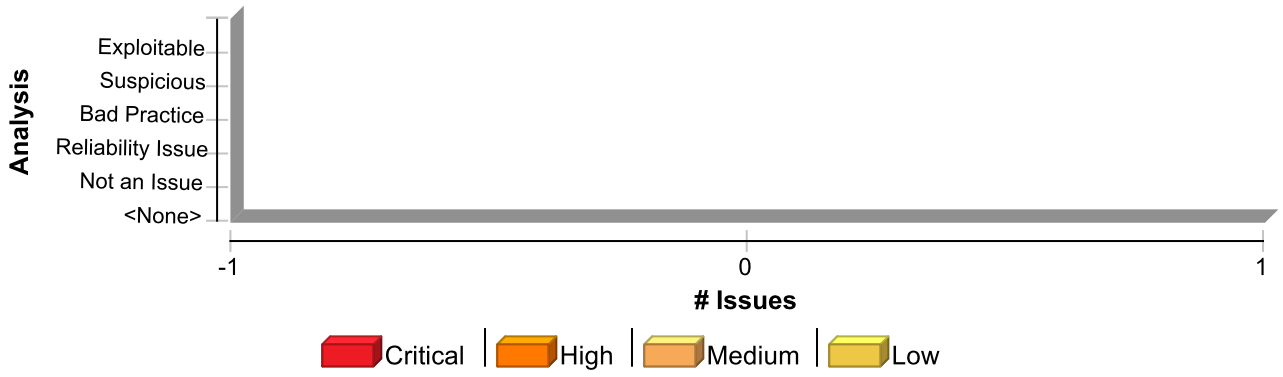
```
conn = DriverManager.getConnection(CONNECT_STRING);
```

Every major web application container provides pooled database connection management as part of its resource management framework. Duplicating this functionality in an application is difficult and error prone, which is part of the reason it is forbidden under the J2EE standard.

Recommendation

Replace direct calls to `DriverManager.getConnection()` with a JNDI lookup of the appropriate connection factory, and obtain a connection from the connection factory.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: getConnection()	5	5	0	10
Total	**5**	**5**	**0**	**10**

J2EE Bad Practices: getConnection() — Low

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 105 (J2EE Bad Practices: getConnection()) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: getConnection()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:105

```
102 {
103 dbName = dbName.concat("database.prp");
104 Class.forName("org.enhydra.instantdb.jdbc.idbDriver");
105 return DriverManager.getConnection("jdbc:idb:" + dbName);
106 }
107 }
108 catch (Exception e)
```

DatabaseUtilities.java, line 98 (J2EE Bad Practices: getConnection()) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: getConnection()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:98

```
95 System.out.println("DBName: " + dbName);
96 Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
97 return DriverManager
98 .getConnection("jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ="
99 + dbName + ";PWD=webgoat");
100 }
101 else
```

DatabaseUtilities.java, line 78 (J2EE Bad Practices: getConnection()) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

J2EE Bad Practices: getConnection() — Low

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 78 (J2EE Bad Practices: getConnection()) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getConnection()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:78

```
75 {
76 Class.forName(driverName);
77
78 return (DriverManager.getConnection(connectionString));
79 }
80
81
```

DatabaseUtilities.java, line 68 (J2EE Bad Practices: getConnection()) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getConnection()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:68

```
65 {
66 Class.forName(s.getDatabaseDriver());
67
68 return (DriverManager.getConnection(s.getDatabaseConnectionString()));
69 }
70
71
```

CreateDB.java, line 70 (J2EE Bad Practices: getConnection()) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

J2EE Bad Practices: getConnection()	Low
Package: org.owasp.webgoat.session	
CreateDB.java, line 70 (J2EE Bad Practices: getConnection())	**Low**

Sink: getConnection()
Enclosing Method: main()
File: CreateDB.java:70

```
67 {
68
69 connection = DriverManager
70 .getConnection(
71 "jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=c:/webgoat.mdb;PWD=webgoat",
72 "webgoat", "webgoat");
73 db.makeDB(connection);
```

J2EE Misconfiguration: Excessive Servlet Mappings (1 issue)

Abstract

Multiple URL patterns map to a single Servlet, which often indicates poor a architecture or a lack of standardization.

Explanation

Multiple URL patterns that map to a single Servlet could be a sign that the Servlet performs too many functions.

Example 1: The following example maps five URL patterns to a single Servlet.

```
<servlet>
    <servlet-class>com.class.MyServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
    <servlet-name>MyServlet</servlet-name>
    <url-pattern>/myservlet</url-pattern>
</servlet-mapping>

<servlet-mapping>
    <servlet-name>MyServlet</servlet-name>
    <url-pattern>/helloworld*</url-pattern>
</servlet-mapping>

<servlet-mapping>
    <servlet-name>MyServlet</servlet-name>
    <url-pattern>/servlet*</url-pattern>
</servlet-mapping>

<servlet-mapping>
    <servlet-name>MyServlet</servlet-name>
    <url-pattern>/mservlet*</url-pattern>
</servlet-mapping>

<servlet-mapping>
    <servlet-name>MyServlet</servlet-name>
    <url-pattern>/*</url-pattern>
</servlet-mapping>
```

This category was derived from the Cigital Java Rulepack. http://www.cigital.com/

Recommendation

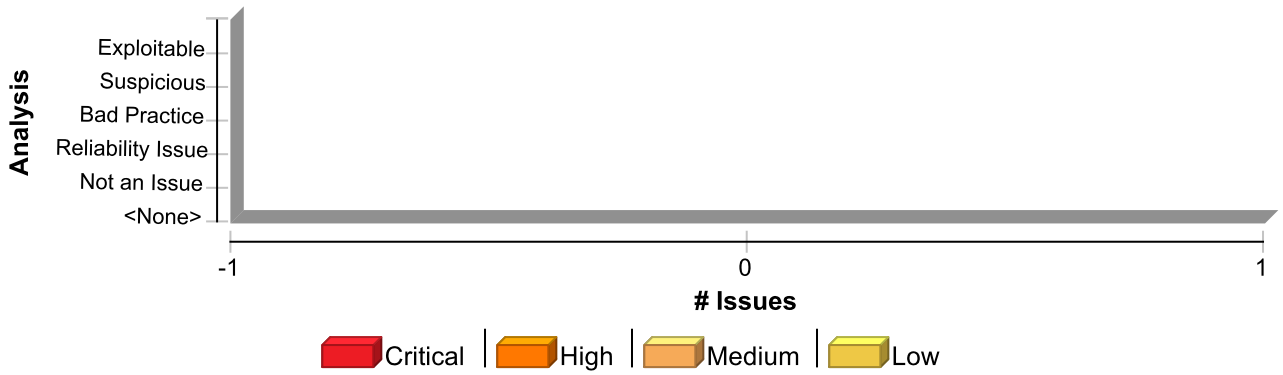
Split up the functionality of any Servlet that performs several functions into separate Servlets, each with its own URL mapping and singular functionality. This is especially important in Servlets that perform privileged or otherwise sensitive operations because it reduces the number of paths of entry to that functionality.

Example 2: The following example maps one URL pattern to a single Servlet.

```
<servlet>
    <servlet-name>MyServlet</servlet-name>
    <servlet-class>com.class.MyServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
```

```
    <servlet-name>MyServlet</servlet-name>
    <url-pattern>/myservlet</url-pattern>
</servlet-mapping>
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Misconfiguration: Excessive Servlet Mappings	1	1	0	2
Total	**1**	**1**	**0**	**2**

J2EE Misconfiguration: Excessive Servlet Mappings — Low

Package: WebContent.WEB-INF

web.xml, line 200 (J2EE Misconfiguration: Excessive Servlet Mappings) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:200

```
197 -->
198
199
200 <servlet-mapping>
201 <servlet-name>AxisServlet</servlet-name>
202 <url-pattern>/servlet/AxisServlet</url-pattern>
203 </servlet-mapping>
```

J2EE Misconfiguration: Excessive Session Timeout (1 issue)

Abstract

An overly long session timeout gives attackers more time to potentially compromise user accounts.

Explanation

The longer a session stays open, the larger the window of opportunity an attacker has to compromise user accounts. While a session remains active, an attacker may be able to brute-force a user's password, crack a user's wireless encryption key, or commandeer a session from an open browser. Longer session timeouts can also prevent memory from being released and eventually result in a denial of service if a sufficiently large number of sessions are created.

Example 1: If the session timeout is zero or less than zero, the session never expires. The following example shows a session timeout set to -1, which will cause the session to remain active indefinitely.

```
<session-config>
    <session-timeout>-1</session-timeout>
</session-config>
```

The `<session-timeout>` tag defines the default session timeout interval for all sessions in the web application. If the `<session-timeout>` tag is missing, it is left to the container to set the default timeout. This category was derived from the Cigital Java Rulepack. http://www.cigital.com/

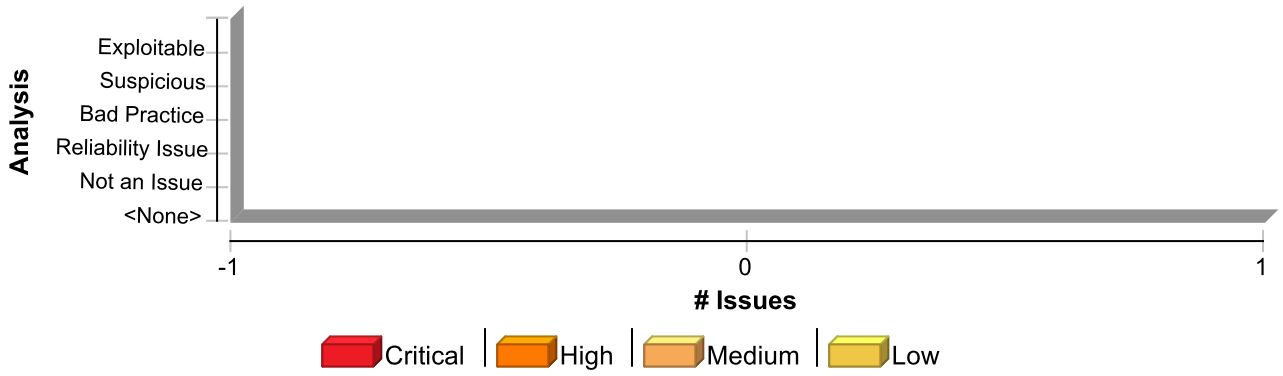
Recommendation

Set a session timeout that is 30 minutes or less, which both allows users to interact with the application over a period of time and provides a reasonable bound for the window of attack.

Example 2: The following example sets the session timeout to 20 minutes.

```
<session-config>
  <session-timeout>20</session-timeout>
</session-config>
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Misconfiguration: Excessive Session Timeout	1	1	0	2
Total	**1**	**1**	**0**	**2**

J2EE Misconfiguration: Excessive Session Timeout — Low

Package: WebContent.WEB-INF

web.xml, line 248 (J2EE Misconfiguration: Excessive Session Timeout) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: web.xml:248

```
245 the timeout for a particular session dynamically by using
246 HttpSession.getMaxInactiveInterval(). -->
247
248 <session-config>
249 <!-- 2 days -->
250 <session-timeout>2880</session-timeout>
251 </session-config>
```

J2EE Misconfiguration: Missing Data Transport Constraint (2 issues)

Abstract

A security constraint that does not specify a user data constraint cannot guarantee that restricted resources will be protected at the transport layer.

Explanation

`web.xml` security constraints are typically used for role based access control, but the optional `user-data-constraint` element specifies a transport guarantee that prevents content from being transmitted insecurely.

Within the `<user-data-constraint>` tag, the `<transport-guarantee>` tag defines how communication should be handled. There are three levels of transport guarantee:

1) `NONE` means that the application does not require any transport guarantees. 2) `INTEGRAL` means that the application requires that data sent between the client and server be sent in such a way that it cannot be changed in transit. 3) `CONFIDENTIAL` means that the application requires that data be transmitted in a fashion that prevents other entities from observing the contents of the transmission.

In most circumstances, the use of `INTEGRAL` or `CONFIDENTIAL` means that SSL/TLS is required. If the `<user-data-constraint>` and `<transport-guarantee>` tags are omitted, the transport guarantee defaults to `NONE`.

Example 1: The following security constraint does not specify a transport guarantee.

```
<security-constraint>
    <web-resource-collection>
        <web-resource-name>Storefront</web-resource-name>
        <description>Allow Customers and Employees access to online store
front</description>
        <url-pattern>/store/shop/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <description>Anyone</description>
        <role-name>anyone</role-name>
    </auth-constraint>
</security-constraint>
```

This category was derived from the Cigital Java Rulepack. http://www.cigital.com/

Recommendation

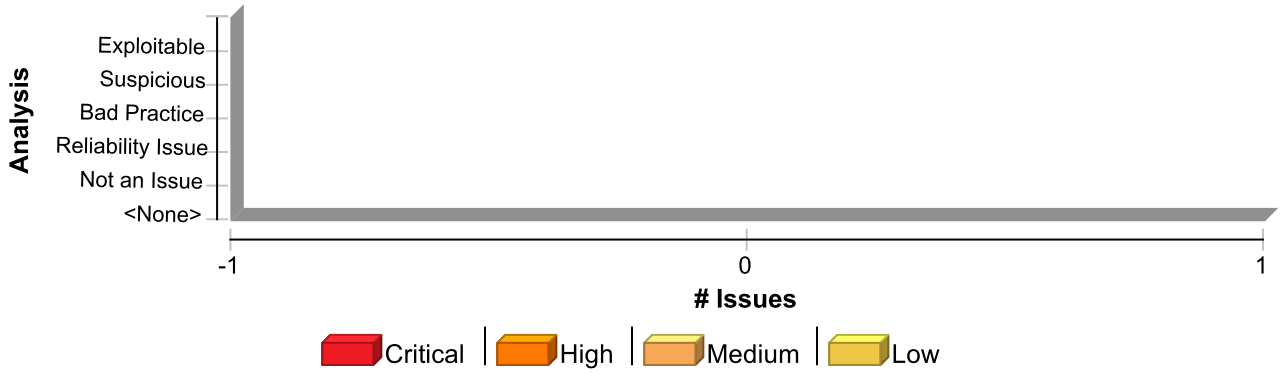
Specify a `CONFIDENTIAL` transport guarantee whenever you define a authorization constraint. Once you decide to encrypt traffic to any part of your application, do not make the mistake of allowing unencrypted traffic to other parts of the application, which could allow session cookies or other sensitive information to be transmitted over insecure channels.

Example 2: The following security constraint specifies a `CONFIDENTIAL` transport guarantee.

```
<security-constraint>
    <web-resource-collection>
        <web-resource-name>Storefront</web-resource-name>
        <description>Allow Customers and Employees access to online store
front</description>
        <url-pattern>/store/shop/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <description>Anyone</description>
```

```
        <role-name>anyone</role-name>
    </auth-constraint>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Misconfiguration: Missing Data Transport Constraint	2	2	0	4
Total	**2**	**2**	**0**	**4**

J2EE Misconfiguration: Missing Data Transport Constraint — Low

Package: WebContent.WEB-INF

web.xml, line 274 (J2EE Misconfiguration: Missing Data Transport Constraint) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:274

```
271
272
273 <!-- Define a Security Constraint on this Application -->
274 <security-constraint>
275 <web-resource-collection>
276 <web-resource-name>WebGoat Application</web-resource-name>
277 <url-pattern>/*</url-pattern>
```

web.xml, line 286 (J2EE Misconfiguration: Missing Data Transport Constraint) — Low

Issue Details

J2EE Misconfiguration: Missing Data Transport Constraint — Low

Package: WebContent.WEB-INF

web.xml, line 286 (J2EE Misconfiguration: Missing Data Transport Constraint) — Low

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction	Not Predicted

Sink Details

File: web.xml:286

```
283 </auth-constraint>
284 </security-constraint>
285
286 <security-constraint>
287 <web-resource-collection>
288 <web-resource-name>WebGoat Application Source</web-resource-name>
289 <url-pattern>/JavaSource/*</url-pattern>
```

J2EE Misconfiguration: Missing Error Handling (3 issues)

Abstract

A web application must define default error pages in order to prevent attackers from mining information from the application container's built-in error response.

Explanation

When an attacker explores a web site looking for vulnerabilities, the amount of information that the site provides is crucial to the eventual success or failure of any attempted attacks. If the application shows the attacker a stack trace, it relinquishes information that makes the attacker's job significantly easier. For example, a stack trace might show the attacker a malformed SQL query string, the type of database being used, and the version of the application container. This information enables the attacker to target known vulnerabilities in these components.

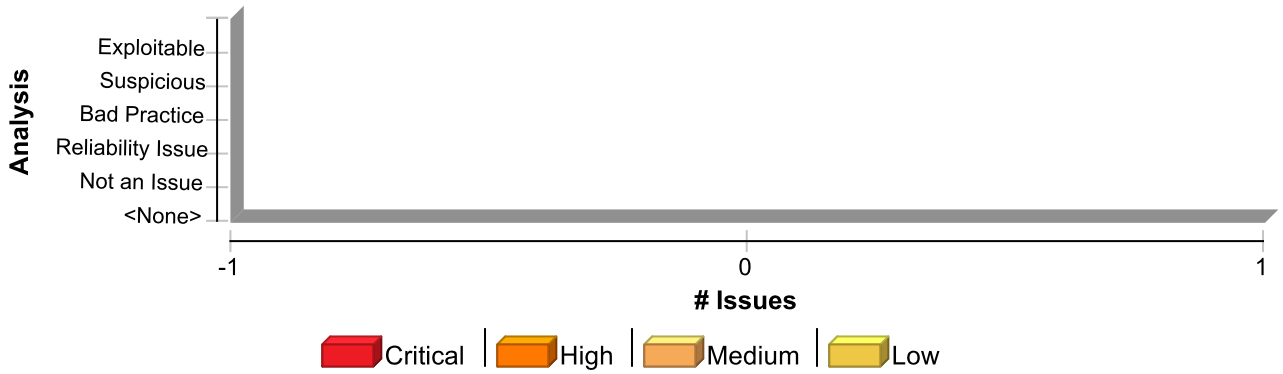
The application configuration should specify a default error page in order to guarantee that the application will never leak error messages to an attacker. Handling standard HTTP error codes is useful and user-friendly in addition to being a good security practice, and a good configuration will also define a last-chance error handler that catches any exception that could possibly be thrown by the application.

Recommendation

A web application must be configured with a default error page. Your `web.xml` should include at least the following entries:

```
<error-page>
   <exception-type>java.lang.Throwable</exception-type>
<location>/error.jsp</location>
</error-page>
<error-page>
   <error-code>404</error-code>
<location>/error.jsp</location>
</error-page>
<error-page>
   <error-code>500</error-code>
<location>/error.jsp</location>
</error-page>
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Misconfiguration: Missing Error Handling	3	3	0	6
Total	**3**	**3**	**0**	**6**

J2EE Misconfiguration: Missing Error Handling — Low

Package: WebContent.WEB-INF

web.xml, line 6 (J2EE Misconfiguration: Missing Error Handling) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: /web-app
File: web.xml:6

```
3  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
4  "http://java.sun.com/dtd/web-app_2_3.dtd">
5
6  <web-app>
7
8  <!-- General description of your web application -->
9  <display-name>WebGoat</display-name>
```

web-unix.xml, line 6 (J2EE Misconfiguration: Missing Error Handling) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: /web-app
File: web-unix.xml:6

```
3  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
4  "http://java.sun.com/dtd/web-app_2_3.dtd">
5
6  <web-app>
7
8  <!-- General description of your web application -->
9  <display-name>WebGoat</display-name>
```

J2EE Misconfiguration: Missing Error Handling — Low

Package: WebContent.WEB-INF

web-windows.xml, line 6 (J2EE Misconfiguration: Missing Error Handling) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: /web-app
File: web-windows.xml:6

```
3  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
4  "http://java.sun.com/dtd/web-app_2_3.dtd">
5
6  <web-app>
7
8  <!-- General description of your web application -->
9  <display-name>WebGoat</display-name>
```

J2EE Misconfiguration: Missing Servlet Mapping (1 issue)

Abstract

A Servlet defined in `web.xml` cannot be accessed without a corresponding servlet mapping.

Explanation

The absence of a valid servlet mapping prevents all access to the unmapped servlet.

Example 1: The following entry from `web.xml` defines `ExampleServlet` but fails to define a corresponding servlet mapping.

```
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/
xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <servlet>
      <servlet-name>ExampleServlet</servlet-name>
      <servlet-class>com.class.ExampleServlet</servlet-class>
      <load-on-startup>1</load-on-startup>
    </servlet>

</web-app>
```

This category is from the Cigital Java Rulepack. http://www.cigital.com/

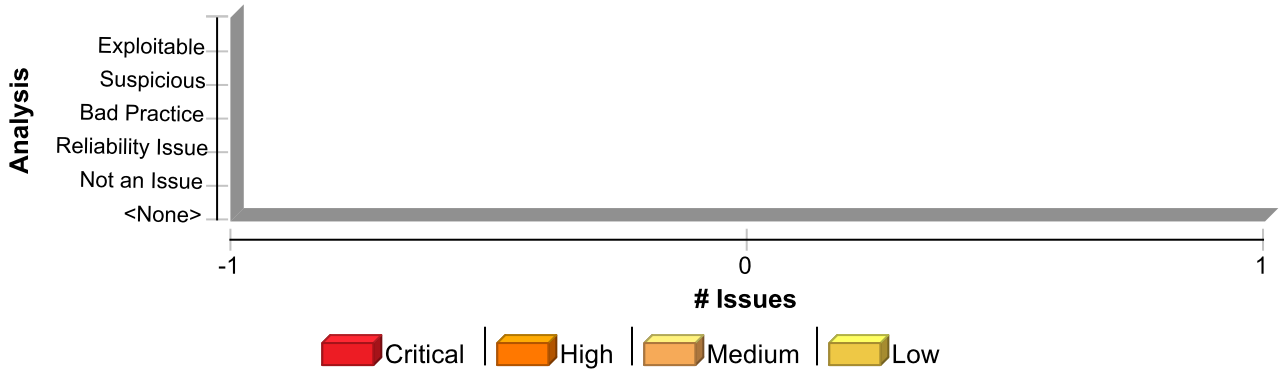
Recommendation

Ensure that every `<servlet>` has a corresponding `<servlet-mapping>`.

Example 2: The following entry from `web.xml` defines `ExampleServlet` and a corresponding servlet mapping.

```
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/
xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">
    <servlet>
        <servlet-name>ExampleServlet</servlet-name>
        <servlet-class>com.class.ExampleServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>ExampleServlet</servlet-name>
        <url-pattern>/exampleservlet</url-pattern>
    </servlet-mapping>

    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>/action</url-pattern>
    </servlet-mapping>
</web-app>
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Misconfiguration: Missing Servlet Mapping	1	1	0	2
Total	**1**	**1**	**0**	**2**

J2EE Misconfiguration: Missing Servlet Mapping — Low

Package: WebContent.WEB-INF

web.xml, line 79 (J2EE Misconfiguration: Missing Servlet Mapping) — Low

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: web.xml:79

```
76 </servlet-class>
77 </servlet>
78
79 <servlet>
80 <servlet-name>AdminServlet</servlet-name>
81 <display-name>Axis Admin Servlet</display-name>
82 <servlet-class>
```

Key Management: Hardcoded Encryption Key (3 issues)

Abstract

Hardcoded encryption keys may compromise system security in a way that cannot be easily remedied.

Explanation

It is never a good idea to hardcode an encryption key because it allows all of the project's developers to view the encryption key, and makes fixing the problem extremely difficult. Once the code is in production, the encryption key cannot be changed without patching the software. If the account that is protected by the encryption key is compromised, the owners of the system will be forced to choose between security and availability.

Example 1: The following code uses a hardcoded encryption key:

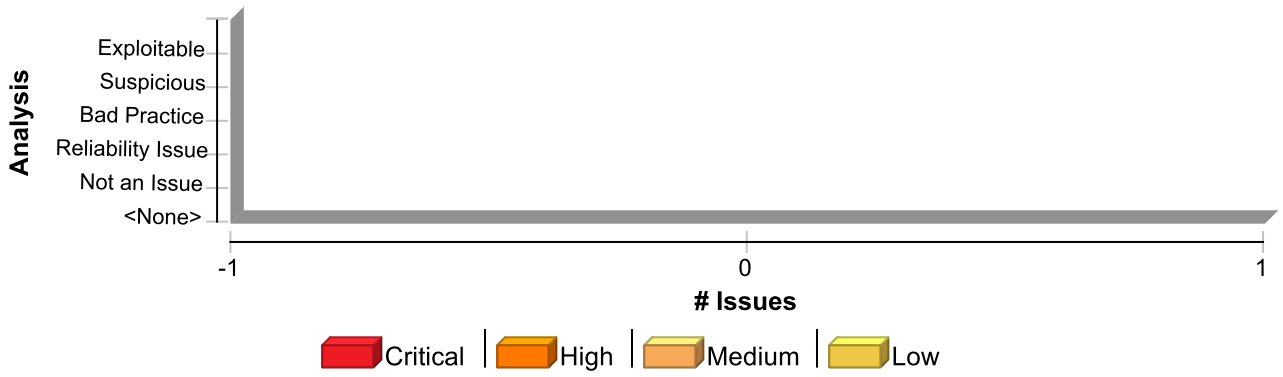
```
...
private static final String encryptionKey = "lakdsljkalkjlksdfkl";
byte[] keyBytes = encryptionKey.getBytes();
SecretKeySpec key = new SecretKeySpec(keyBytes, "AES");
Cipher encryptCipher = Cipher.getInstance("AES");
encryptCipher.init(Cipher.ENCRYPT_MODE, key);
...
```

Anyone who has access to the code will have access to the encryption key. Once the application has shipped, there is no way to change the encryption key unless the program is patched. An employee with access to this information could use it to break into the system. Even worse, if attackers had access to the executable for the application, they could extract the encryption key value.

Recommendation

Encryption keys should never be hardcoded and should be obfuscated and managed in an external source. Storing encryption keys in plain text anywhere on the system allows anyone with sufficient permissions to read and potentially misuse the encryption key.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Key Management: Hardcoded Encryption Key	3	3	0	6
Total	**3**	**3**	**0**	**6**

Key Management: Hardcoded Encryption Key — Critical

Package: org.owasp.webgoat.lessons

DOMInjection.java, line 57 (Key Management: Hardcoded Encryption Key) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: KEY
Enclosing Method: ()
File: DOMInjection.java:57

```
54
55 private final static Integer DEFAULT_RANKING = new Integer(10);
56
57 private final static String KEY = "key";
58
59 private final static IMG MAC_LOGO = new IMG("images/logos/macadamian.gif").setAlt(
60 "Macadamian Technologies").setBorder(0).setHspace(0).setVspace(0);
```

DOMInjection.java, line 65 (Key Management: Hardcoded Encryption Key) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: VariableAccess: key
Enclosing Method: createContent()
File: DOMInjection.java:65

```
62 protected Element createContent(WebSession s)
63 {
64
65 String key = "K1JFWP8BSO8HI52LNPQS8F5L01N";
66 ElementContainer ec = new ElementContainer();
67
68 try
```

Encoding.java, line 77 (Key Management: Hardcoded Encryption Key) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Key Management: Hardcoded Encryption Key — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 77 (Key Management: Hardcoded Encryption Key) — Critical

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: KEY
Enclosing Method: ()
File: Encoding.java:77

```
74
75  private final static String INPUT = "input";
76
77  private final static String KEY = "key";
78
79  // local encoders
80
```

Log Forging (12 issues)

Abstract

Writing unvalidated user input to log files can allow an attacker to forge log entries or inject malicious content into the logs.

Explanation

Log forging vulnerabilities occur when:

1. Data enters an application from an untrusted source.

2. The data is written to an application or system log file.

Applications typically use log files to store a history of events or transactions for later review, statistics gathering, or debugging. Depending on the nature of the application, the task of reviewing log files may be performed manually on an as-needed basis or automated with a tool that automatically culls logs for important events or trending information.

Interpretation of the log files may be hindered or misdirected if an attacker can supply data to the application that is subsequently logged verbatim. In the most benign case, an attacker may be able to insert false entries into the log file by providing the application with input that includes appropriate characters. If the log file is processed automatically, the attacker may be able to render the file unusable by corrupting the format of the file or injecting unexpected characters. A more subtle attack might involve skewing the log file statistics. Forged or otherwise, corrupted log files can be used to cover an attacker's tracks or even to implicate another party in the commission of a malicious act [1]. In the worst case, an attacker may inject code or other commands into the log file and take advantage of a vulnerability in the log processing utility [2].

Example 1: The following web application code attempts to read an integer value from a request object. If the value fails to parse as an integer, then the input is logged with an error message indicating what happened.

```
...
    String val = request.getParameter("val");
    try {
        int value = Integer.parseInt(val);
    }
    catch (NumberFormatException nfe) {
        log.info("Failed to parse val = " + val);
    }
...
```

If a user submits the string "`twenty-one`" for `val`, the following entry is logged:

```
INFO: Failed to parse val=twenty-one
```

However, if an attacker submits the string "`twenty-one%0a%0aINFO:+User+logged+out%3dbadguy`", the following entry is logged:

```
INFO: Failed to parse val=twenty-one

INFO: User logged out=badguy
```

Clearly, attackers may use this same mechanism to insert arbitrary log entries.

Some think that in the mobile world, classic web application vulnerabilities, such as log forging, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 2: The following code adapts Example 1 to the Android platform.

```
...
    String val = this.getIntent().getExtras().getString("val");
    try {
        int value = Integer.parseInt();
    }
    catch (NumberFormatException nfe) {
        Log.e(TAG, "Failed to parse val = " + val);
        }
...
```

Recommendation

Prevent log forging attacks with indirection: create a set of legitimate log entries that correspond to different events that must be logged and only log entries from this set. To capture dynamic content, such as users logging out of the system, always use server-controlled values rather than user-supplied data. This ensures that the input provided by the user is never used directly in a log entry.

Example 1 can be rewritten to use a pre-defined log entry that corresponds to a `NumberFormatException` as follows:

```
...
    public static final String NFE = "Failed to parse val. The input is
required to be an integer value."
...
    String val = request.getParameter("val");
        try {
                int value = Integer.parseInt(val);
        }
        catch (NumberFormatException nfe) {
                log.info(NFE);
        }
..
```

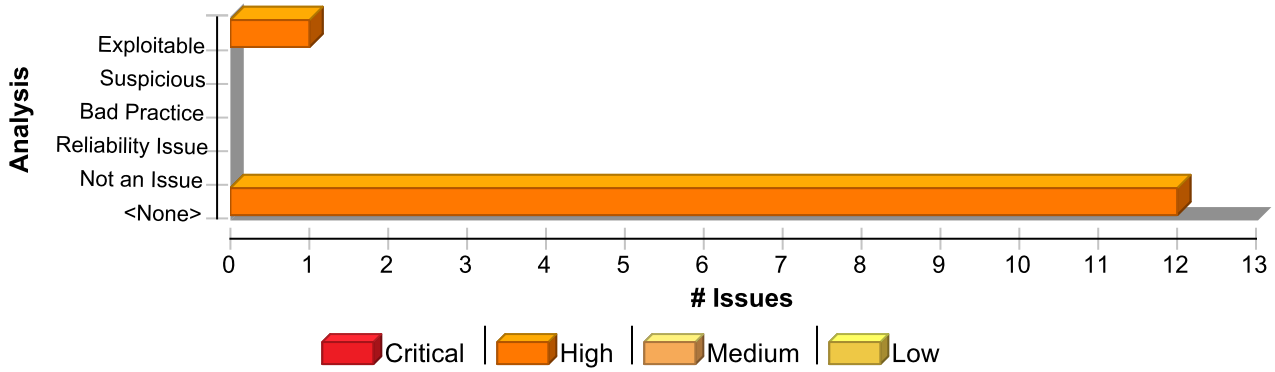
And here is an Android equivalent:

```
...
    public static final String NFE = "Failed to parse val. The input is
required to be an integer value."
...
        String val = this.getIntent().getExtras().getString("val");
        try {
                int value = Integer.parseInt();
        }
        catch (NumberFormatException nfe) {
                Log.e(TAG, NFE);
        }
```

. . .

In some situations this approach is impractical because the set of legitimate log entries is too large or complicated. In these situations, developers often fall back on blacklisting. Blacklisting selectively rejects or escapes potentially dangerous characters before using the input. However, a list of unsafe characters can quickly become incomplete or outdated. A better approach is to create a whitelist of characters that are allowed to appear in log entries and accept input composed exclusively of characters in the approved set. The most critical character in most log forging attacks is the '\n' (newline) character, which should never appear on a log entry whitelist.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Log Forging	12	12	0	24
Total	**12**	**12**	**0**	**24**

Log Forging — High

URL: /attack

HammerHead.java, line 306 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getParameterValues
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:593
URL: /attack

Log Forging — High

URL: /attack

HammerHead.java, line 306 (Log Forging) — High

```
590  return (null);
591  }
592
593  return request.getParameterValues(name);
594  }
595
596
```

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: log()
File: HammerHead.java:306
Taint Flags: WEB, XSS

```
303  {
304  String output = new Date() + " | " + request.getRemoteHost() + ":"
305  + request.getRemoteAddr() + " | " + message;
306  log(output);
307  System.out.println(output);
308  }
309
```

HammerHead.java, line 306 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterNames()
From: org.owasp.webgoat.session.ParameterParser.getParameterNames
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:576
URL: /attack

```
573  return (null);
574  }
575
576  return request.getParameterNames();
577  }
578
579
```

Log Forging — High

URL: /attack

HammerHead.java, line 306 (Log Forging) — High

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: log()
File: HammerHead.java:306
Taint Flags: PRIMARY_KEY, WEB, XSS

```
303 {
304 String output = new Date() + " | " + request.getRemoteHost() + ":"
305 + request.getRemoteAddr() + " | " + message;
306 log(output);
307 System.out.println(output);
308 }
309
```

HammerHead.java, line 205 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177
URL: /attack

```
174 + mySession.getParser().toString());
175
176 // Redirect the request to our View servlet
177 String userAgent = request.getHeader("user-agent");
178 String clientBrowser = "Not known!";
179 if (userAgent != null)
180 {
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:205
Taint Flags: WEB, XSS

```
202 catch (Throwable thr)
```

Log Forging — High

URL: /attack

HammerHead.java, line 205 (Log Forging) — High

```
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
```

HammerHead.java, line 205 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177
URL: /attack

```
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
177  String userAgent = request.getHeader("user-agent");
178  String clientBrowser = "Not known!";
179  if (userAgent != null)
180  {
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:205
Taint Flags: WEB, XSS

```
202  catch (Throwable thr)
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
```

Log Forging — High

URL: /attack

HammerHead.java, line 205 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177
URL: /attack

```
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
177  String userAgent = request.getHeader("user-agent");
178  String clientBrowser = "Not known!";
179  if (userAgent != null)
180  {
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:205
Taint Flags: WEB, XSS

```
202  catch (Throwable thr)
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
```

HammerHead.java, line 173 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Log Forging — High

URL: /attack

HammerHead.java, line 173 (Log Forging) — High

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177
URL: /attack

```
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
177  String userAgent = request.getHeader("user-agent");
178  String clientBrowser = "Not known!";
179  if (userAgent != null)
180  {
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:173
Taint Flags: WEB, XSS

```
170  // log the access to this screen for this user
171  UserTracker userTracker = UserTracker.instance();
172  userTracker.update(mySession, screen);
173  log(request, screen.getClass().getName() + " | "
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
```

HammerHead.java, line 173 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65
URL: /attack

```
62 undefined
63 undefined
```

Log Forging — High

URL: /attack

HammerHead.java, line 173 (Log Forging) — High

```
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:173
Taint Flags: WEB, XSS

```
170  // log the access to this screen for this user
171  UserTracker userTracker = UserTracker.instance();
172  userTracker.update(mySession, screen);
173  log(request, screen.getClass().getName() + " | "
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
```

HammerHead.java, line 205 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65
URL: /attack

```
62 undefined
63 undefined
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Log Forging — High

URL: /attack

HammerHead.java, line 205 (Log Forging) — High

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:205
Taint Flags: WEB, XSS

```
202 catch (Throwable thr)
203 {
204 thr.printStackTrace();
205 log(request, "Could not write error screen: "
206 + thr.getMessage());
207 }
208 // System.out.println( "HH Leaving doPost: " );
```

HammerHead.java, line 205 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65
URL: /attack

```
62 undefined
63 undefined
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:205
Taint Flags: WEB, XSS

```
202 catch (Throwable thr)
203 {
204 thr.printStackTrace();
205 log(request, "Could not write error screen: "
```

Log Forging — High

URL: /attack

HammerHead.java, line 205 (Log Forging) — High

```
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
```

HammerHead.java, line 205 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65
URL: /attack

```
62 undefined
63 undefined
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: HammerHead.java:205
Taint Flags: WEB, XSS

```
202  catch (Throwable thr)
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
```

URL: /source

LessonSource.java, line 105 (Log Forging) — High

Issue Details

Log Forging — High

URL: /source

LessonSource.java, line 105 (Log Forging) — High

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65
URL: /source

```
62 undefined
63 undefined
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: LessonSource.java:105
Taint Flags: WEB, XSS

```
102 catch (Throwable thr)
103 {
104 thr.printStackTrace();
105 log(request, "Could not write error screen: "
106 + thr.getMessage());
107 }
108 //System.out.println( "Leaving doPost: " );
```

LessonSource.java, line 105 (Log Forging) — High

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Log Forging — High

URL: /source

LessonSource.java, line 105 (Log Forging) — High

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177
URL: /source

```
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
177  String userAgent = request.getHeader("user-agent");
178  String clientBrowser = "Not known!";
179  if (userAgent != null)
180  {
```

Sink Details

Sink: org.owasp.webgoat.HammerHead.log()
Enclosing Method: doPost()
File: LessonSource.java:105
Taint Flags: WEB, XSS

```
102  catch (Throwable thr)
103  {
104  thr.printStackTrace();
105  log(request, "Could not write error screen: "
106  + thr.getMessage());
107  }
108  //System.out.println( "Leaving doPost: " );
```

Missing Check against Null (4 issues)

Abstract

The program can dereference a null pointer because it does not check the return value of a function that might return null.

Explanation

Just about every serious attack on a software system begins with the violation of a programmer's assumptions. After the attack, the programmer's assumptions seem flimsy and poorly founded, but before an attack many programmers would defend their assumptions well past the end of their lunch break.

Two dubious assumptions that are easy to spot in code are "this function call can never fail" and "it doesn't matter if this function call fails". When a programmer ignores the return value from a function, they implicitly state that they are operating under one of these assumptions.

Example 1: The following code does not check to see if the string returned by `getParameter()` is `null` before calling the member function `compareTo()`, potentially causing a null dereference.

```
String itemName = request.getParameter(ITEM_NAME);
    if (itemName.compareTo(IMPORTANT_ITEM)) {
        ...
    }
    ...
```

Example 2:. The following code shows a system property that is set to `null` and later dereferenced by a programmer who mistakenly assumes it will always be defined.

```
System.clearProperty("os.name");
...
String os = System.getProperty("os.name");
if (os.equalsIgnoreCase("Windows 95") )
    System.out.println("Not supported");
```

The traditional defense of this coding error is:

"I know the requested value will always exist because.... If it does not exist, the program cannot perform the desired behavior so it doesn't matter whether I handle the error or simply allow the program to die dereferencing a null value."

But attackers are skilled at finding unexpected paths through programs, particularly when exceptions are involved.

Recommendation

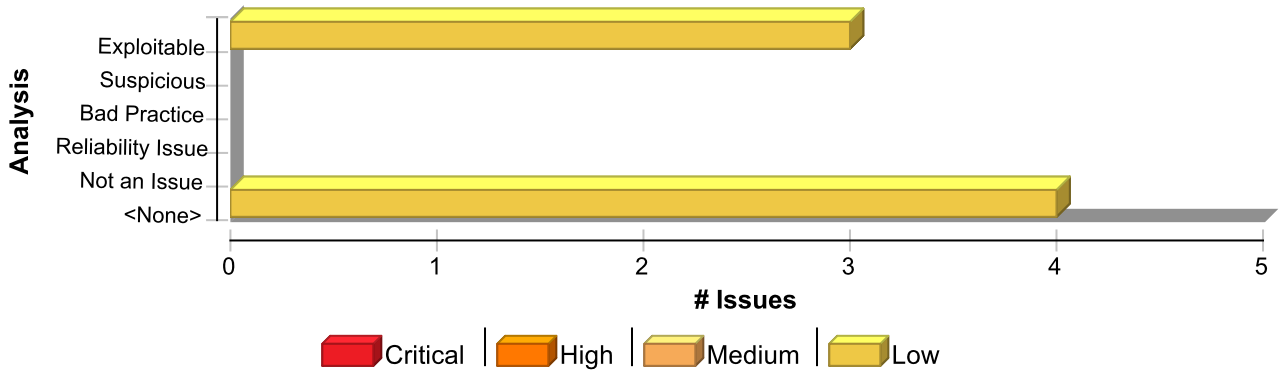
If a function can return an error code or any other evidence of its success or failure, always check for the error condition, even if there is no obvious way for it to occur. In addition to preventing security errors, many initially mysterious bugs have eventually led back to a failed method call with an unchecked return value.

Create an easy to use and standard way for dealing with failure in your application. If error handling is straightforward, programmers will be less inclined to omit it. One approach to standardized error handling is to write wrappers around commonly-used functions that check and handle error conditions without additional programmer intervention. When wrappers are implemented and adopted, the use of non-wrapped equivalents can be prohibited and enforced by using custom rules.

Example 3: The following code implements a wrapper around `getParameter()` that checks the return value of `getParameter()` against `null` and uses a default value if the requested parameter is not defined.

```
String safeGetParameter (HttpRequest request, String name)
{
    String value = request.getParameter(name);
    if (value == null) {
        return getDefaultValue(name)
    }
    return value;
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Missing Check against Null	4	4	0	8
Total	**4**	**4**	**0**	**8**

Missing Check against Null — Low

Package: org.owasp.webgoat.lessons

HttpOnly.java, line 299 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: displayed = getParameter(...) : ServletRequest.getParameter may return NULL
Enclosing Method: handleReadAction()
File: HttpOnly.java:299

```
296
297 private void handleReadAction(WebSession s) {
298
```

Missing Check against Null — Low

Package: org.owasp.webgoat.lessons

HttpOnly.java, line 299 (Missing Check against Null) — Low

```
299 String displayed = s.getRequest().getParameter(READ_RESULT);
300
301 if(httpOnly == true) {
302 if(displayed.indexOf(UNIQUE2U) != -1) {
```

Challenge2Screen.java, line 644 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: osName = getProperty(?) : System.getProperty may return NULL
Enclosing Method: getNetstatResults()
File: Challenge2Screen.java:644

```
641
642 String protocol = s.getParser().getRawParameter(PROTOCOL, "tcp");
643
644 String osName = System.getProperty("os.name");
645 ExecResults er = null;
646 if (osName.indexOf("Windows") != -1)
647 {
```

WSDLScanning.java, line 201 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Missing Check against Null — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 201 (Missing Check against Null) — Low

Sink Details

Sink: fields = getParameterValues(...) : ParameterParser.getParameterValues may return NULL
Enclosing Method: createContent()
File: WSDLScanning.java:201

```
198
199 try
200 {
201 String[] fields = s.getParser().getParameterValues("field");
202 int id = s.getParser().getIntParameter("id");
203 if (connection == null)
204 {
```

Package: org.owasp.webgoat.session

ParameterParser.java, line 1072 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: getParameterValues(...) : ParameterParser.getParameterValues may return NULL
Enclosing Method: toString()
File: ParameterParser.java:1072

```
1069 while (e.hasMoreElements())
1070 {
1071 String key = (String) e.nextElement();
1072 s.append(key + "=" + getParameterValues(key)[0]);
1073
1074 // FIXME: Other values?
1075 if (e.hasMoreElements())
```

Missing Check for Null Parameter (2 issues)

Abstract

This function violates the contract that it must compare its parameter with null.

Explanation

The Java standard requires that implementations of `Object.equals()`, `Comparable.compareTo()`, and `Comparator.compare()` must return a specified value if their parameters are null. Failing to follow this contract may result in unexpected behavior.

Example 1: The following implementation of the `equals()` method does not compare its parameter with null.

```
public boolean equals(Object object)
{
   return (toString().equals(object.toString()));
}
```

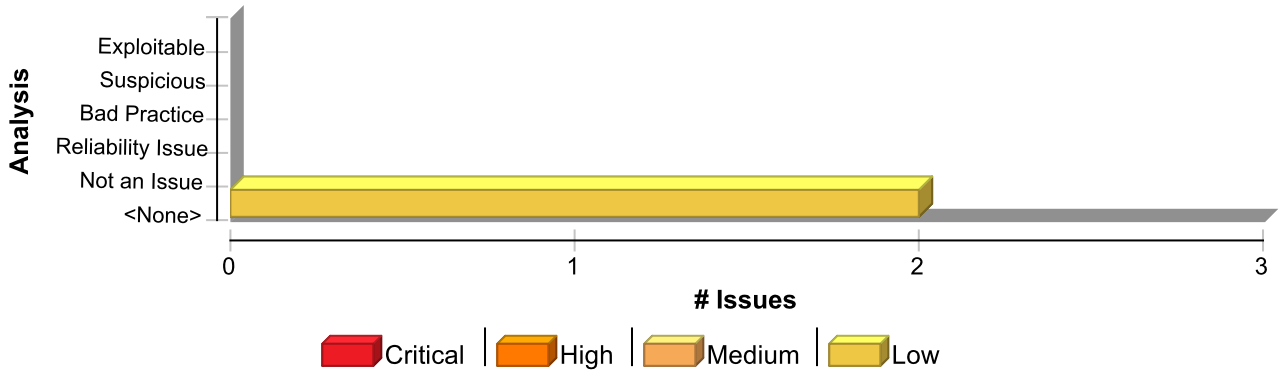
Recommendation

Always follow the contract that `Object.equals()` must return `null` if it receives a null parameter.

Example 2: The previous example is rewritten to explicitly check for a null argument and return false if one is found.

```
public boolean equals(Object object)
{
    if (object == null)
        return false;
    return (toString().equals(object.toString()));
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Missing Check for Null Parameter	2	2	0	4
Total	**2**	**2**	**0**	**4**

Missing Check for Null Parameter — Low

Package: org.owasp.webgoat.lessons

Category.java, line 81 (Missing Check for Null Parameter) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: obj = #param(0)
Enclosing Method: equals()
File: Category.java:81

```
78 }
79
80
81 public boolean equals(Object obj)
82 {
83 return getName().equals(((Category) obj).getName());
84 }
```

AbstractLesson.java, line 317 (Missing Check for Null Parameter) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: obj = #param(0)
Enclosing Method: equals()
File: AbstractLesson.java:317

```
314 * Description of the Parameter
315 * @return Description of the Return Value
316 */
317 public boolean equals(Object obj)
318 {
319 return this.getScreenId() == ((AbstractLesson) obj).getScreenId();
320 }
```

Null Dereference (8 issues)

Abstract

The program can potentially dereference a null pointer, thereby causing a null pointer exception.

Explanation

Null pointer exceptions usually occur when one or more of the programmer's assumptions is violated. A dereference-after-store error occurs when a program explicitly sets an object to `null` and dereferences it later. This error is often the result of a programmer initializing a variable to `null` when it is declared.

Most null pointer issues result in general software reliability problems, but if attackers can intentionally trigger a null pointer dereference, they can use the resulting exception to bypass security logic or to cause the application to reveal debugging information that will be valuable in planning subsequent attacks.

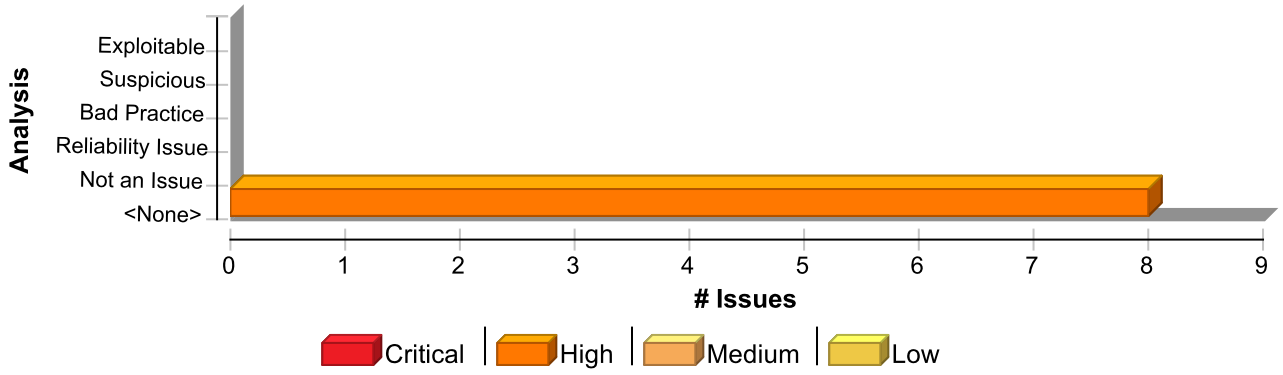
Example: In the following code, the programmer explicitly sets the variable `foo` to `null`. Later, the programmer dereferences `foo` before checking the object for a null value.

```
Foo foo = null;
...
foo.setBar(val);
...
}
```

Recommendation

Implement careful checks before dereferencing objects that might be null. When possible, abstract null checks into wrappers around code that manipulates resources to ensure that they are applied in all cases and to minimize the places where mistakes can occur.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Null Dereference	8	8	0	16
Total	**8**	**8**	**0**	**16**

Null Dereference — High

Package: org.owasp.webgoat

LessonSource.java, line 134 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : realSession
Enclosing Method: updateSession_DELETEME()
File: LessonSource.java:134

```
131 {
132 realSession = (WebSession) o;
133 }
134 session.setCurrentScreen(realSession.getCurrentScreen());
135 session.setCourse(realSession.getCourse());
136 session.setRequest(request);
137
```

Package: org.owasp.webgoat.lessons

Encoding.java, line 674 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : md
Enclosing Method: hashSHA()
File: Encoding.java:674

```
671 // it's got to be there
672 e.printStackTrace();
673 }
674 return ( base64Encode( md.digest() ) );
675 }
676
677
```

Null Dereference — High

Package: org.owasp.webgoat.lessons

Encoding.java, line 648 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : md
Enclosing Method: hashMD5()
File: Encoding.java:648

```
645 // it's got to be there
646 e.printStackTrace();
647 }
648 return ( base64Encode( md.digest() ) );
649 }
650
651
```

XPATHInjection.java, line 185 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : t2
Enclosing Method: createContent()
File: XPATHInjection.java:185

```
182 tr.addElement(new TD().addElement(arrTokens[1]));
183 tr.addElement(new TD().addElement(arrTokens[2]));
184 tr.addElement(new TD().addElement(arrTokens[4]));
185 t2.addElement(tr);
186
187 }
188 if (nodes.getLength() > 1)
```

Null Dereference — High

Package: org.owasp.webgoat.session

LessonTracker.java, line 401 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : out
Enclosing Method: store()
File: LessonTracker.java:401

```
398 {
399 try
400 {
401 out.close();
402 }
403 catch (Exception e)
404 {}
```

LessonTracker.java, line 260 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : in
Enclosing Method: load()
File: LessonTracker.java:260

```
257 {
258 try
259 {
260 in.close();
261 }
262 catch (Exception e)
263 {}
```

CreateDB.java, line 88 (Null Dereference) — High

Issue Details

Null Dereference — High

Package: org.owasp.webgoat.session

CreateDB.java, line 88 (Null Dereference) — High

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : connection
Enclosing Method: main()
File: CreateDB.java:88

```
85
86 try
87 {
88 Statement answer_statement = connection.createStatement(
89 ResultSet.TYPE_SCROLL_INSENSITIVE,
90 ResultSet.CONCUR_READ_ONLY);
91 ResultSet answer_results = answer_statement.executeQuery(query);
```

WebgoatProperties.java, line 124 (Null Dereference) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : properties
Enclosing Method: main()
File: WebgoatProperties.java:124

```
121 System.out.println("Error loading properties");
122 e.printStackTrace();
123 }
124 System.out.println(properties.getProperty("CommandInjection.category"));
125 }
126
127 }
```

Object Model Violation: Just one of equals() and hashCode() Defined (2 issues)

Abstract

This class overrides only one of `equals()` and `hashCode()`.

Explanation

Java objects are expected to obey a number of invariants related to equality. One of these invariants is that equal objects must have equal hashcodes. In other words, if `a.equals(b) == true` then `a.hashCode() == b.hashCode()`.

Failure to uphold this invariant is likely to cause trouble if objects of this class are stored in a collection. If the objects of the class in question are used as a key in a Hashtable or if they are inserted into a Map or Set, it is critical that equal objects have equal hashcodes.

Example 1: The following class overrides `equals()` but not `hashCode()`.

```
public class halfway() {
  public boolean equals(Object obj) {
    ...
  }
}
```

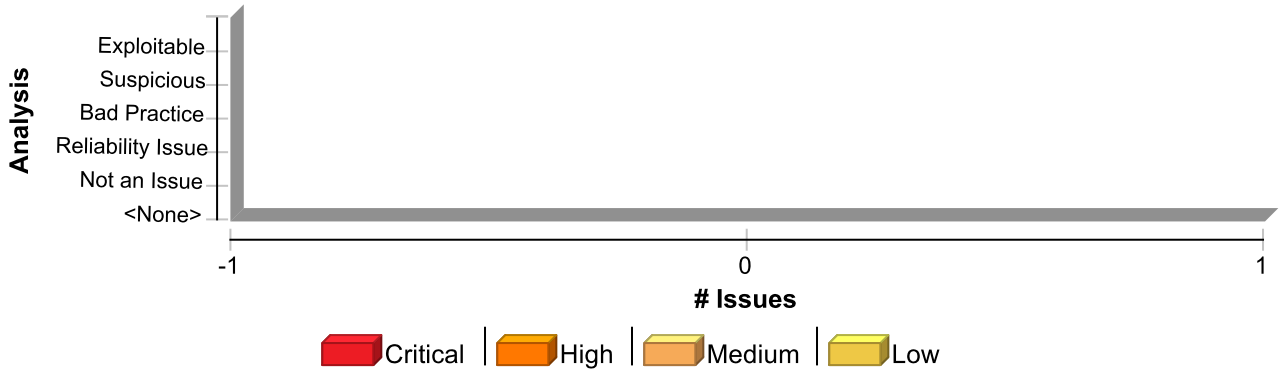
Recommendation

The FindBugs documentation recommends the following simple "starter" implementation of `hashCode()` [1]. It is highly inefficient, but it will produce correct results. If you do not believe that `hashCode()` is important for your program, consider using this implementation.

Example 2: The code in Example 1 could be rewritten in the following way:

```
public class halfway() {
  public boolean equals(Object obj) {
    ...
  }

public int hashCode() {
    assert false : "hashCode not designed";
    return 42; // any arbitrary constant will do
  }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Object Model Violation: Just one of equals() and hashCode() Defined	2	2	0	4
Total	**2**	**2**	**0**	**4**

Object Model Violation: Just one of equals() and hashCode() Defined — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 317 (Object Model Violation: Just one of equals() and hashCode() Defined) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: equals
Enclosing Method: equals()
File: AbstractLesson.java:317

```
314 * Description of the Parameter
315 * @return Description of the Return Value
316 */
317 public boolean equals(Object obj)
318 {
319 return this.getScreenId() == ((AbstractLesson) obj).getScreenId();
320 }
```

Category.java, line 81 (Object Model Violation: Just one of equals() and hashCode() Defined) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Object Model Violation: Just one of equals() and hashCode() Defined	Low
Package: org.owasp.webgoat.lessons	
Category.java, line 81 (Object Model Violation: Just one of equals() and hashCode() Defined)	**Low**

Sink Details

Sink: Function: equals
Enclosing Method: equals()
File: Category.java:81

```
78 }
79
80
81 public boolean equals(Object obj)
82 {
83 return getName().equals(((Category) obj).getName());
84 }
```

Open Redirect (5 issues)

Abstract

Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Explanation

Redirects allow web applications to direct users to different pages within the same application or to external sites. Applications utilize redirects to aid in site navigation and, in some cases, to track how users exit the site. Open redirect vulnerabilities occur when a web application redirects clients to any arbitrary URL that can be controlled by an attacker.

Attackers may utilize open redirects to trick users into visiting a URL to a trusted site and redirecting them to a malicious site. By encoding the URL, an attacker is able to make it more difficult for end-users to notice the malicious destination of the redirect, even when it is passed as a URL parameter to the trusted site. Open redirects are often abused as part of phishing scams to harvest sensitive end-user data.

Example 1: The following JSP code instructs the user's browser to open a URL parsed from the `dest` request parameter when a user clicks the link.

```
<%
    ...
    String strDest = request.getParameter("dest");
    pageContext.forward(strDest);
    ...
%>
```

If a victim received an email instructing the user to follow a link to "http://trusted.example.com/ecommerce/redirect.asp?dest=www.wilyhacker.com", the user would likely click on the link believing they would be transferred to the trusted site. However, when the user clicks the link, the code above will redirect the browser to "http://www.wilyhacker.com".

Many users have been educated to always inspect URLs they receive in emails to make sure the link specifies a trusted site they know. However, if the attacker Hex encoded the destination url as follows: "http://trusted.example.com/ecommerce/redirect.asp?dest=%77%69%6C%79%68%61%63%6B%65%72%2E%63%6F%6D"

then even a savvy end-user may be fooled into following the link.

Recommendation

Unvalidated user input should not be allowed to control the destination URL in a redirect. Instead, use a level of indirection: create a list of legitimate URLs that users are allowed to specify and only allow users to select from the list. With this approach, input provided by users is never used directly to specify a URL for redirects.

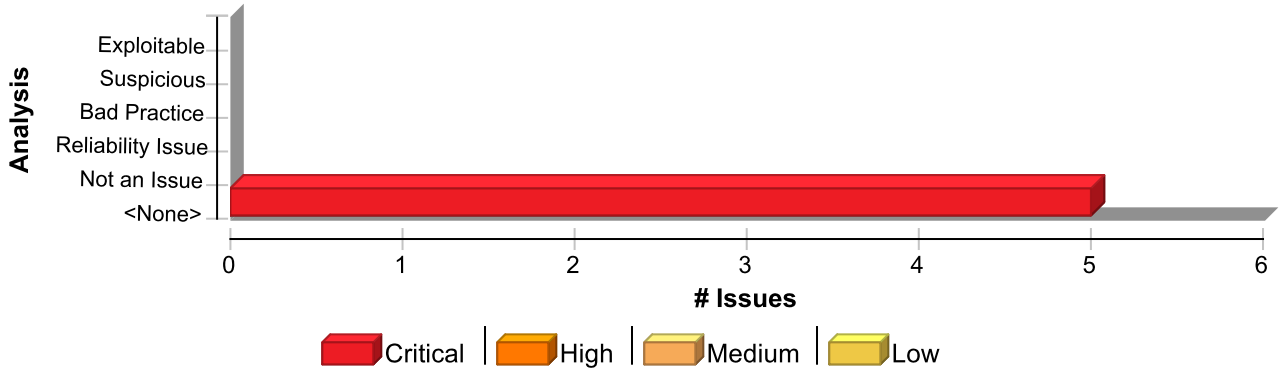
Example 2: The following code references an array populated with valid URLs. The link the user clicks passes in the array index that corresponds to the desired URL.

```
<%
    ...
    try {
        int strDest = Integer.parseInt(request.getParameter("dest"));
        if((strDest >= 0) && (strDest <= strURLArray.length -1 ))
        {
```

```
                strFinalURL = strURLArray[strDest];
                pageContext.forward(strFinalURL);
            }
        }
        catch (NumberFormatException nfe) {
            // Handle exception
            ...
        }
        ...
%>
```

In some situations this approach is impractical because the set of legitimate URLs is too large or too hard to keep track of. In such cases, use a similar approach to restrict the domains that users can be redirected to, which can at least prevent attackers from sending users to malicious external sites.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Open Redirect	5	5	0	10
Total	**5**	**5**	**0**	**10**

Open Redirect	**Critical**
Package: /lessons/ConfManagement	
config.jsp, line 12 (Open Redirect)	**Critical**

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/ConfManagement.config.jsp._jspService
File: WebContent/lessons/ConfManagement/config.jsp:12

Open Redirect — Critical

Package: /lessons/ConfManagement

config.jsp, line 12 (Open Redirect) — Critical

```
9  <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&succeeded=yes");
14 %>
15
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: _jspService()
File: config.jsp:12
Taint Flags: WEB, XSS

```
9  <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&succeeded=yes");
14 %>
15
```

config.jsp, line 12 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/ConfManagement.config.jsp._jspService
File: WebContent/lessons/ConfManagement/config.jsp:11

```
8  </head>
9  <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&succeeded=yes");
14 %>
```

Open Redirect — Critical

Package: /lessons/ConfManagement

config.jsp, line 12 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: _jspService()
File: config.jsp:12
Taint Flags: WEB, XSS

```
9 <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&succeeded=yes");
14 %>
15
```

Package: /lessons/General

redirect.jsp, line 12 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/General.redirect.jsp._jspService
File: WebContent/lessons/General/redirect.jsp:13

```
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&fromRedirect=yes&language=" + request.getParameter("language"));
14 %>
15 </body>
16 </html>
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: _jspService()
File: redirect.jsp:12
Taint Flags: WEB, XSS

```
9 <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
```

Open Redirect — Critical

Package: /lessons/General

redirect.jsp, line 12 (Open Redirect) — Critical

```
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&fromRedirect=yes&language=" + request.getParameter("language"));
14 %>
15 </body>
```

redirect.jsp, line 12 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/General.redirect.jsp._jspService
File: WebContent/lessons/General/redirect.jsp:11

```
8 </head>
9 <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&fromRedirect=yes&language=" + request.getParameter("language"));
14 %>
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: _jspService()
File: redirect.jsp:12
Taint Flags: WEB, XSS

```
9 <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&fromRedirect=yes&language=" + request.getParameter("language"));
14 %>
15 </body>
```

redirect.jsp, line 12 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: /lessons/General

redirect.jsp, line 12 (Open Redirect) — Critical

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: /lessons/General.redirect.jsp._jspService
File: WebContent/lessons/General/redirect.jsp:12

```
9  <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&fromRedirect=yes&language=" + request.getParameter("language"));
14 %>
15 </body>
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: _jspService()
File: redirect.jsp:12
Taint Flags: WEB, XSS

```
9  <body>
10 <% response.sendRedirect("/WebGoat/attack?" +
11  "Screen=" + request.getParameter("Screen") +
12  "&menu=" + request.getParameter("menu") +
13  "&fromRedirect=yes&language=" + request.getParameter("language"));
14 %>
15 </body>
```

Password Management: Empty Password (2 issues)

Abstract

Empty passwords may compromise system security in a way that cannot be easily remedied.

Explanation

It is never a good idea to assign an empty string to a password variable. If the empty password is used to successfully authenticate against another system, then the corresponding account's security is likely compromised because it accepts an empty password. If the empty password is merely a placeholder until a legitimate value can be assigned to the variable, then it can confuse anyone unfamiliar with the code and potentially cause problems on unexpected control flow paths.

Example 1: The code below attempts to connect to a database with an empty password.

```
...
DriverManager.getConnection(url, "scott", "");
...
```

If the code in Example 1 succeeds, it indicates that the database user account "scott" is configured with an empty password, which can be easily guessed by an attacker. Even worse, once the program has shipped, updating the account to use a non-empty password will require a code change.

Example 2: The code below initializes a password variable to an empty string, attempts to read a stored value for the password, and compares it against a user-supplied value.

```
...
String storedPassword = "";
String temp;

if ((temp = readPassword()) != null) {
    storedPassword = temp;
}

if(storedPassword.equals(userPassword))
    // Access protected resources
    ...
}
...
```

If `readPassword()` fails to retrieve the stored password due to a database error or another problem, then an attacker could trivially bypass the password check by providing an empty string for `userPassword`.

In the mobile environment, password management is especially important given that there is such a high chance of device loss. **Example 3:** The code below initializes username and password variables to empty strings, reads credentials from an Android WebView store if they have not been previously rejected by the server for the current request, and uses them to setup authentication for viewing protected pages.

```
...
webview.setWebViewClient(new WebViewClient() {
  public void onReceivedHttpAuthRequest(WebView view,
        HttpAuthHandler handler, String host, String realm) {
    String username = "";
    String password = "";
```

```
    if (handler.useHttpAuthUsernamePassword()) {
      String[] credentials = view.getHttpAuthUsernamePassword(host, realm);
      username = credentials[0];
      password = credentials[1];
    }
    handler.proceed(username, password);
  }
});
...
```

Similar to Example 2, if `useHttpAuthUsernamePassword()` returns `false`, an attacker will be able to view protected pages by supplying an empty password.

Recommendation

Always read stored password values from encrypted, external resources and assign password variables meaningful values. Ensure that sensitive resources are never protected with empty or null passwords.

For Android, as well as any other platform that uses SQLite database, SQLCipher is a good alternative. SQLCipher is an extension to the SQLite database that provides transparent 256-bit AES encryption of database files. Thus, credentials can be stored in an encrypted database.

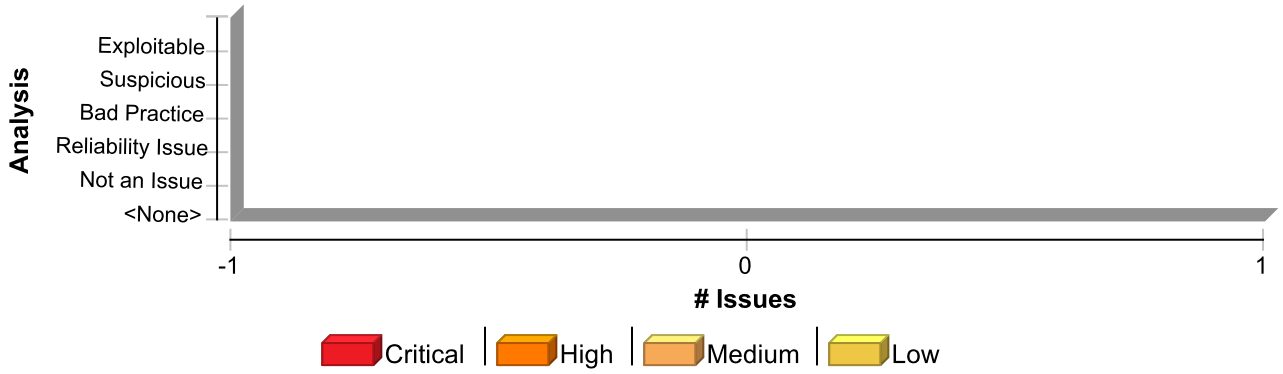
Example 4: The code below demonstrates how to integrate SQLCipher into an Android application after downloading the necessary binaries, and store credentials into the database file.

```
import net.sqlcipher.database.SQLiteDatabase;
...
  SQLiteDatabase.loadLibs(this);
  File dbFile = getDatabasePath("credentials.db");
  dbFile.mkdirs();
  dbFile.delete();
  SQLiteDatabase db = SQLiteDatabase.openOrCreateDatabase(dbFile,
"credentials", null);
  db.execSQL("create table credentials(u, p)");
  db.execSQL("insert into credentials(u, p) values(?, ?)", new Object[]
{username, password});
...
```

Note that references to `android.database.sqlite.SQLiteDatabase` are substituted with those of `net.sqlcipher.database.SQLiteDatabase`.

To enable encryption on the WebView store, WebKit has to be re-compiled with the `sqlcipher.so` library.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Password Management: Empty Password	2	2	0	4
Total	**2**	**2**	**0**	**4**

Password Management: Empty Password — High

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 87 (Password Management: Empty Password) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: VariableAccess: password
Enclosing Method: createContent()
File: DOS_Login.java:87

```
84 try
85 {
86 String username = "";
87 String password = "";
88 username = s.getParser().getRawParameter(USERNAME);
89 password = s.getParser().getRawParameter(PASSWORD);
90
```

FailOpenAuthentication.java, line 68 (Password Management: Empty Password) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Password Management: Empty Password	High
Package: org.owasp.webgoat.lessons	
FailOpenAuthentication.java, line 68 (Password Management: Empty Password)	**High**

Sink: VariableAccess: password
Enclosing Method: createContent()
File: FailOpenAuthentication.java:68

```
65 try
66 {
67 String username = "";
68 String password = "";
69
70 try
71 {
```

Password Management: Hardcoded Password (20 issues)

Abstract

Hardcoded passwords may compromise system security in a way that cannot be easily remedied.

Explanation

It is never a good idea to hardcode a password. Not only does hardcoding a password allow all of the project's developers to view the password, it also makes fixing the problem extremely difficult. Once the code is in production, the password cannot be changed without patching the software. If the account protected by the password is compromised, the owners of the system will be forced to choose between security and availability.

Example 1: The following code uses a hardcoded password to connect to a database:

```
...
DriverManager.getConnection(url, "scott", "tiger");
...
```

This code will run successfully, but anyone who has access to it will have access to the password. Once the program has shipped, there is likely no way to change the database user "scott" with a password of "tiger" unless the program is patched. An employee with access to this information could use it to break into the system. Even worse, if attackers have access to the bytecode for the application they can use the `javap -c` command to access the disassembled code, which will contain the values of the passwords used. The result of this operation might look something like the following for the example above:

```
javap -c ConnMngr.class

22: ldc   #36; //String jdbc:mysql://ixne.com/rxsql
24: ldc   #38; //String scott
26: ldc   #17; //String tiger
```

In the mobile environment, password management is especially important given that there is such a high chance of device loss. **Example 2:** The code below uses hardcoded username and password to setup authentication for viewing protected pages with Android's WebView.

```
...
webview.setWebViewClient(new WebViewClient() {
  public void onReceivedHttpAuthRequest(WebView view,
        HttpAuthHandler handler, String host, String realm) {
    handler.proceed("guest", "allow");
  }
});
...
```

Similar to Example 1, this code will run successfully, but anyone who has access to it will have access to the password.

Recommendation

Passwords should never be hardcoded and should generally be obfuscated and managed in an external source. Storing passwords in plain text anywhere on the system allows anyone with sufficient permissions to read and potentially misuse the password. At the very least, passwords should be hashed before being stored.

Some third-party products claim the ability to manage passwords in a more secure way. For example, WebSphere Application Server 4.x uses a simple XOR encryption algorithm for obfuscating values, but be skeptical about such facilities. WebSphere and other application servers offer outdated and relatively weak encryption mechanisms that are insufficient for security-sensitive environments. For a secure generic solution, the best option today appears to be a proprietary mechanism that you create.

For Android, as well as any other platform that uses SQLite database, SQLCipher is a good alternative. SQLCipher is an extension to the SQLite database that provides transparent 256-bit AES encryption of database files. Thus, credentials can be stored in an encrypted database.

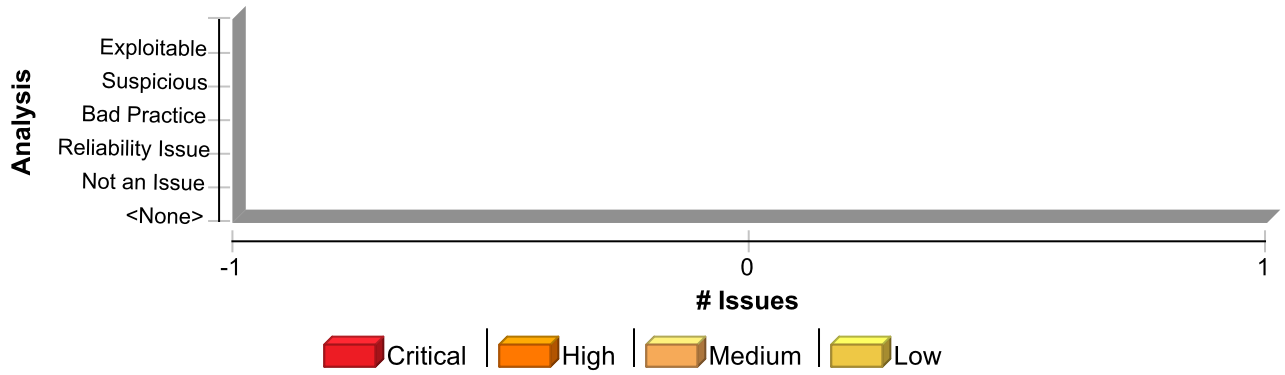
Example 3: The code below demonstrates how to integrate SQLCipher into an Android application after downloading the necessary binaries, and store credentials into the database file.

```
import net.sqlcipher.database.SQLiteDatabase;
...
  SQLiteDatabase.loadLibs(this);
  File dbFile = getDatabasePath("credentials.db");
  dbFile.mkdirs();
  dbFile.delete();
  SQLiteDatabase db = SQLiteDatabase.openOrCreateDatabase(dbFile,
"credentials", null);
  db.execSQL("create table credentials(u, p)");
  db.execSQL("insert into credentials(u, p) values(?, ?)", new Object[]
{username, password});
...
```

Note that references to `android.database.sqlite.SQLiteDatabase` are substituted with those of `net.sqlcipher.database.SQLiteDatabase`.

To enable encryption on the WebView store, WebKit has to be re-compiled with the `sqlcipher.so` library.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Password Management: Hardcoded Password	20	20	0	40
Total	**20**	**20**	**0**	**40**

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons

HtmlClues.java, line 83 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: equals
Enclosing Method: backdoor()
File: HtmlClues.java:83

```
80 String password = s.getParser().getRawParameter(PASSWORD, "");
81
82 //<START_OMIT_SOURCE>
83 return (username.equals("admin") && password.equals("adminpw"));
84 //<END_OMIT_SOURCE>
85 }
86
```

WsSAXInjection.java, line 69 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: WsSAXInjection.java:69

```
66 public class WsSAXInjection extends LessonAdapter
67 {
68
69 private final static String PASSWORD = "password";
70
71 private String password;
72
```

Challenge2Screen.java, line 108 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 108 (Password Management: Hardcoded Password) — Critical

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: Challenge2Screen.java:108

```
105 /**
106 * Description of the Field
107 */
108 protected final static String PASSWORD = "Password";
109
110 /**
111 * Description of the Field
```

WeakAuthenticationCookie.java, line 133 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: equals
Enclosing Method: checkParams()
File: WeakAuthenticationCookie.java:133

```
130 {
131 String loginID = "";
132
133 if (username.equals("webgoat") && password.equals("webgoat"))
134 {
135 loginID = encode("webgoat12345");
136 }
```

DOS_Login.java, line 64 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 64 (Password Management: Hardcoded Password) — Critical

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: DOS_Login.java:64

```
61 /**
62 * Description of the Field
63 */
64 protected final static String PASSWORD = "Password";
65
66 /**
67 * Description of the Field
```

LogSpoofing.java, line 59 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: LogSpoofing.java:59

```
56
57 private static final String USERNAME = "username";
58
59 private static final String PASSWORD = "password";
60
61 private final static IMG MAC_LOGO = new IMG("images/logos/macadamian.gif").setAlt(
62 "Macadamian Technologies").setBorder(0).setHspace(0).setVspace(0);
```

WeakAuthenticationCookie.java, line 137 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: equals
Enclosing Method: checkParams()
File: WeakAuthenticationCookie.java:137

```
134 {
```

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 137 (Password Management: Hardcoded Password) — Critical

```
135 loginID = encode("webgoat12345");
136 }
137 else if (username.equals("aspect") && password.equals("aspect"))
138 {
139 loginID = encode("aspect12345");
140 }
```

XPATHInjection.java, line 79 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: XPATHInjection.java:79

```
76
77 private final static String USERNAME = "Username";
78
79 private final static String PASSWORD = "Password";
80
81 private final static IMG MAC_LOGO = new IMG("images/logos/macadamian.gif").setAlt(
82 "Macadamian Technologies").setBorder(0).setHspace(0).setVspace(0);
```

WeakAuthenticationCookie.java, line 70 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: WeakAuthenticationCookie.java:70

```
67 /**
68 * Description of the Field
69 */
70 protected final static String PASSWORD = "Password";
71
```

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 70 (Password Management: Hardcoded Password) — Critical

```
72 /**
73 * Description of the Field
```

FailOpenAuthentication.java, line 76 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: equals
Enclosing Method: createContent()
File: FailOpenAuthentication.java:76

```
73 password = s.getParser().getRawParameter(PASSWORD);
74
75 // if credentials are bad, send the login page
76 if (!"webgoat".equals(username) || !password.equals("webgoat"))
77 {
78 s.setMessage("Invalid username and password entered.");
79
```

WeakSessionID.java, line 68 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: WeakSessionID.java:68

```
65 /**
66 * Description of the Field
67 */
68 protected final static String PASSWORD = "Password";
69
70 /**
71 * Description of the Field
```

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons

HtmlClues.java, line 63 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: HtmlClues.java:63

```
60 /**
61 * Description of the Field
62 */
63 protected final static String PASSWORD = "Password";
64
65 /**
66 * Description of the Field
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 95 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: CrossSiteScripting.java:95

```
92
93 public final static String FIRST_NAME = "firstName";
94
95 public final static String PASSWORD = "password";
96
97 public final static String EMPLOYEE_ID = "employee_id";
98
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 88 (Password Management: Hardcoded Password) — Critical

Issue Details

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 88 (Password Management: Hardcoded Password) — Critical

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: RoleBasedAccessControl.java:88

```
85
86  public final static String FIRST_NAME = "firstName";
87
88  public final static String PASSWORD = "password";
89
90  public final static String EMPLOYEE_ID = "employee_id";
91
```

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 94 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: SQLInjection.java:94

```
91
92  public final static String FIRST_NAME = "firstName";
93
94  public final static String PASSWORD = "password";
95
96  public final static String EMPLOYEE_ID = "employee_id";
97
```

Package: org.owasp.webgoat.session

ECSFactory.java, line 70 (Password Management: Hardcoded Password) — Critical

Issue Details

Kingdom: Security Features

Password Management: Hardcoded Password — Critical

Package: org.owasp.webgoat.session

ECSFactory.java, line 70 (Password Management: Hardcoded Password) — Critical

Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD
Enclosing Method: ()
File: ECSFactory.java:70

```
67  * Description of the Field
68  */
69
70  public final static String PASSWORD = "Password";
71
72
73  /**
```

Password Management: Hardcoded Password — High

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 147 (Password Management: Hardcoded Password) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: setPassword()
Enclosing Method: accessWGService()
File: WSDLScanning.java:147

```
144  call.addParameter(parameterName, serviceName, ParameterMode.INOUT);
145  call.setReturnType(XMLType.XSD_STRING);
146  call.setUsername("guest");
147  call.setPassword("guest");
148  call.setTargetEndpointAddress("http://localhost/WebGoat/services/"
149  + serv);
150  Object result = call.invoke(new Object[] { parameterValue });
```

WsSAXInjection.java, line 208 (Password Management: Hardcoded Password) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Password Management: Hardcoded Password — High

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 208 (Password Management: Hardcoded Password) — High

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FieldAccess: PASSWORD_TAG
Enclosing Method: ()
File: WsSAXInjection.java:208

```
205 private static class PasswordChanger extends DefaultHandler
206 {
207
208 private static String PASSWORD_TAG = "password";
209
210 private static String ID_TAG = "id";
211
```

Package: org.owasp.webgoat.session

CreateDB.java, line 70 (Password Management: Hardcoded Password) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getConnection()
Enclosing Method: main()
File: CreateDB.java:70

```
67 {
68
69 connection = DriverManager
70 .getConnection(
71 "jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=c:/webgoat.mdb;PWD=webgoat",
72 "webgoat", "webgoat");
73 db.makeDB(connection);
```

DatabaseUtilities.java, line 98 (Password Management: Hardcoded Password) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Password Management: Hardcoded Password — High

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 98 (Password Management: Hardcoded Password) — High

Sink Details

Sink: getConnection()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:98

```
95  System.out.println("DBName: " + dbName);
96  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
97  return DriverManager
98  .getConnection("jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ="
99  + dbName + ";PWD=webgoat");
100 }
101 else
```

Password Management: Password in Comment (28 issues)

Abstract

Storing passwords or password details in plain text anywhere in the system or system code may compromise system security in a way that cannot be easily remedied.

Explanation

It is never a good idea to hardcode a password. Storing password details within comments is equivalent to hardcoding passwords. Not only does it allow all of the project's developers to view the password, it also makes fixing the problem extremely difficult. Once the code is in production, the password is now leaked to the outside world and cannot be protected or changed without patching the software. If the account protected by the password is compromised, the owners of the system will be forced to choose between security and availability.

Example: The following comment specifies the default password to connect to a database:

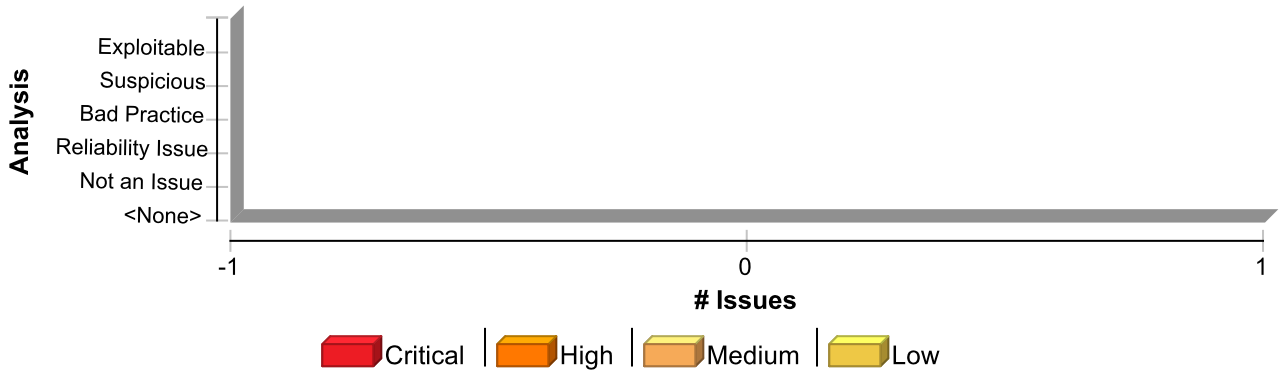
```
...
// Default username for database connection is "scott"
// Default password for database connection is "tiger"
...
```

This code will run successfully, but anyone who has access to it will have access to the password. Once the program has shipped, there is likely no way to change the database user "scott" with a password of "tiger" unless the program is patched. An employee with access to this information could use it to break into the system.

Recommendation

Passwords should never be hardcoded and should generally be obfuscated and managed in an external source. Storing passwords in plain text anywhere on the system allows anyone with sufficient permissions to read and potentially misuse the password.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Password Management: Password in Comment	28	28	0	56
Total	**28**	**28**	**0**	**56**

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons

BasicAuthentication.java, line 258 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: BasicAuthentication.java:258

```
255 }
256
257
258 /**
259 * Gets the category attribute of the ForgotPassword object
260 *
261 * @return The category value
```

BufferOverflow.java, line 59 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: BufferOverflow.java:59

```
56 }
57
58
59 /**
60 * Gets the category attribute of the ForgotPassword object
61 *
62 * @return The category value
```

ForcedBrowsing.java, line 103 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons

ForcedBrowsing.java, line 103 (Password Management: Password in Comment) — Low

Sink Details

Sink: Comment
File: ForcedBrowsing.java:103

```
100 }
101
102
103 /**
104 * Gets the category attribute of the ForgotPassword object
105 *
106 * @return The category value
```

ForgotPassword.java, line 285 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Comment
File: ForgotPassword.java:285

```
282 }
283
284
285 /**
286 * Gets the category attribute of the ForgotPassword object
287 *
288 * @return The category value
```

RemoteAdminFlaw.java, line 69 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Comment
File: RemoteAdminFlaw.java:69

```
66 }
67
```

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons

RemoteAdminFlaw.java, line 69 (Password Management: Password in Comment) — Low

```
68
69 /**
70 * Gets the category attribute of the ForgotPassword object
71 *
72 * @return The category value
```

Challenge2Screen.java, line 139 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: Challenge2Screen.java:139

```
136 }
137
138
139 /**
140 * Determine the username and password
141 *
142 * @param s Description of the Parameter
```

FailOpenAuthentication.java, line 102 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: FailOpenAuthentication.java:102

```
99 if (password.length() == 0)
100 {
101 // We make sure the username was submitted to avoid telling the user an invalid
102 // username/password was entered when they first enter the lesson via the side menu.
103 // This also suppresses the error if they just hit the login and both fields are empty.
104 if (username.length() != 0)
105 {
```

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons

FailOpenAuthentication.java, line 98 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: FailOpenAuthentication.java:98

```
95  }
96  }
97
98  // Don't let the fail open pass with a blank password.
99  if (password.length() == 0)
100 {
101 // We make sure the username was submitted to avoid telling the user an invalid
```

Package: .org.owasp.webgoat.lessons.CrossSiteScripting

EditProfile.java, line 100 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: EditProfile.java:100

```
97  ResultSet answer_results = answer_statement.executeQuery();
98  if (answer_results.next())
99  {
100 // Note: Do NOT get the password field.
101 profile = new Employee(answer_results.getInt("userid"),
102 answer_results.getString("first_name"),
103 answer_results.getString("last_name"),
```

EditProfile.java, line 158 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.CrossSiteScripting

EditProfile.java, line 158 (Password Management: Password in Comment) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: EditProfile.java:158

```
155 ResultSet answer_results = answer_statement.executeQuery();
156 if (answer_results.next())
157 {
158 // Note: Do NOT get the password field.
159 profile = new Employee(answer_results.getInt("userid"),
160 answer_results.getString("first_name"),
161 answer_results.getString("last_name"),
```

FindProfile.java, line 184 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: FindProfile.java:184

```
181 if (answer_results.next())
182 {
183 int id = answer_results.getInt("userid");
184 // Note: Do NOT get the password field.
185 profile = new Employee(id, answer_results
186 .getString("first_name"), answer_results
187 .getString("last_name"), answer_results
```

ViewProfile.java, line 115 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.java:115

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 115 (Password Management: Password in Comment) — Low

```
112 ResultSet answer_results = answer_statement.executeQuery(query);
113 if (answer_results.next())
114 {
115 // Note: Do NOT get the password field.
116 profile = new Employee(answer_results.getInt("userid"),
117 answer_results.getString("first_name"),
118 answer_results.getString("last_name"),
```

ViewProfile.java, line 175 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.java:175

```
172 ResultSet answer_results = answer_statement.executeQuery(query);
173 if (answer_results.next())
174 {
175 // Note: Do NOT get the password field.
176 profile = new Employee(answer_results.getInt("userid"),
177 answer_results.getString("first_name"),
178 answer_results.getString("last_name"),
```

UpdateProfile.java, line 221 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: UpdateProfile.java:221

```
218 {
219 try
220 {
221 // Note: The password field is ONLY set by ChangePassword
222 String query = "UPDATE employee SET first_name = '"
223 + employee.getFirstName() + "', last_name = '"
```

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 221 (Password Management: Password in Comment) — Low

```
224  + employee.getLastName() + "', ssn = '" + employee.getSsn()
```

UpdateProfile.java, line 270 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: UpdateProfile.java:270

```
267  {
268  try
269  {
270  // Note: The password field is ONLY set by ChangePassword
271  String query = "UPDATE employee SET first_name = '"
272  + employee.getFirstName() + "', last_name = '"
273  + employee.getLastName() + "', ssn = '" + employee.getSsn()
```

Package: .org.owasp.webgoat.lessons.RoleBasedAccessControl

EditProfile.java, line 100 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: EditProfile.java:100

```
97  ResultSet answer_results = answer_statement.executeQuery();
98  if (answer_results.next())
99  {
100  // Note: Do NOT get the password field.
101  profile = new Employee(answer_results.getInt("userid"),
102  answer_results.getString("first_name"),
103  answer_results.getString("last_name"),
```

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.RoleBasedAccessControl

EditProfile.java, line 161 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: EditProfile.java:161

```
158 ResultSet answer_results = answer_statement.executeQuery();
159 if (answer_results.next())
160 {
161 // Note: Do NOT get the password field.
162 profile = new Employee(answer_results.getInt("userid"),
163 answer_results.getString("first_name"),
164 answer_results.getString("last_name"),
```

FindProfile.java, line 152 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: FindProfile.java:152

```
149 if (answer_results.next())
150 {
151 int id = answer_results.getInt("userid");
152 // Note: Do NOT get the password field.
153 profile = new Employee(id, answer_results
154 .getString("first_name"), answer_results
155 .getString("last_name"), answer_results
```

ViewProfile.java, line 135 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.RoleBasedAccessControl

ViewProfile.java, line 135 (Password Management: Password in Comment) — Low

Sink Details

Sink: Comment
File: ViewProfile.java:135

```
132 ResultSet answer_results = answer_statement.executeQuery(query);
133 if (answer_results.next())
134 {
135 // Note: Do NOT get the password field.
136 profile = new Employee(answer_results.getInt("userid"),
137 answer_results.getString("first_name"),
138 answer_results.getString("last_name"),
```

ViewProfile.java, line 195 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.java:195

```
192 ResultSet answer_results = answer_statement.executeQuery(query);
193 if (answer_results.next())
194 {
195 // Note: Do NOT get the password field.
196 profile = new Employee(answer_results.getInt("userid"),
197 answer_results.getString("first_name"),
198 answer_results.getString("last_name"),
```

DeleteProfile.java, line 107 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: DeleteProfile.java:107

```
104 {
105 try
```

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 107 (Password Management: Password in Comment) — Low

```
106 {
107 // Note: The password field is ONLY set by ChangePassword
108 String query = "DELETE FROM employee WHERE userid = " + employeeId;
109 //System.out.println("Query: " + query);
110 try
```

DeleteProfile.java, line 136 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: DeleteProfile.java:136

```
133 {
134 try
135 {
136 // Note: The password field is ONLY set by ChangePassword
137 String query = "DELETE FROM employee WHERE userid = " + employeeId;
138 //System.out.println("Query: " + query);
139 try
```

UpdateProfile.java, line 149 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: UpdateProfile.java:149

```
146 {
147 try
148 {
149 // Note: The password field is ONLY set by ChangePassword
150 String query = "UPDATE employee SET first_name = '"
151 + employee.getFirstName() + "', last_name = '"
152 + employee.getLastName() + "', ssn = '" + employee.getSsn()
```

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 198 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Comment
File: UpdateProfile.java:198

```
195 {
196 try
197 {
198 // Note: The password field is ONLY set by ChangePassword
199 String query = "UPDATE employee SET first_name = '"
200 + employee.getFirstName() + "', last_name = '"
201 + employee.getLastName() + "', ssn = '" + employee.getSsn()
```

Package: .org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 121 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.java:121

```
118 ResultSet answer_results = answer_statement.executeQuery(query);
119 if (answer_results.next())
120 {
121 // Note: Do NOT get the password field.
122 profile = new Employee(answer_results.getInt("userid"),
123 answer_results.getString("first_name"),
124 answer_results.getString("last_name"),
```

ViewProfile.java, line 181 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Password Management: Password in Comment — Low

Package: .org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 181 (Password Management: Password in Comment) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.java:181

```
178 ResultSet answer_results = answer_statement.executeQuery(query);
179 if (answer_results.next())
180 {
181 // Note: Do NOT get the password field.
182 profile = new Employee(answer_results.getInt("userid"),
183 answer_results.getString("first_name"),
184 answer_results.getString("last_name"),
```

Package: .org.owasp.webgoat.session

Course.java, line 76 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: Course.java:76

```
73 }
74 }
75
76 /**
77 * Take an absolute file and return the filename.
78 *
79 * Ex. /etc/password becomes password
```

ECSFactory.java, line 169 (Password Management: Password in Comment) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Password Management: Password in Comment	Low
Package: .org.owasp.webgoat.session	
ECSFactory.java, line 169 (Password Management: Password in Comment)	**Low**

Sink: Comment
File: ECSFactory.java:169

```
166 Input field = new Input().setName(name).setValue(value).setSize(size)
167 .setMaxlength(size);
168
169 // double check in case someone means to make a * starred out password field
170
171 if (name.equals(PASSWORD))
172 {
```

Password Management: Password in Configuration File (2 issues)

Abstract

Storing a plain text password in a configuration file may result in a system compromise.

Explanation

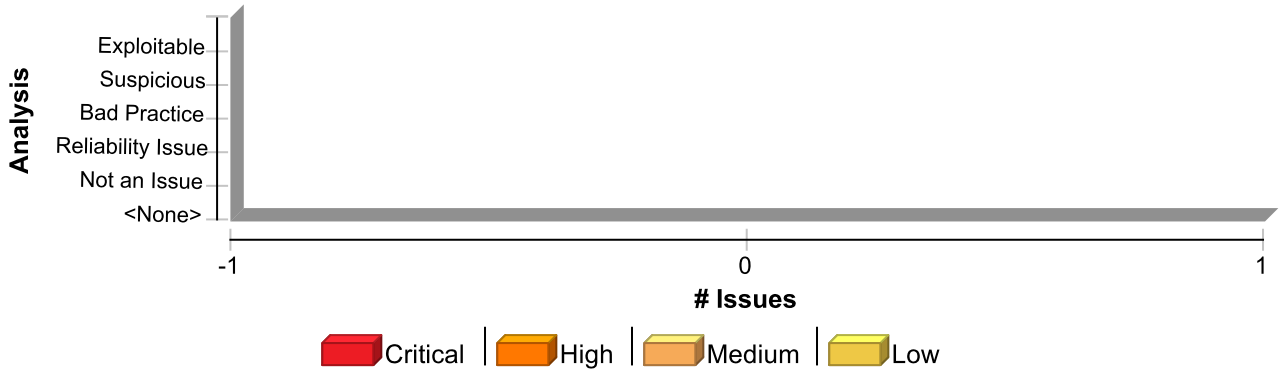
Storing a plain text password in a configuration file allows anyone who can read the file access to the password-protected resource. Developers sometimes believe that they cannot defend the application from someone who has access to the configuration, but this attitude makes an attacker's job easier. Good password management guidelines require that a password never be stored in plain text.

Recommendation

A password should never be stored in plain text. Instead, the password should be entered by an administrator when the system starts. If that approach is impractical, a less secure but often adequate solution is to obfuscate the password and scatter the de-obfuscation material around the system so that an attacker has to obtain and correctly combine multiple system resources to decipher the password.

Some third-party products claim the ability to manage passwords in a more secure way. For example, WebSphere Application Server 4.x uses a simple XOR encryption algorithm for obfuscating values, but be skeptical about such facilities. WebSphere and other application servers offer outdated and relatively weak encryption mechanisms that are insufficient for security-sensitive environments. For a secure solution the only viable option is a proprietary one.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Password Management: Password in Configuration File	2	2	0	4
Total	**2**	**2**	**0**	**4**

Password Management: Password in Configuration File	**High**
Package: <none>	
build.xml, line 85 (Password Management: Password in Configuration File)	**High**

Issue Details

Kingdom: Environment

Password Management: Password in Configuration File — High

Package: <none>

build.xml, line 85 (Password Management: Password in Configuration File) — High

Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: build.xml:85

```
82 <property name="doc.home" value="${basedir}/doc"/>
83 <property name="manager.url" value="http://localhost/manager"/>
84 <property name="manager.username" value="admin"/> <!-- UPDATE THIS! -->
85 <property name="manager.password" value="admin"/> <!-- UPDATE THIS! -->
86 <property name="src.home" value="${basedir}/JavaSource"/>
87 <property name="web.home" value="${basedir}/WebContent"/>
88 <property name="zip_distributions.home" value="${basedir}/zip_distributions"/>
```

Package: WebContent.WEB-INF

server-config.wsdd, line 6 (Password Management: Password in Configuration File) — High

Issue Details

Kingdom: Environment
Scan Engine: SCA (Configuration)

Audit Details

AA_Prediction Not Predicted

Sink Details

File: server-config.wsdd:6

```
3 <globalConfiguration>
4 <parameter name="sendMultiRefs" value="true"/>
5 <parameter name="disablePrettyXML" value="true"/>
6 <parameter name="adminPassword" value="admin"/>
7 <!--
8 <parameter name="attachments.Directory" value="C:\webgoat\tomcat\webapps\WebGoat\WEB-INF\attachments"/>
9 -->
```

Path Manipulation (5 issues)

Abstract

Allowing user input to control paths used in file system operations could enable an attacker to access or modify otherwise protected system resources.

Explanation

Path manipulation errors occur when the following two conditions are met:

1. An attacker is able to specify a path used in an operation on the file system.

2. By specifying the resource, the attacker gains a capability that would not otherwise be permitted.

For example, the program may give the attacker the ability to overwrite the specified file or run with a configuration controlled by the attacker.

Example 1: The following code uses input from an HTTP request to create a file name. The programmer has not considered the possibility that an attacker could provide a file name such as "`../../tomcat/conf/server.xml`", which causes the application to delete one of its own configuration files.

```
String rName = request.getParameter("reportName");
File rFile = new File("/usr/local/apfr/reports/" + rName);
...
rFile.delete();
```

Example 2: The following code uses input from a configuration file to determine which file to open and echo back to the user. If the program runs with adequate privileges and malicious users can change the configuration file, they can use the program to read any file on the system that ends with the extension `.txt`.

```
fis = new FileInputStream(cfg.getProperty("sub")+".txt");
amt = fis.read(arr);
out.println(arr);
```

Some think that in the mobile world, classic vulnerabilities, such as path manipulation, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 3: The following code adapts Example 1 to the Android platform.

```
...
        String rName = this.getIntent().getExtras().getString("reportName");
        File rFile = getBaseContext().getFileStreamPath(rName);
...
        rFile.delete();
...
```

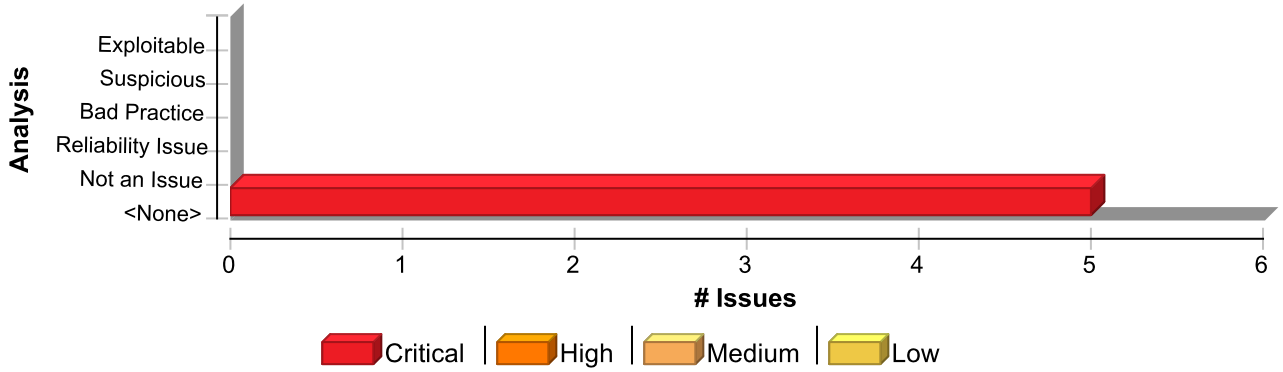
Recommendation

The best way to prevent path manipulation is with a level of indirection: create a list of legitimate resource names that

a user is allowed to specify, and only allow the user to select from the list. With this approach the input provided by the user is never used directly to specify the resource name.

In some situations this approach is impractical because the set of legitimate resource names is too large or too hard to keep track of. Programmers often resort to blacklisting in these situations. Blacklisting selectively rejects or escapes potentially dangerous characters before using the input. However, any such list of unsafe characters is likely to be incomplete and will almost certainly become out of date. A better approach is to create a whitelist of characters that are allowed to appear in the resource name and accept input composed exclusively of characters in the approved set.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Path Manipulation	5	5	0	10
Total	**5**	**5**	**0**	**10**

Path Manipulation — Critical

Package: org.owasp.webgoat.lessons

PathBasedAccessControl.java, line 136 (Path Manipulation) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
```

Path Manipulation — Critical

Package: org.owasp.webgoat.lessons

PathBasedAccessControl.java, line 136 (Path Manipulation) — Critical

```
629  if (values == null)
630  {
```

Sink Details

Sink: java.io.File.File()
Enclosing Method: createContent()
File: PathBasedAccessControl.java:136
Taint Flags: VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
133  // We could force the user to use encoded '/'s == %2f to make the lesson more difficult.
134  // We url Encode our dir name to avoid problems with special characters in our own path.
135  //File f = new File( new URI("file:///" + Encoding.urlEncode(dir).replaceAll("\\\\","/") + "/" + file.replaceAll("\\\\","/")) );
136  File f = new File((dir + "\\" + file).replaceAll("\\\\", "/"));
137
138  if (s.isDebug())
139  {
```

CommandInjection.java, line 172 (Path Manipulation) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.io.File.File()
Enclosing Method: createContent()
File: CommandInjection.java:172
Taint Flags: VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

Path Manipulation — Critical

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 172 (Path Manipulation) — Critical

```
169  results = exec(s, "cmd.exe /c dir /b \""
170  + safeDir.getPath() + "\"");
171  fileData = exec(s, "cmd.exe /c type \""
172  + new File(safeDir, helpFile).getPath() + "\"");
173
174  }
175  else
```

CommandInjection.java, line 183 (Path Manipulation) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.io.File.File()
Enclosing Method: createContent()
File: CommandInjection.java:183
Taint Flags: VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR, WEB, XSS

```
180  String[] cmd2 = {
181  "/bin/sh",
182  "-c",
183  "cat \"" + new File(safeDir, helpFile).getPath()
184  + "\"" };
185  fileData = exec(s, cmd2);
186  }
```

Path Manipulation — Critical

Package: org.owasp.webgoat.session

LessonTracker.java, line 238 (Path Manipulation) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterNames()
From: org.owasp.webgoat.session.ParameterParser.getParameterNames
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:576

```
573  return (null);
574  }
575
576  return request.getParameterNames();
577  }
578
579
```

Sink Details

Sink: java.io.FileInputStream.FileInputStream()
Enclosing Method: load()
File: LessonTracker.java:238
Taint Flags: PRIMARY_KEY, WEB, XSS

```
235  {
236  Properties tempProps = new Properties();
237  //System.out.println("Loading lesson state from: " + fileName);
238  in = new FileInputStream(fileName);
239  tempProps.load(in);
240  // allow the screen to use any custom properties it may have set
241  LessonTracker tempLessonTracker = screen
```

LessonTracker.java, line 238 (Path Manipulation) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Path Manipulation — Critical

Package: org.owasp.webgoat.session

LessonTracker.java, line 238 (Path Manipulation) — Critical

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.io.FileInputStream.FileInputStream()
Enclosing Method: load()
File: LessonTracker.java:238
Taint Flags: WEB, XSS

```
235  {
236  Properties tempProps = new Properties();
237  //System.out.println("Loading lesson state from: " + fileName);
238  in = new FileInputStream(fileName);
239  tempProps.load(in);
240  // allow the screen to use any custom properties it may have set
241  LessonTracker tempLessonTracker = screen
```

Poor Error Handling: Empty Catch Block (32 issues)

Abstract

Ignoring an exception can cause the program to overlook unexpected states and conditions.

Explanation

Just about every serious attack on a software system begins with the violation of a programmer's assumptions. After the attack, the programmer's assumptions seem flimsy and poorly founded, but before an attack many programmers would defend their assumptions well past the end of their lunch break.

Two dubious assumptions that are easy to spot in code are "this method call can never fail" and "it doesn't matter if this call fails". When a programmer ignores an exception, they implicitly state that they are operating under one of these assumptions.

Example 1: The following code excerpt ignores a rarely-thrown exception from `doExchange()`.

```
try {
  doExchange();
}
catch (RareException e) {
  // this can never happen
}
```

If a `RareException` were to ever be thrown, the program would continue to execute as though nothing unusual had occurred. The program records no evidence indicating the special situation, potentially frustrating any later attempt to explain the program's behavior.

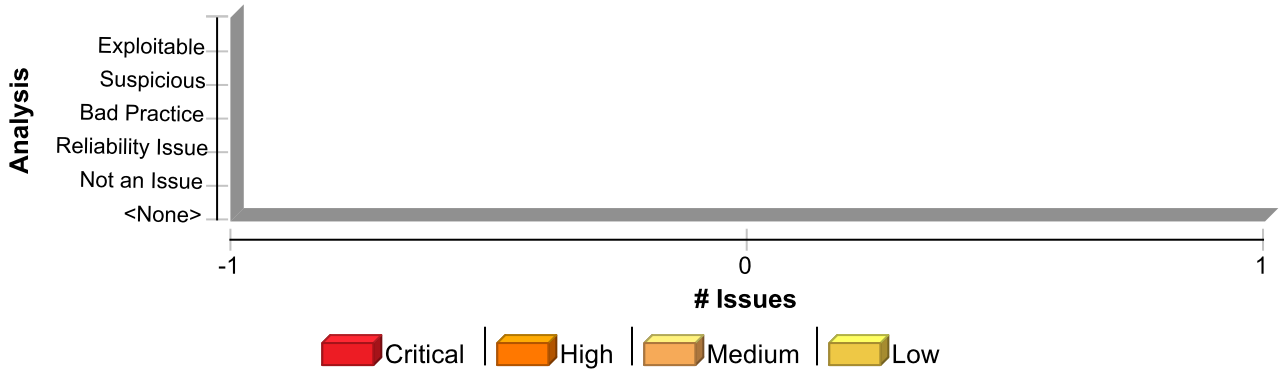
Recommendation

At a minimum, log the fact that the exception was thrown so that it will be possible to come back later and make sense of the resulting program behavior. Better yet, abort the current operation. If the exception is being ignored because the caller cannot properly handle it but the context makes it inconvenient or impossible for the caller to declare that it throws the exception itself, consider throwing a `RuntimeException` or an `Error`, both of which are unchecked exceptions. As of JDK 1.4, `RuntimeException` has a constructor that makes it easy to wrap another exception.

Example 2: The code in Example 1 could be rewritten in the following way:

```
try {
  doExchange();
}
catch (RareException e) {
  throw new RuntimeException("This can never happen", e);
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Error Handling: Empty Catch Block	32	32	0	64
Total	**32**	**32**	**0**	**64**

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 228 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: makeLogin()
File: WeakSessionID.java:228

```
225 {
226 username = s.getParser().getStringParameter(USERNAME);
227 }
228 catch (ParameterNotFoundException pnfe)
229 {}
230 try
231 {
```

WeakSessionID.java, line 234 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 234 (Poor Error Handling: Empty Catch Block) — Low

Sink: CatchBlock
Enclosing Method: makeLogin()
File: WeakSessionID.java:234

```
231 {
232 password = s.getParser().getStringParameter(PASSWORD);
233 }
234 catch (ParameterNotFoundException pnfe)
235 {}
236
237 if (username != null || password != null)
```

WSDLScanning.java, line 290 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: WSDLScanning.java:290

```
287 return results.getString(field);
288 }
289 }
290 catch (SQLException sqle)
291 {}
292 }
293 catch (Exception e)
```

WSDLScanning.java, line 293 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: WSDLScanning.java:293

```
290 catch (SQLException sqle)
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 293 (Poor Error Handling: Empty Catch Block) — Low

```
291 {}
292 }
293 catch (Exception e)
294 {}
295 return null;
296 }
```

WsSqlInjection.java, line 246 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: WsSqlInjection.java:246

```
243 catch (SQLException sqle)
244 {}
245 }
246 catch (Exception e)
247 {}
248 return null;
249 }
```

WsSqlInjection.java, line 243 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: WsSqlInjection.java:243

```
240 ResultSet results = statement.executeQuery(query);
241 return results;
242 }
243 catch (SQLException sqle)
244 {}
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 243 (Poor Error Handling: Empty Catch Block) — Low

```
245 }
246 catch (Exception e)
```

SoapRequest.java, line 265 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: viewWsdl()
File: SoapRequest.java:265

```
262 //ec.addElement( new P().addElement( nfe.getMessage() ) );
263 s.setMessage("Sorry, that answer is invalid. Try again.");
264 }
265 catch (ParameterNotFoundException pnfe)
266 {
267 //DEVNOTE: Eat the exception.
268 // ec.addElement( new P().addElement( pnfe.getMessage() ) );
```

DefaultLessonAction.java, line 145 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getIntSessionAttribute()
File: DefaultLessonAction.java:145

```
142 {
143 value = Integer.parseInt(ss);
144 }
145 catch (NumberFormatException nfe)
146 {
147 }
148 }
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 322 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: determineType()
File: SoapRequest.java:322

```
319 s.setMessage("Sorry, that is an incorrect type. Try Again.");
320 }
321 }
322 catch (ParameterNotFoundException pnfe)
323 {
324 //DEVNOTE: Eat the exception.
325 // ec.addElement( new P().addElement( pnfe.getMessage() ) );
```

LessonAdapter.java, line 307 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getInstructions()
File: LessonAdapter.java:307

```
304 }
305 }
306 }
307 catch (Exception e)
308 {}
309
310 return buff.toString();
```

DefaultLessonAction.java, line 236 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 236 (Poor Error Handling: Empty Catch Block) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: isAuthenticated()
File: DefaultLessonAction.java:236

```
233 {
234 authenticated = getBooleanSessionAttribute(s, getLessonName() + ".isAuthenticated");
235 }
236 catch (ParameterNotFoundException e)
237 {
238 }
239
```

SoapRequest.java, line 428 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: SoapRequest.java:428

```
425 return results.getString(field);
426 }
427 }
428 catch (SQLException sqle)
429 {}
430 }
431 catch (Exception e)
```

SoapRequest.java, line 431 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 431 (Poor Error Handling: Empty Catch Block) — Low

Sink: CatchBlock
Enclosing Method: getResults()
File: SoapRequest.java:431

```
428 catch (SQLException sqle)
429 {}
430 }
431 catch (Exception e)
432 {}
433 return null;
434 }
```

WsSqlInjection.java, line 273 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getCreditCard()
File: WsSqlInjection.java:273

```
270 }
271 return users;
272 }
273 catch (SQLException sqle)
274 {}
275 }
276 return null;
```

DefaultLessonAction.java, line 178 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getIntRequestAttribute()
File: DefaultLessonAction.java:178

```
175 {
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 178 (Poor Error Handling: Empty Catch Block) — Low

```
176 value = Integer.parseInt(ss);
177 }
178 catch (NumberFormatException nfe)
179 {
180 }
181 }
```

WSDLScanning.java, line 232 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: WSDLScanning.java:232

```
229 t.addElement(results);
230 ec.addElement(new P().addElement(t));
231 }
232 catch (Exception e)
233 {
234
235 }
```

LessonAdapter.java, line 101 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: LessonAdapter.java:101

```
98 }
99 ec.addElement(pre);
100 }
101 catch (Exception e)
102 {}
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 101 (Poor Error Handling: Empty Catch Block) — Low

```
103 }
104 return (ec);
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

FindProfile.java, line 235 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: foundEmployee()
File: FindProfile.java:235

```
232 + CrossSiteScripting.EMPLOYEE_ID);
233 found = true;
234 }
235 catch (ParameterNotFoundException e)
236 {}
237
238 return found;
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 176 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: updateLessonStatus()
File: DeleteProfile.java:176

```
173 setStage(s, 2);
174 }
175 }
176 catch (ParameterNotFoundException e)
177 {}
178 }
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 176 (Poor Error Handling: Empty Catch Block) — Low

```
179
```

FindProfile.java, line 118 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: foundEmployee()
File: FindProfile.java:118

```
115  + RoleBasedAccessControl.EMPLOYEE_ID);
116  found = true;
117  }
118  catch (ParameterNotFoundException e)
119  {}
120
121  return found;
```

ViewProfile.java, line 105 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: updateLessonStatus()
File: ViewProfile.java:105

```
102  setStage(s, 4);
103  }
104  }
105  catch (ParameterNotFoundException e)
106  {}
107  }
108
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 297 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: updateLessonStatus()
File: Login.java:297

```
294 break;
295 }
296 }
297 catch (ParameterNotFoundException pnfe)
298 {}
299 }
300
```

ViewProfile.java, line 253 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: updateLessonStatus()
File: ViewProfile.java:253

```
250 targetEmployee = getEmployeeProfile_BACKUP(s,
251 userId, employeeId);
252 }
253 catch (UnauthorizedException e)
254 {}
255 if (targetEmployee != null
256 && targetEmployee.getId() == SQLInjection.PRIZE_EMPLOYEE_ID)
```

ViewProfile.java, line 268 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 268 (Poor Error Handling: Empty Catch Block) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: updateLessonStatus()
File: ViewProfile.java:268

```
265 break;
266 }
267 }
268 catch (ParameterNotFoundException pnfe)
269 {}
270 }
271
```

Package: org.owasp.webgoat.lessons.admin

ReportCardScreen.java, line 87 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: ReportCardScreen.java:87

```
84 user = s.getUserName();
85 }
86 }
87 catch (Exception e)
88 {}
89
90 if (user == null)
```

Package: org.owasp.webgoat.session

UserTracker.java, line 133 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.session

UserTracker.java, line 133 (Poor Error Handling: Empty Catch Block) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllUsers()
File: UserTracker.java:133

```
130 }
131 usersDB.close();
132 }
133 catch (Exception e)
134 {}
135 return allUsers;
136 }
```

LessonTracker.java, line 247 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: load()
File: LessonTracker.java:247

```
244 return tempLessonTracker;
245 }
246 }
247 catch (FileNotFoundException e)
248 {
249 // Normal if the lesson has not been accessed yet.
250 }
```

LessonTracker.java, line 262 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 262 (Poor Error Handling: Empty Catch Block) — Low

Sink: CatchBlock
Enclosing Method: load()
File: LessonTracker.java:262

```
259 {
260 in.close();
261 }
262 catch (Exception e)
263 {}
264 }
265
```

Course.java, line 377 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: loadLessons()
File: Course.java:377

```
374 }
375 }
376 }
377 catch (Exception e)
378 {
379 //System.out.println("Warning: " + e.getMessage());
380 }
```

LessonTracker.java, line 403 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: store()
File: LessonTracker.java:403

```
400 {
```

Poor Error Handling: Empty Catch Block — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 403 (Poor Error Handling: Empty Catch Block) — Low

```
401 out.close();
402 }
403 catch (Exception e)
404 {}
405 }
406
```

WebSession.java, line 1027 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: update()
File: WebSession.java:1027

```
1024 }
1025 }
1026 }
1027 catch ( Exception e )
1028 {
1029 }
1030
```

UserTracker.java, line 161 (Poor Error Handling: Empty Catch Block) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: deleteUser()
File: UserTracker.java:161

```
158 usersDB.close();
159
160 }
161 catch (Exception e)
162 {}
```

Poor Error Handling: Empty Catch Block	Low
Package: org.owasp.webgoat.session	
UserTracker.java, line 161 (Poor Error Handling: Empty Catch Block)	**Low**

```
163 }
164 }
```

Poor Error Handling: Overly Broad Catch (149 issues)

Abstract

The catch block handles a broad swath of exceptions, potentially trapping dissimilar issues or problems that should not be dealt with at this point in the program.

Explanation

Multiple catch blocks can get ugly and repetitive, but "condensing" catch blocks by catching a high-level class such as `Exception` can obscure exceptions that deserve special treatment or that should not be caught at this point in the program. Catching an overly broad exception essentially defeats the purpose of Java's typed exceptions, and can become particularly dangerous if the program grows and begins to throw new types of exceptions. The new exception types will not receive any attention.

Example: The following code excerpt handles three types of exceptions in an identical fashion.

```
try {
  doExchange();
}
catch (IOException e) {
  logger.error("doExchange failed", e);
}
catch (InvocationTargetException e) {
  logger.error("doExchange failed", e);
}
catch (SQLException e) {
  logger.error("doExchange failed", e);
}
```

At first blush, it may seem preferable to deal with these exceptions in a single catch block, as follows:

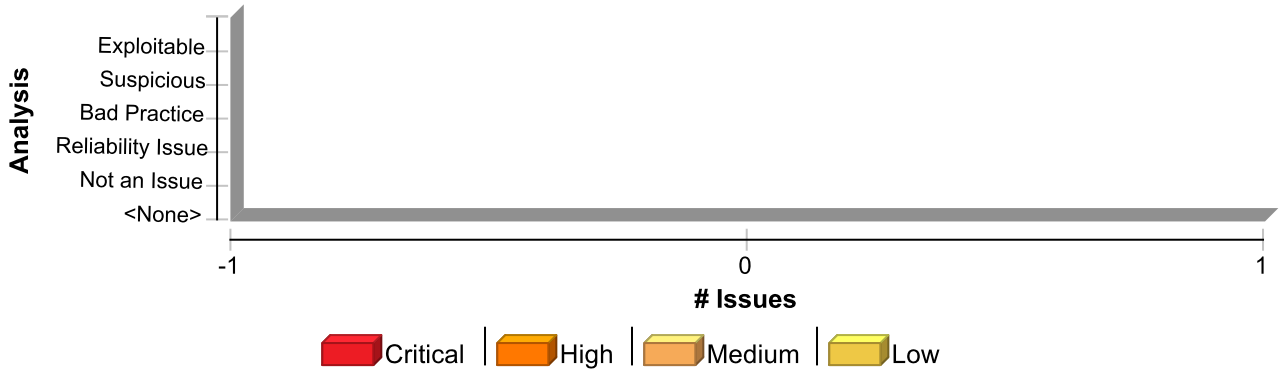
```
try {
  doExchange();
}
catch (Exception e) {
  logger.error("doExchange failed", e);
}
```

However, if `doExchange()` is modified to throw a new type of exception that should be handled in some different kind of way, the broad catch block will prevent the compiler from pointing out the situation. Further, the new catch block will now also handle exceptions derived from `RuntimeException` such as `ClassCastException`, and `NullPointerException`, which is not the programmer's intent.

Recommendation

Do not catch broad exception classes such as `Exception`, `Throwable`, `Error`, or <RuntimeException> except at the very top level of the program or thread.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Error Handling: Overly Broad Catch	149	149	0	298
Total	**149**	**149**	**0**	**298**

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 810 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: isAuthorized()
File: AbstractLesson.java:810

```
807 sqle.printStackTrace();
808 }
809 }
810 catch (Exception e)
811 {
812 s.setMessage("Error authorizing");
813 e.printStackTrace();
```

Challenge2Screen.java, line 346 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 346 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: doStage3()
File: Challenge2Screen.java:346

```
343
344 ec.addElement(t);
345 }
346 catch (Exception e)
347 {
348 ec
349 .addElement(new P()
```

CommandInjection.java, line 208 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: CommandInjection.java:208

```
205
206 }
207 }
208 catch (Exception e)
209 {
210 s.setMessage("Error generating " + this.getClass().getName());
211 e.printStackTrace();
```

JavaScriptValidation.java, line 256 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: JavaScriptValidation.java:256

```
253 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 256 (Poor Error Handling: Overly Broad Catch) — Low

```
254 }
255
256 catch (Exception e)
257 {
258 s.setMessage("Error generating " + this.getClass().getName());
259 e.printStackTrace();
```

AbstractLesson.java, line 1042 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: readFromURL()
File: AbstractLesson.java:1042

```
1039
1040 reader.close();
1041 }
1042 catch (Exception e)
1043 {
1044 System.out.println(e);
1045 e.printStackTrace();
```

FailOpenAuthentication.java, line 120 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: FailOpenAuthentication.java:120

```
117 "Parameters. You did not exploit the fail open."));
118 }
119 }
120 catch (Exception e)
121 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

FailOpenAuthentication.java, line 120 (Poor Error Handling: Overly Broad Catch) — Low

```
122 s.setMessage("Error generating " + this.getClass().getName());
123 }
```

SqlStringInjection.java, line 219 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: parameterizedQuery()
File: SqlStringInjection.java:219

```
216 ec.addElement(new P().addElement(sqle.getMessage()));
217 }
218 }
219 catch (Exception e)
220 {
221 s.setMessage("Error generating " + this.getClass().getName());
222 e.printStackTrace();
```

PathBasedAccessControl.java, line 205 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: PathBasedAccessControl.java:205

```
202 "<br>\\s<br>", "<br>").replaceAll("<\\?",
203 "&lt;").replaceAll("<(r|u|t)", "&lt;$1")));
204 }
205 catch (Exception e)
206 {
207 ec.addElement(new BR());
208 ec
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 159 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: injectableQuery()
File: SqlNumericInjection.java:159

```
156 ec.addElement(new P().addElement(sqle.getMessage()));
157 }
158 }
159 catch (Exception e)
160 {
161 s.setMessage("Error generating " + this.getClass().getName());
162 e.printStackTrace();
```

Encoding.java, line 506 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: decryptString()
File: Encoding.java:506

```
503 return new String( utf8, "UTF-8" );
504 }
505
506 catch ( Exception e )
507 {
508
509 return ( "This is not an encrypted string" );
```

LessonAdapter.java, line 307 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 307 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getInstructions()
File: LessonAdapter.java:307

```
304 }
305 }
306 }
307 catch (Exception e)
308 {}
309
310 return buff.toString();
```

ThreadSafetyProblem.java, line 214 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: ThreadSafetyProblem.java:214

```
211 connection = DatabaseUtilities.makeConnection(s);
212 }
213 }
214 catch (Exception e)
215 {
216 System.out.println("Exception caught: " + e);
217 e.printStackTrace(System.out);
```

WSDLScanning.java, line 161 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 161 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: accessWGService()
File: WSDLScanning.java:161

```
158 {
159 e.printStackTrace();
160 }
161 catch (Exception e)
162 {
163 e.printStackTrace();
164 }
```

DOMInjection.java, line 89 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: DOMInjection.java:89

```
86 makeSuccess(s);
87 }
88 }
89 catch (Exception e)
90 {
91 s.setMessage("Error generating " + this.getClass().getName());
92 e.printStackTrace();
```

SqlStringInjection.java, line 314 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: SqlStringInjection.java:314

```
311 connection = DatabaseUtilities.makeConnection(s);
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 314 (Poor Error Handling: Overly Broad Catch) — Low

```
312 }
313 }
314 catch (Exception e)
315 {
316 System.out.println("Exception caught: " + e);
317 e.printStackTrace(System.out);
```

Encoding.java, line 913 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: unicodeEncode()
File: Encoding.java:913

```
910 ByteBuffer bbuf = encoder.encode( CharBuffer.wrap( str ) );
911 return ( new String( bbuf.array() ) );
912 }
913 catch ( Exception e )
914 {
915 return ( "Encoding problem" );
916 }
```

DefaultLessonAction.java, line 275 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: isAuthorized()
File: DefaultLessonAction.java:275

```
272 sqle.printStackTrace();
273 }
274 }
275 catch ( Exception e )
276 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 275 (Poor Error Handling: Overly Broad Catch) — Low

```
277 s.setMessage( "Error authorizing" );
278 e.printStackTrace();
```

SoapRequest.java, line 270 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: viewWsdl()
File: SoapRequest.java:270

```
267 //DEVNOTE: Eat the exception.
268 // ec.addElement( new P().addElement( pnfe.getMessage() ) );
269 }
270 catch (Exception e)
271 {
272 s.setMessage("Error generating " + this.getClass().getName());
273 e.printStackTrace();
```

PathBasedAccessControl.java, line 214 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: PathBasedAccessControl.java:214

```
211 }
212 }
213 }
214 catch (Exception e)
215 {
216 s.setMessage("Error generating " + this.getClass().getName());
217 e.printStackTrace();
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 135 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: WeakSessionID.java:135

```
132 return makeLogin(s);
133 }
134 }
135 catch (Exception e)
136 {
137 s.setMessage("Error generating " + this.getClass().getName());
138 e.printStackTrace();
```

Challenge2Screen.java, line 726 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: sendMessage()
File: Challenge2Screen.java:726

```
723 OutputStreamWriter osw = new OutputStreamWriter(s.getOutputStream());
724 osw.write(message);
725 }
726 catch (Exception e)
727 {
728 System.out.println("Couldn't write " + message + " to " + s);
729 e.printStackTrace();
```

BackDoors.java, line 134 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

BackDoors.java, line 134 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: concept1()
File: BackDoors.java:134

```
131 }
132 }
133 }
134 catch (Exception ex)
135 {
136 ec.addElement(new PRE(ex.getMessage()));
137 }
```

Challenge2Screen.java, line 388 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: isDefaced()
File: Challenge2Screen.java:388

```
385
386 defaced = (!origText.equals(defacedText));
387 }
388 catch (Exception e)
389 {
390 e.printStackTrace();
391 }
```

CSRF.java, line 100 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

CSRF.java, line 100 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: addMessage()
File: CSRF.java:100

```
97  statement.setString(4, s.getUserName());
98  statement.executeQuery();
99  }
100 catch ( Exception e )
101 {
102 // ignore the empty resultset on the insert. There are a few more SQL Injection errors
103 // that could be trapped here but we will let them try. One error would be something
```

HtmlClues.java, line 117 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: HtmlClues.java:117

```
114 ec.addElement(makeLogin(s));
115 }
116 }
117 catch (Exception e)
118 {
119 s.setMessage("Error generating " + this.getClass().getName());
120 }
```

AbstractLesson.java, line 420 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getFileMethod()
File: AbstractLesson.java:420

```
417
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 420 (Poor Error Handling: Overly Broad Catch) — Low

```
418 reader.close();
419 }
420 catch (Exception e)
421 {
422 System.out.println(e);
423 e.printStackTrace();
```

HttpSplitting.java, line 134 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: doHTTPSplitting()
File: HttpSplitting.java:134

```
131 }
132 }
133 }
134 catch (Exception e)
135 {
136 s.setMessage("Error generating " + this.getClass().getName());
137 e.printStackTrace();
```

Encoding.java, line 1010 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: xorDecode()
File: Encoding.java:1010

```
1007 String decoded = base64Decode( input );
1008 return new String( xor( decoded, userKey ) );
1009 }
1010 catch ( Exception e )
1011 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 1010 (Poor Error Handling: Overly Broad Catch) — Low

```
1012  return "String not XOR encoded.";
1013  }
```

LessonAdapter.java, line 101 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: LessonAdapter.java:101

```
98  }
99  ec.addElement(pre);
100 }
101 catch (Exception e)
102 {}
103 }
104 return (ec);
```

HttpOnly.java, line 182 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createCustomCookieValue()
File: HttpOnly.java:182

```
179 value = encoder.encode(md.digest());
180 original = value;
181
182 } catch (Exception e) {
183 e.printStackTrace();
184 }
185
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 213 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getUserName()
File: DefaultLessonAction.java:213

```
210 sqle.printStackTrace();
211 }
212 }
213 catch ( Exception e )
214 {
215 s.setMessage( "Error getting user name" );
216 e.printStackTrace();
```

StoredXss.java, line 271 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: makeCurrent()
File: StoredXss.java:271

```
268 }
269 }
270 }
271 catch (Exception e)
272 {
273 s.setMessage("Error generating " + this.getClass().getName());
274 e.printStackTrace();
```

AccessControlMatrix.java, line 111 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

AccessControlMatrix.java, line 111 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: AccessControlMatrix.java:111

```
108 + resource);
109 }
110 }
111 catch (Exception e)
112 {
113 s.setMessage("Error generating " + this.getClass().getName());
114 e.printStackTrace();
```

LessonAdapter.java, line 133 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createStagedContent()
File: LessonAdapter.java:133

```
130 throw new Exception("Invalid stage");
131 }
132 }
133 catch (Exception e)
134 {
135 s.setMessage("Error generating " + this.getClass().getName());
136 System.out.println(e);
```

StoredXss.java, line 361 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

StoredXss.java, line 361 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: makeList()
File: StoredXss.java:361

```
358 }
359 }
360 }
361 catch (Exception e)
362 {
363 s.setMessage("Error while getting message list.");
364 }
```

WSDLScanning.java, line 261 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: WSDLScanning.java:261

```
258
259 // accessWGService("WSDLScanning", "getCreditCard", "acct_num", new Integer(101));
260 }
261 catch (Exception e)
262 {
263 s.setMessage("Error generating " + this.getClass().getName());
264 e.printStackTrace();
```

HiddenFieldTampering.java, line 162 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: HiddenFieldTampering.java:162

```
159 .addElement("This amount will be charged to your credit card immediately."));
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

HiddenFieldTampering.java, line 162 (Poor Error Handling: Overly Broad Catch) — Low

```
160 }
161 }
162 catch (Exception e)
163 {
164 s.setMessage("Error generating " + this.getClass().getName());
165 e.printStackTrace();
```

WSDLScanning.java, line 232 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: WSDLScanning.java:232

```
229 t.addElement(results);
230 ec.addElement(new P().addElement(t));
231 }
232 catch (Exception e)
233 {
234
235 }
```

CSRF.java, line 272 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: makeCurrent()
File: CSRF.java:272

```
269 }
270
271 }
272 catch ( Exception e )
273 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

CSRF.java, line 272 (Poor Error Handling: Overly Broad Catch) — Low

```
274 s.setMessage( "Error generating " + this.getClass().getName() );
275 e.printStackTrace();
```

Encoding.java, line 934 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: urlDecode()
File: Encoding.java:934

```
931 {
932 return ( URLDecoder.decode( str, "UTF-8" ) );
933 }
934 catch ( Exception e )
935 {
936 return ( "Decoding error" );
937 }
```

Challenge2Screen.java, line 439 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: resetWebPage()
File: Challenge2Screen.java:439

```
436 fw.close();
437 // System.out.println("webgoat_guest replaced: " + getFileText( new BufferedReader( new FileReader( defacedpath ) ), false ) );
438 }
439 catch (Exception e)
440 {
441 e.printStackTrace();
442 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

BlindSqlInjection.java, line 142 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: BlindSqlInjection.java:142

```
139 .addElement("An error occurred, please try again."));
140 }
141 }
142 catch (Exception e)
143 {
144 s.setMessage("Error generating " + this.getClass().getName());
145 e.printStackTrace();
```

ReflectedXSS.java, line 219 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: ReflectedXSS.java:219

```
216 ec.addElement(new BR());
217 ec.addElement(new HR().setWidth("90%"));
218 }
219 catch (Exception e)
220 {
221 s.setMessage("Error generating " + this.getClass().getName());
222 e.printStackTrace();
```

Challenge2Screen.java, line 265 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 265 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: doStage2()
File: Challenge2Screen.java:265

```
262 ec.addElement(input);
263 }
264 }
265 catch (Exception e)
266 {
267 s.setMessage("An error occurred in the woods");
268 }
```

ThreadSafetyProblem.java, line 125 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: ThreadSafetyProblem.java:125

```
122 }
123
124 }
125 catch (Exception e)
126 {
127 s.setMessage("Error generating " + this.getClass().getName());
128 e.printStackTrace();
```

Encoding.java, line 955 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 955 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: urlEncode()
File: Encoding.java:955

```
952 {
953 return ( URLEncoder.encode( str, "UTF-8" ) );
954 }
955 catch ( Exception e )
956 {
957 return ( "Encoding error" );
958 }
```

AbstractLesson.java, line 571 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getLessonPlan()
File: AbstractLesson.java:571

```
568 .getWebResource(getLessonPlanFileName()))), false);
569
570 }
571 catch (Exception e)
572 {
573 // s.setMessage( "Could not find lesson plan for " +
574 // getLessonName());
```

SqlNumericInjection.java, line 239 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: parameterizedQuery()
File: SqlNumericInjection.java:239

```
236 + npe.getMessage()));
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 239 (Poor Error Handling: Overly Broad Catch) — Low

```
237 }
238 }
239 catch (Exception e)
240 {
241 s.setMessage("Error generating " + this.getClass().getName());
242 e.printStackTrace();
```

Encoding.java, line 454 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: Encoding.java:454

```
451
452 }
453
454 catch ( Exception e )
455 {
456
457 s.setMessage( "Error generating " + this.getClass().getName() );
```

BasicAuthentication.java, line 248 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: doStage2()
File: BasicAuthentication.java:248

```
245 }
246
247 }
248 catch (Exception e)
249 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

BasicAuthentication.java, line 248 (Poor Error Handling: Overly Broad Catch) — Low

```
250 s.setMessage("Error generating " + this.getClass().getName());
251 e.printStackTrace();
```

WeakAuthenticationCookie.java, line 194 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: WeakAuthenticationCookie.java:194

```
191 return (makeUser(s, user, "PARAMETERS"));
192 }
193 }
194 catch (Exception e)
195 {
196 s.setMessage("Error generating " + this.getClass().getName());
197 e.printStackTrace();
```

HttpSplitting.java, line 235 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: doCachePoisining()
File: HttpSplitting.java:235

```
232 }
233 }
234 }
235 catch (Exception ex)
236 {
237 ec.addElement(new P().addElement(ex.getMessage()));
238 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

HttpOnly.java, line 135 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: HttpOnly.java:135

```
132 {
133 ec.addElement(makeContent(s));
134 }
135 catch ( Exception e )
136 {
137 s.setMessage( "Error generating " + this.getClass().getName() );
138 e.printStackTrace();
```

SqlStringInjection.java, line 149 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: injectableQuery()
File: SqlStringInjection.java:149

```
146 ec.addElement(new P().addElement(sqle.getMessage()));
147 }
148 }
149 catch (Exception e)
150 {
151 s.setMessage("Error generating " + this.getClass().getName());
152 e.printStackTrace();
```

AbstractLesson.java, line 476 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 476 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getFileText()
File: AbstractLesson.java:476

```
473
474 reader.close();
475 }
476 catch (Exception e)
477 {
478 System.out.println(e);
479 e.printStackTrace();
```

HttpBasics.java, line 75 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: HttpBasics.java:75

```
72 Element b = ECSFactory.makeButton("Go!");
73 ec.addElement(b);
74 }
75 catch (Exception e)
76 {
77 s.setMessage("Error generating " + this.getClass().getName());
78 e.printStackTrace();
```

WSDLScanning.java, line 293 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 293 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: getResults()
File: WSDLScanning.java:293

```
290 catch (SQLException sqle)
291 {}
292 }
293 catch (Exception e)
294 {}
295 return null;
296 }
```

SilentTransactions.java, line 115 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: SilentTransactions.java:115

```
112 }
113 }
114 }
115 catch (Exception ex)
116 {
117 ex.printStackTrace();
118 }
```

WsSqlInjection.java, line 246 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: WsSqlInjection.java:246

```
243 catch (SQLException sqle)
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 246 (Poor Error Handling: Overly Broad Catch) — Low

```
244 {}
245 }
246 catch (Exception e)
247 {}
248 return null;
249 }
```

XMLInjection.java, line 126 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: XMLInjection.java:126

```
123 }
124 }
125 }
126 catch (Exception ex)
127 {
128 ex.printStackTrace();
129 }
```

Encoding.java, line 550 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: encryptString()
File: Encoding.java:550

```
547 return encoder.encode( enc );
548 }
549
550 catch ( Exception e )
551 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 550 (Poor Error Handling: Overly Broad Catch) — Low

```
552
553  return ( "Encryption error" );
```

BlindSqlInjection.java, line 341 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: BlindSqlInjection.java:341

```
338  connection = DatabaseUtilities.makeConnection(s);
339  }
340  }
341  catch (Exception e)
342  {
343  System.out.println("Exception caught: " + e);
344  e.printStackTrace(System.out);
```

JSONInjection.java, line 96 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: JSONInjection.java:96

```
93  return;
94  }
95  }
96  catch (Exception ex)
97  {
98  ex.printStackTrace();
99  }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 220 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: TraceXSS.java:220

```
217 ec.addElement(new BR());
218 ec.addElement(new HR().setWidth("90%"));
219 }
220 catch (Exception e)
221 {
222 s.setMessage("Error generating " + this.getClass().getName());
223 e.printStackTrace();
```

SqlNumericInjection.java, line 398 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: SqlNumericInjection.java:398

```
395 connection = DatabaseUtilities.makeConnection(s);
396 }
397 }
398 catch (Exception e)
399 {
400 System.out.println("Exception caught: " + e);
401 e.printStackTrace(System.out);
```

WsSqlInjection.java, line 216 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 216 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: WsSqlInjection.java:216

```
213 ec.addElement(a);
214 getLessonTracker(s).setCompleted(completed);
215 }
216 catch (Exception e)
217 {
218 s.setMessage("Error generating " + this.getClass().getName());
219 e.printStackTrace();
```

SoapRequest.java, line 431 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getResults()
File: SoapRequest.java:431

```
428 catch (SQLException sqle)
429 {}
430 }
431 catch (Exception e)
432 {}
433 return null;
434 }
```

WsSAXInjection.java, line 163 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 163 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: createContent()
File: WsSAXInjection.java:163

```
160 ec.addElement(checkXML(s, xml));
161 }
162 }
163 catch (Exception e)
164 {
165 s.setMessage("Error generating " + this.getClass().getName());
166 e.printStackTrace();
```

StoredXss.java, line 110 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: addMessage()
File: StoredXss.java:110

```
107 statement.setString(4, s.getUserName());
108 statement.executeQuery();
109 }
110 catch (Exception e)
111 {
112 // ignore the empty resultset on the insert. There are a few more SQL Injection errors
113 // that could be trapped here but we will let them try. One error would be something
```

CSRF.java, line 199 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: makeList()
File: CSRF.java:199

```
196 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

CSRF.java, line 199 (Poor Error Handling: Overly Broad Catch) — Low

```
197 }
198 }
199 catch ( Exception e )
200 {
201 s.setMessage( "Error while getting message list." );
202 }
```

UncheckedEmail.java, line 193 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: UncheckedEmail.java:193

```
190 makeSuccess(s);
191 }
192 }
193 catch (Exception e)
194 {
195 s.setMessage("Error generating " + this.getClass().getName());
196 e.printStackTrace();
```

DefaultLessonAction.java, line 323 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: isAuthorizedForEmployee()
File: DefaultLessonAction.java:323

```
320 sqle.printStackTrace();
321 }
322 }
323 catch ( Exception e )
324 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 323 (Poor Error Handling: Overly Broad Catch) — Low

```
325 s.setMessage( "Error authorizing" );
326 e.printStackTrace();
```

BasicAuthentication.java, line 157 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: doStage1()
File: BasicAuthentication.java:157

```
154 ec.addElement(b);
155
156 }
157 catch (Exception e)
158 {
159 s.setMessage("Error generating " + this.getClass().getName());
160 e.printStackTrace();
```

FailOpenAuthentication.java, line 83 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: FailOpenAuthentication.java:83

```
80 return (makeLogin(s));
81 }
82 }
83 catch (Exception e)
84 {
85 // The parameter was omitted. set fail open status complete
86 if (username.length() > 0
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 175 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: DOS_Login.java:175

```
172 sqle.printStackTrace();
173 }
174 }
175 catch (Exception e)
176 {
177 s.setMessage("Error generating " + this.getClass().getName());
178 }
```

Encoding.java, line 888 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: unicodeDecode()
File: Encoding.java:888

```
885 CharBuffer cbuf = decoder.decode( bbuf );
886 return ( cbuf.toString() );
887 }
888 catch ( Exception e )
889 {
890 return ( "Encoding problem" );
891 }
```

AbstractLesson.java, line 903 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 903 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: makeRequestDump_DELETEME()
File: AbstractLesson.java:903

```
900 el = new StringElement(readFromFile(s.getRequest().getReader(),
901 false));
902 }
903 catch (Exception e)
904 {
905 s.setMessage("Couldn't read HTTP request");
906 }
```

SoapRequest.java, line 327 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: determineType()
File: SoapRequest.java:327

```
324 //DEVNOTE: Eat the exception.
325 // ec.addElement( new P().addElement( pnfe.getMessage() ) );
326 }
327 catch (Exception e)
328 {
329 s.setMessage("Error generating " + this.getClass().getName());
330 e.printStackTrace();
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 203 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 203 (Poor Error Handling: Overly Broad Catch) — Low

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:203

```
200 sqle.printStackTrace();
201 }
202 }
203 catch (Exception e)
204 {
205 s.setMessage("Error getting employee profile");
206 e.printStackTrace();
```

FindProfile.java, line 216 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:216

```
213 sqle.printStackTrace();
214 }
215 }
216 catch (Exception e)
217 {
218 s.setMessage("Error finding employee profile");
219 e.printStackTrace();
```

CrossSiteScripting.java, line 381 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:381

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 381 (Poor Error Handling: Overly Broad Catch) — Low

```
378 System.out.println("Authorization failure");
379 ue2.printStackTrace();
380 }
381 catch (Exception e)
382 {
383 // All other errors send the user to the generic error page
384 System.out.println("handleRequest() error");
```

UpdateProfile.java, line 306 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: doChangeEmployeeProfile_BACKUP()
File: UpdateProfile.java:306

```
303 }
304
305 }
306 catch (Exception e)
307 {
308 s.setMessage("Error updating employee profile");
309 e.printStackTrace();
```

UpdateProfile.java, line 390 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createEmployeeProfile_BACKUP()
File: UpdateProfile.java:390

```
387 sqle.printStackTrace();
388 }
389 }
390 catch (Exception e)
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 390 (Poor Error Handling: Overly Broad Catch) — Low

```
391 {
392 s.setMessage("Error updating employee profile");
393 e.printStackTrace();
```

UpdateProfile.java, line 348 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:348

```
345 sqle.printStackTrace();
346 }
347 }
348 catch (Exception e)
349 {
350 s.setMessage("Error updating employee profile");
351 e.printStackTrace();
```

EditProfile.java, line 186 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile_BACKUP()
File: EditProfile.java:186

```
183 sqle.printStackTrace();
184 }
185 }
186 catch (Exception e)
187 {
188 s.setMessage("Error getting employee profile");
189 e.printStackTrace();
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

EditProfile.java, line 128 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:128

```
125 sqle.printStackTrace();
126 }
127 }
128 catch (Exception e)
129 {
130 s.setMessage("Error getting employee profile");
131 e.printStackTrace();
```

ViewProfile.java, line 143 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:143

```
140 sqle.printStackTrace();
141 }
142 }
143 catch (Exception e)
144 {
145 s.setMessage("Error getting employee profile");
146 e.printStackTrace();
```

UpdateProfile.java, line 257 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 257 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:257

```
254 }
255
256 }
257 catch (Exception e)
258 {
259 s.setMessage("Error updating employee profile");
260 e.printStackTrace();
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Login.java, line 213 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllEmployees()
File: Login.java:213

```
210 sqle.printStackTrace();
211 }
212 }
213 catch (Exception e)
214 {
215 s.setMessage("Error getting employees");
216 e.printStackTrace();
```

RoleBasedAccessControl.java, line 454 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 454 (Poor Error Handling: Overly Broad Catch) — Low

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:454

```
451 setCurrentAction(s, ERROR_ACTION);
452 ue2.printStackTrace();
453 }
454 catch (Exception e)
455 {
456 // All other errors send the user to the generic error page
457 System.out.println("handleRequest() error");
```

FindProfile.java, line 184 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:184

```
181 sqle.printStackTrace();
182 }
183 }
184 catch (Exception e)
185 {
186 s.setMessage("Error finding employee profile");
187 e.printStackTrace();
```

UpdateProfile.java, line 319 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:319

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 319 (Poor Error Handling: Overly Broad Catch) — Low

```
316 s.setMessage("Error updating employee profile");
317 }
318 }
319 catch (Exception e)
320 {
321 e.printStackTrace();
322 s.setMessage("Error updating employee profile");
```

ViewProfile.java, line 163 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:163

```
160 sqle.printStackTrace();
161 }
162 }
163 catch (Exception e)
164 {
165 s.setMessage("Error getting employee profile");
166 e.printStackTrace();
```

UpdateProfile.java, line 234 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: changeEmployeeProfile_BACKUP()
File: UpdateProfile.java:234

```
231 }
232
233 }
234 catch (Exception e)
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 234 (Poor Error Handling: Overly Broad Catch) — Low

```
235 {
236 s.setMessage("Error updating employee profile");
237 e.printStackTrace();
```

Login.java, line 166 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: login()
File: Login.java:166

```
163 sqle.printStackTrace();
164 }
165 }
166 catch (Exception e)
167 {
168 s.setMessage("Error logging in");
169 e.printStackTrace();
```

DeleteProfile.java, line 123 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: deleteEmployeeProfile()
File: DeleteProfile.java:123

```
120 sqle.printStackTrace();
121 }
122 }
123 catch (Exception e)
124 {
125 s.setMessage("Error deleting employee profile");
126 e.printStackTrace();
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 360 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:360

```
357 setCurrentAction(s, ERROR_ACTION);
358 ue2.printStackTrace();
359 }
360 catch (Exception e)
361 {
362 // All other errors send the user to the generic error page
363 System.out.println("handleRequest() error");
```

ViewProfile.java, line 223 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:223

```
220 sqle.printStackTrace();
221 }
222 }
223 catch (Exception e)
224 {
225 s.setMessage("Error getting employee profile");
226 e.printStackTrace();
```

DeleteProfile.java, line 152 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 152 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: deleteEmployeeProfile_BACKUP()
File: DeleteProfile.java:152

```
149 sqle.printStackTrace();
150 }
151 }
152 catch (Exception e)
153 {
154 s.setMessage("Error deleting employee profile");
155 e.printStackTrace();
```

UpdateProfile.java, line 185 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:185

```
182 }
183
184 }
185 catch (Exception e)
186 {
187 s.setMessage("Error updating employee profile");
188 e.printStackTrace();
```

ListStaff.java, line 166 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

ListStaff.java, line 166 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:166

```
163 sqle.printStackTrace();
164 }
165 }
166 catch (Exception e)
167 {
168 s.setMessage("Error getting employees");
169 e.printStackTrace();
```

EditProfile.java, line 128 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:128

```
125 sqle.printStackTrace();
126 }
127 }
128 catch (Exception e)
129 {
130 s.setMessage("Error getting employee profile");
131 e.printStackTrace();
```

ListStaff.java, line 118 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllEmployees()
File: ListStaff.java:118

```
115 sqle.printStackTrace();
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

ListStaff.java, line 118 (Poor Error Handling: Overly Broad Catch) — Low

```
116 }
117 }
118 catch (Exception e)
119 {
120 s.setMessage("Error getting employees");
121 e.printStackTrace();
```

EditProfile.java, line 189 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile_BACKUP()
File: EditProfile.java:189

```
186 sqle.printStackTrace();
187 }
188 }
189 catch (Exception e)
190 {
191 s.setMessage("Error getting employee profile");
192 e.printStackTrace();
```

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 255 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllEmployees()
File: Login.java:255

```
252 sqle.printStackTrace();
253 }
254 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 255 (Poor Error Handling: Overly Broad Catch) — Low

```
255 catch (Exception e)
256 {
257 s.setMessage("Error getting employees");
258 e.printStackTrace();
```

ListStaff.java, line 166 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:166

```
163 sqle.printStackTrace();
164 }
165 }
166 catch (Exception e)
167 {
168 s.setMessage("Error getting employees");
169 e.printStackTrace();
```

Login.java, line 165 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: login()
File: Login.java:165

```
162 sqle.printStackTrace();
163 }
164 }
165 catch (Exception e)
166 {
167 s.setMessage("Error logging in");
168 e.printStackTrace();
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 165 (Poor Error Handling: Overly Broad Catch) — Low

Login.java, line 208 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: login_BACKUP()
File: Login.java:208

```
205 sqle.printStackTrace();
206 }
207 }
208 catch (Exception e)
209 {
210 s.setMessage("Error logging in");
211 e.printStackTrace();
```

ListStaff.java, line 118 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllEmployees()
File: ListStaff.java:118

```
115 sqle.printStackTrace();
116 }
117 }
118 catch (Exception e)
119 {
120 s.setMessage("Error getting employees");
121 e.printStackTrace();
```

SQLInjection.java, line 363 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 363 (Poor Error Handling: Overly Broad Catch) — Low

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: handleRequest()
File: SQLInjection.java:363

```
360 System.out.println("Authorization failure");
361 ue2.printStackTrace();
362 }
363 catch (Exception e)
364 {
365 // All other errors send the user to the generic error page
366 System.out.println("handleRequest() error");
```

ViewProfile.java, line 209 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:209

```
206 sqle.printStackTrace();
207 }
208 }
209 catch (Exception e)
210 {
211 s.setMessage("Error getting employee profile");
212 e.printStackTrace();
```

ViewProfile.java, line 149 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 149 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:149

```
146 sqle.printStackTrace();
147 }
148 }
149 catch (Exception e)
150 {
151 s.setMessage("Error getting employee profile");
152 e.printStackTrace();
```

Package: org.owasp.webgoat.lessons.admin

ViewDatabase.java, line 102 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: ViewDatabase.java:102

```
99
100 }
101 }
102 catch (Exception e)
103 {
104 s.setMessage("Error generating " + this.getClass().getName());
105 e.printStackTrace();
```

UserAdminScreen.java, line 85 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.admin

UserAdminScreen.java, line 85 (Poor Error Handling: Overly Broad Catch) — Low

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: UserAdminScreen.java:85

```
82 resultsMetaData));
83 }
84 }
85 catch (Exception e)
86 {
87 s.setMessage("Error generating " + this.getClass().getName());
88 e.printStackTrace();
```

SummaryReportCardScreen.java, line 99 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: SummaryReportCardScreen.java:99

```
96 }
97 }
98 }
99 catch (Exception e)
100 {
101 e.printStackTrace();
102 }
```

RefreshDBScreen.java, line 96 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: RefreshDBScreen.java:96

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.admin

RefreshDBScreen.java, line 96 (Poor Error Handling: Overly Broad Catch) — Low

```
93 ec.addElement(t);
94 }
95 }
96 catch (Exception e)
97 {
98 s.setMessage("Error generating " + this.getClass().getName());
99 e.printStackTrace();
```

RefreshDBScreen.java, line 165 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: refreshDB()
File: RefreshDBScreen.java:165

```
162 db.makeDB(connection);
163 System.out.println("Successfully refreshed the database.");
164 }
165 catch (Exception e)
166 {
167 s.setMessage("Error refreshing database "
168 + this.getClass().getName());
```

ReportCardScreen.java, line 87 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: ReportCardScreen.java:87

```
84 user = s.getUserName();
85 }
86 }
87 catch (Exception e)
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.lessons.admin

ReportCardScreen.java, line 87 (Poor Error Handling: Overly Broad Catch) — Low

```
88 {}
89
90 if (user == null)
```

ProductsAdminScreen.java, line 85 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: createContent()
File: ProductsAdminScreen.java:85

```
82 resultsMetaData));
83 }
84 }
85 catch (Exception e)
86 {
87 s.setMessage("Error generating " + this.getClass().getName());
88 e.printStackTrace();
```

Package: org.owasp.webgoat.session

ParameterParser.java, line 356 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getIPParameter()
File: ParameterParser.java:356

```
353 {
354 return getIPParameter(name);
355 }
356 catch (Exception e)
357 {
358 return def;
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 356 (Poor Error Handling: Overly Broad Catch) — Low

```
359 }
```

ParameterParser.java, line 484 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getIntParameter()
File: ParameterParser.java:484

```
481 {
482 return getIntParameter(name);
483 }
484 catch (Exception e)
485 {
486 return def;
487 }
```

ParameterParser.java, line 752 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getSubParameter()
File: ParameterParser.java:752

```
749 {
750 return getSubParameter(first, next);
751 }
752 catch (Exception e)
753 {
754 return def;
755 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 428 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getIPParameter()
File: ParameterParser.java:428

```
425 valid = false;
426 }
427 }
428 catch (Exception e)
429 {
430 valid = false;
431 }
```

LessonTracker.java, line 403 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: store()
File: LessonTracker.java:403

```
400 {
401 out.close();
402 }
403 catch (Exception e)
404 {}
405 }
406
```

LessonTracker.java, line 251 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 251 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: load()
File: LessonTracker.java:251

```
248 {
249 // Normal if the lesson has not been accessed yet.
250 }
251 catch (Exception e)
252 {
253 System.out.println("Failed to load lesson state for " + screen);
254 e.printStackTrace();
```

ParameterParser.java, line 335 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getFloatParameter()
File: ParameterParser.java:335

```
332 {
333 return getFloatParameter(name);
334 }
335 catch (Exception e)
336 {
337 return def;
338 }
```

ParameterParser.java, line 261 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 261 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: getClassNameParameter()
File: ParameterParser.java:261

```
258 {
259 return getClassNameParameter(name);
260 }
261 catch (Exception e)
262 {
263 return def;
264 }
```

ParameterParser.java, line 859 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getWordParameter()
File: ParameterParser.java:859

```
856 {
857 return getWordParameter(name);
858 }
859 catch (Exception e)
860 {
861 return def;
862 }
```

LessonTracker.java, line 391 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: store()
File: LessonTracker.java:391

```
388 out = new FileOutputStream(fileName);
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 391 (Poor Error Handling: Overly Broad Catch) — Low

```
389 lessonProperties.store(out, s.getUserName());
390 }
391 catch (Exception e)
392 {
393 // what do we want to do, I think nothing.
394 System.out.println("Warning User data for " + s.getUserName()
```

UserTracker.java, line 161 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: deleteUser()
File: UserTracker.java:161

```
158 usersDB.close();
159
160 }
161 catch (Exception e)
162 {}
163 }
164 }
```

UserTracker.java, line 133 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getAllUsers()
File: UserTracker.java:133

```
130 }
131 usersDB.close();
132 }
133 catch (Exception e)
134 {}
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

UserTracker.java, line 133 (Poor Error Handling: Overly Broad Catch) — Low

```
135 return allUsers;
136 }
```

ErrorScreen.java, line 107 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: fixCurrentScreen()
File: ErrorScreen.java:107

```
104 {
105 s.setCurrentScreen( s.getCourse().getFirstLesson().getScreenId() );
106 }
107 catch ( Throwable t )
108 {
109 s.setCurrentScreen( WebSession.WELCOME );
110 }
```

ParameterParser.java, line 672 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getShortParameter()
File: ParameterParser.java:672

```
669 {
670 return getShortParameter(name);
671 }
672 catch (Exception e)
673 {
674 return def;
675 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

Course.java, line 377 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: loadLessons()
File: Course.java:377

```
374 }
375 }
376 }
377 catch (Exception e)
378 {
379 //System.out.println("Warning: " + e.getMessage());
380 }
```

ParameterParser.java, line 179 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getByteParameter()
File: ParameterParser.java:179

```
176 {
177 return getByteParameter(name);
178 }
179 catch (Exception e)
180 {
181 return def;
182 }
```

LessonTracker.java, line 262 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 262 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: load()
File: LessonTracker.java:262

```
259 {
260 in.close();
261 }
262 catch (Exception e)
263 {}
264 }
265
```

WebSession.java, line 1027 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: update()
File: WebSession.java:1027

```
1024 }
1025 }
1026 }
1027 catch ( Exception e )
1028 {
1029 }
1030
```

ParameterParser.java, line 521 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 521 (Poor Error Handling: Overly Broad Catch) — Low

Sink: CatchBlock
Enclosing Method: getLongParameter()
File: ParameterParser.java:521

```
518 {
519 return getLongParameter(name);
520 }
521 catch (Exception e)
522 {
523 return def;
524 }
```

ParameterParser.java, line 610 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getRawParameter()
File: ParameterParser.java:610

```
607 {
608 return getRawParameter(name);
609 }
610 catch (Exception e)
611 {
612 return def;
613 }
```

ParameterParser.java, line 731 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getStringParameter()
File: ParameterParser.java:731

```
728 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 731 (Poor Error Handling: Overly Broad Catch) — Low

```
729 return getStringParameter(name);
730 }
731 catch (Exception e)
732 {
733 return def;
734 }
```

ParameterParser.java, line 298 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getDoubleParameter()
File: ParameterParser.java:298

```
295 {
296 return getDoubleParameter(name);
297 }
298 catch (Exception e)
299 {
300 return def;
301 }
```

ParameterParser.java, line 223 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getCharParameter()
File: ParameterParser.java:223

```
220 {
221 return getCharParameter(name);
222 }
223 catch (Exception e)
224 {
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 223 (Poor Error Handling: Overly Broad Catch) — Low

```
225 return def;
226 }
```

ParameterParser.java, line 120 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getBooleanParameter()
File: ParameterParser.java:120

```
117 {
118 return getBooleanParameter(name);
119 }
120 catch (Exception e)
121 {
122 return def;
123 }
```

ParameterParser.java, line 141 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getBooleanSubParameter()
File: ParameterParser.java:141

```
138 {
139 return new Boolean(getSubParameter(first, next)).booleanValue();
140 }
141 catch (Exception e)
142 {
143 return def;
144 }
```

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

WebSession.java, line 445 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:445

```
442
443 return realConnectionString;
444 }
445 catch ( Exception e )
446 {
447 System.out.println( "Couldn't open database: check web.xml database parameters" );
448 e.printStackTrace();
```

DatabaseUtilities.java, line 108 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:108

```
105 return DriverManager.getConnection("jdbc:idb:" + dbName);
106 }
107 }
108 catch (Exception e)
109 {
110 e.printStackTrace();
111 return null;
```

ParameterParser.java, line 882 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.session

ParameterParser.java, line 882 (Poor Error Handling: Overly Broad Catch) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: getRegexParameter()
File: ParameterParser.java:882

```
879 {
880 return getRegexParameter(name, regexpattern);
881 }
882 catch (Exception e)
883 {
884 //System.out.println("Exception occured in defined pattern match");
885 //e.printStackTrace();
```

Package: org.owasp.webgoat.util

Exec.java, line 242 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: CatchBlock
Enclosing Method: execOptions()
File: Exec.java:242

```
239 {
240 results.setInterrupted();
241 }
242 catch (Throwable t)
243 {
244 results.setThrowable(t);
245 }
```

Exec.java, line 431 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Poor Error Handling: Overly Broad Catch — Low

Package: org.owasp.webgoat.util

Exec.java, line 431 (Poor Error Handling: Overly Broad Catch) — Low

Sink Details

Sink: CatchBlock
Enclosing Method: execOptions()
File: Exec.java:431

```
428 {
429 results.setInterrupted();
430 }
431 catch (Throwable t)
432 {
433 results.setThrowable(t);
434 }
```

Poor Error Handling: Overly Broad Throws (15 issues)

Abstract

The method throws a generic exception making it harder for callers to do a good job of error handling and recovery.

Explanation

Declaring a method to throw `Exception` or `Throwable` makes it difficult for callers to do good error handling and error recovery. Java's exception mechanism is set up to make it easy for callers to anticipate what can go wrong and write code to handle each specific exceptional circumstance. Declaring that a method throws a generic form of exception defeats this system.

Example: The following method throws three types of exceptions.

```
public void doExchange()
  throws IOException, InvocationTargetException,
         SQLException {
  ...
}
```

While it might seem tidier to write

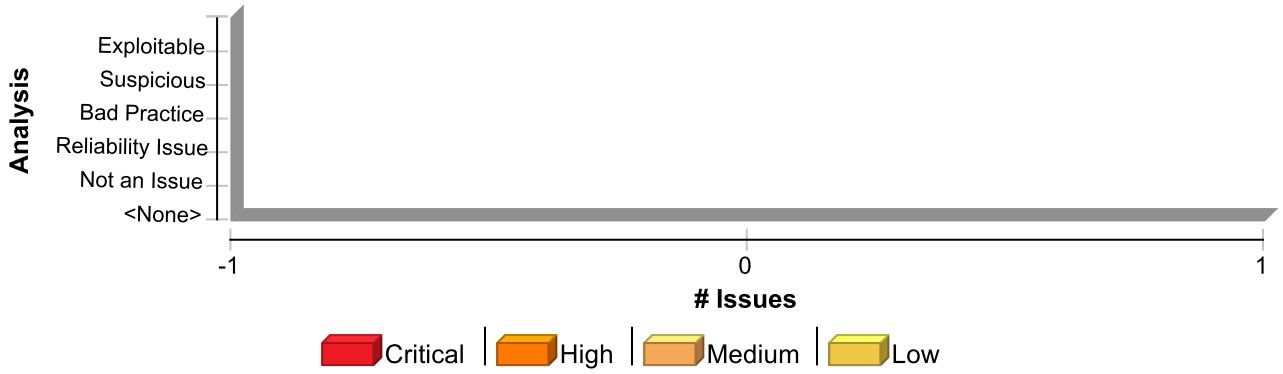
```
public void doExchange()
  throws Exception {
  ...
}
```

doing so hampers the caller's ability to understand and handle the exceptions that occur. Further, if a later revision of `doExchange()` introduces a new type of exception that should be treated differently than previous exceptions, there is no easy way to enforce this requirement.

Recommendation

Do not declare methods to throw `Exception` or `Throwable`. If the exceptions thrown by a method are not recoverable or should not generally be caught by the caller, consider throwing unchecked exceptions rather than checked exceptions. This can be accomplished by implementing exception classes that extend `RuntimeException` or `Error` instead of `Exception`, or add a try/catch wrapper in your method to convert checked exceptions to unchecked exceptions.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Error Handling: Overly Broad Throws	15	15	0	30
Total	**15**	**15**	**0**	**30**

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

BackDoors.java, line 87 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: concept1
Enclosing Method: concept1()
File: BackDoors.java:87

```
84 }
85
86
87 protected Element concept1(WebSession s) throws Exception
88 {
89 ElementContainer ec = new ElementContainer();
90
```

LessonAdapter.java, line 168 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 168 (Poor Error Handling: Overly Broad Throws) — Low

Sink: Function: doStage4
Enclosing Method: doStage4()
File: LessonAdapter.java:168

```
165 }
166
167
168 protected Element doStage4(WebSession s) throws Exception
169 {
170 ElementContainer ec = new ElementContainer();
171 ec.addElement("Stage 4 Stub");
```

LessonAdapter.java, line 144 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: doStage1
Enclosing Method: doStage1()
File: LessonAdapter.java:144

```
141 }
142
143
144 protected Element doStage1(WebSession s) throws Exception
145 {
146 ElementContainer ec = new ElementContainer();
147 ec.addElement("Stage 1 Stub");
```

BackDoors.java, line 142 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: concept2
Enclosing Method: concept2()
File: BackDoors.java:142

```
139 }
```

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

BackDoors.java, line 142 (Poor Error Handling: Overly Broad Throws) — Low

```
140
141
142 protected Element concept2(WebSession s) throws Exception
143 {
144 ElementContainer ec = new ElementContainer();
145 ec.addElement(makeUsername(s));
```

LessonAdapter.java, line 160 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: doStage3
Enclosing Method: doStage3()
File: LessonAdapter.java:160

```
157 }
158
159
160 protected Element doStage3(WebSession s) throws Exception
161 {
162 ElementContainer ec = new ElementContainer();
163 ec.addElement("Stage 3 Stub");
```

HtmlClues.java, line 135 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: makeUser
Enclosing Method: makeUser()
File: HtmlClues.java:135

```
132 * @return Description of the Return Value
133 * @exception Exception Description of the Exception
134 */
135 protected Element makeUser(WebSession s, String user, String method)
136 throws Exception
```

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

HtmlClues.java, line 135 (Poor Error Handling: Overly Broad Throws) — Low

```
137 {
138 ElementContainer ec = new ElementContainer();
```

WeakAuthenticationCookie.java, line 85 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: checkCookie
Enclosing Method: checkCookie()
File: WeakAuthenticationCookie.java:85

```
82 * @return Description of the Return Value
83 * @exception Exception Description of the Exception
84 */
85 protected String checkCookie(WebSession s) throws Exception
86 {
87 String cookie = getCookie(s);
88
```

LessonAdapter.java, line 152 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: doStage2
Enclosing Method: doStage2()
File: LessonAdapter.java:152

```
149 }
150
151
152 protected Element doStage2(WebSession s) throws Exception
153 {
154 ElementContainer ec = new ElementContainer();
155 ec.addElement("Stage 2 Stub");
```

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 124 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: checkParams
Enclosing Method: checkParams()
File: WeakAuthenticationCookie.java:124

```
121 * @return Description of the Return Value
122 * @exception Exception Description of the Exception
123 */
124 protected String checkParams(WebSession s) throws Exception
125 {
126 String username = s.getParser().getStringParameter(USERNAME, "");
127 String password = s.getParser().getStringParameter(PASSWORD, "");
```

LessonAdapter.java, line 184 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: doStage6
Enclosing Method: doStage6()
File: LessonAdapter.java:184

```
181 }
182
183
184 protected Element doStage6(WebSession s) throws Exception
185 {
186 ElementContainer ec = new ElementContainer();
187 ec.addElement("Stage 6 Stub");
```

HttpSplitting.java, line 191 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

HttpSplitting.java, line 191 (Poor Error Handling: Overly Broad Throws) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: doCachePoisining
Enclosing Method: doCachePoisining()
File: HttpSplitting.java:191

```
188 }
189
190
191 protected Element doCachePoisining(WebSession s) throws Exception
192 {
193 ElementContainer ec = new ElementContainer();
194
```

HttpSplitting.java, line 161 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: createAttackEnvironment
Enclosing Method: createAttackEnvironment()
File: HttpSplitting.java:161

```
158 }
159
160
161 protected Element createAttackEnvironment(WebSession s) throws Exception
162 {
163 ElementContainer ec = new ElementContainer();
164 String lang = null;
```

Challenge2Screen.java, line 397 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Error Handling: Overly Broad Throws — Low

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 397 (Poor Error Handling: Overly Broad Throws) — Low

Sink: Function: showDefaceAttempt
Enclosing Method: showDefaceAttempt()
File: Challenge2Screen.java:397

```
394 }
395
396
397 private Element showDefaceAttempt(WebSession s) throws Exception
398 {
399 ElementContainer ec = new ElementContainer();
400
```

WeakAuthenticationCookie.java, line 373 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: makeUser
Enclosing Method: makeUser()
File: WeakAuthenticationCookie.java:373

```
370 * @return Description of the Return Value
371 * @exception Exception Description of the Exception
372 */
373 protected Element makeUser(WebSession s, String user, String method)
374 throws Exception
375 {
376 ElementContainer ec = new ElementContainer();
```

LessonAdapter.java, line 176 (Poor Error Handling: Overly Broad Throws) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Function: doStage5
Enclosing Method: doStage5()
File: LessonAdapter.java:176

```
173 }
```

Poor Error Handling: Overly Broad Throws	Low
Package: org.owasp.webgoat.lessons	
LessonAdapter.java, line 176 (Poor Error Handling: Overly Broad Throws)	**Low**

```
174
175
176  protected Element doStage5(WebSession s) throws Exception
177  {
178  ElementContainer ec = new ElementContainer();
179  ec.addElement("Stage 5 Stub");
```

Poor Error Handling: Throw Inside Finally (1 issue)

Abstract

Using a `throw` statement inside a `finally` block breaks the logical progression through the `try-catch-finally`.

Explanation

In Java, `finally` blocks are always executed after their corresponding `try-catch` blocks and are often used to free allocated resources, such as file handles or database cursors. Throwing an exception in a `finally` block can bypass critical cleanup code since normal program execution will be disrupted.

Example 1: In the following code, the call to `stmt.close()` is bypassed when the `FileNotFoundException` is thrown.

```
public void processTransaction(Connection conn) throws FileNotFoundException
{
    FileInputStream fis = null;
    Statement stmt = null;
    try
    {
        stmt = conn.createStatement();
        fis = new FileInputStream("badFile.txt");
        ...
    }
    catch (FileNotFoundException fe)
    {
        log("File not found.");
    }
    catch (SQLException se)
    {
        //handle error
    }
    finally
    {
        if (fis == null)
        {
            throw new FileNotFoundException();
        }

        if (stmt != null)
        {
            try
            {
                stmt.close();
            }
            catch (SQLException e)
            {
                log(e);
            }
        }
    }
}
```

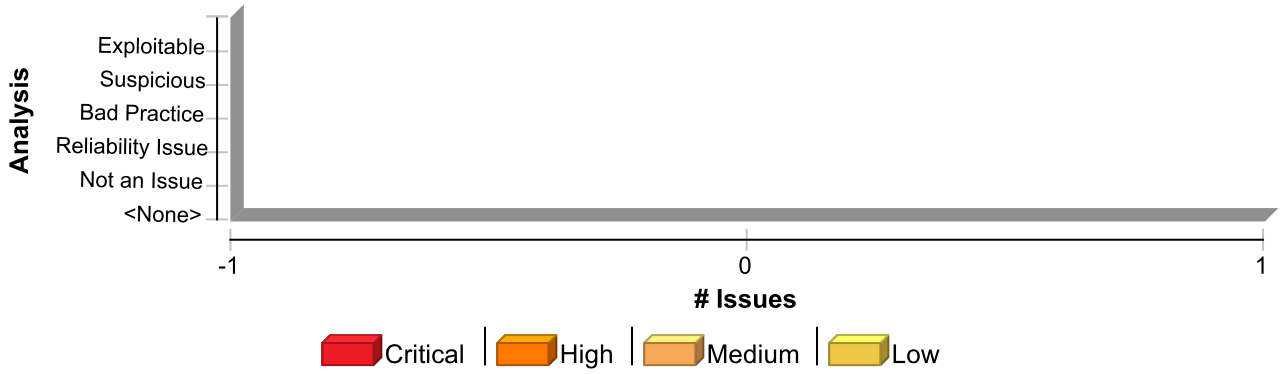
This category is from the Cigital Java Rulepack. http://www.cigital.com/

Recommendation

Never throw exceptions from within `finally` blocks. If you must re-throw an exception, do it inside a `catch` block so as not to interrupt the normal execution of the `finally` block. **Example 2:** The following code re-throws the `FileNotFoundException` in the `catch` block.

```
public void processTransaction(Connection conn) throws FileNotFoundException
{
    FileInputStream fis = null;
    Statement stmt = null;
    try
    {
        stmt = conn.createStatement();
        fis = new FileInputStream("badFile.txt");
        ...
    }
    catch (FileNotFoundException fe)
    {
        log("File not found.");
        throw fe;
    }
    catch (SQLException se)
    {
        //handle error
    }
    finally
    {
        if (fis != null)
        {
            try
            {
                fis.close();
            }
            catch (IOException ie)
            {
                log(ie);
            }
        }

        if (stmt != null)
        {
            try
            {
                stmt.close();
            }
            catch (SQLException e)
            {
                log(e);
            }
        }
    }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Error Handling: Throw Inside Finally	1	1	0	2
Total	**1**	**1**	**0**	**2**

Poor Error Handling: Throw Inside Finally — Low

Package: org.owasp.webgoat.util

Interceptor.java, line 118 (Poor Error Handling: Throw Inside Finally) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FinallyBlock
Enclosing Method: doFilter()
File: Interceptor.java:118

```
115 e.printStackTrace();
116 }
117 finally
118 {
119 if (out != null)
120 {
121 out.close();
```

Poor Logging Practice: Use of a System Output Stream (137 issues)

Abstract

Using `System.out` or `System.err` rather than a dedicated logging facility makes it difficult to monitor the behavior of the program.

Explanation

Example 1: The first Java program that a developer learns to write often looks like this:

```
public class MyClass
  public static void main(String[] args) {
    System.out.println("hello world");
  }
}
```

While most programmers go on to learn many nuances and subtleties about Java, a surprising number hang on to this first lesson and never give up on writing messages to standard output using `System.out.println()`.

The problem is that writing directly to standard output or standard error is often used as an unstructured form of logging. Structured logging facilities provide features like logging levels, uniform formatting, a logger identifier, timestamps, and, perhaps most critically, the ability to direct the log messages to the right place. When the use of system output streams is jumbled together with the code that uses loggers properly, the result is often a well-kept log that is missing critical information.

Developers widely accept the need for structured logging, but many continue to use system output streams in their "pre-production" development. If the code you are reviewing is past the initial phases of development, use of `System.out` or `System.err` may indicate an oversight in the move to a structured logging system.

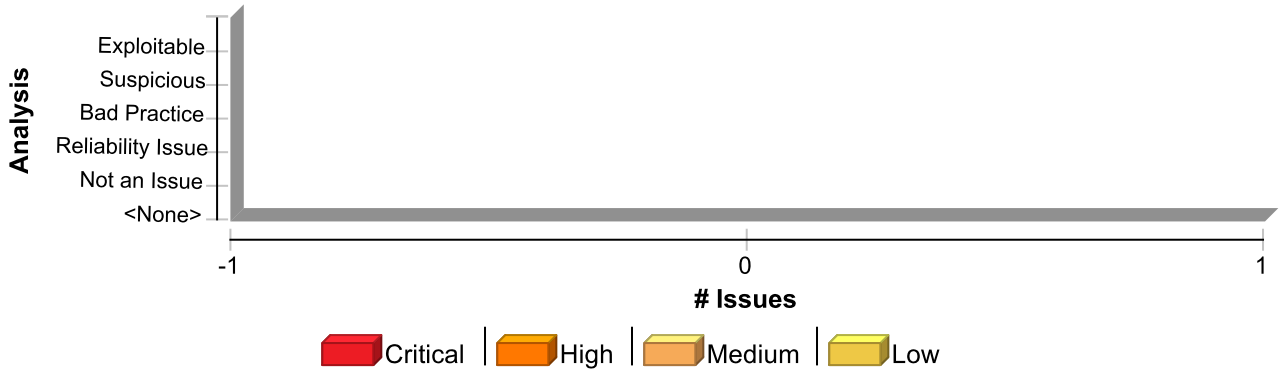
Recommendation

Use a Java logging facility rather than `System.out` or `System.err`.

Example 2: For example, the "hello world" program above can be re-written using log4j like this:

```
import org.apache.log4j.Logger;
import org.apache.log4j.BasicConfigurator;

public class MyClass {
  private final static Logger logger =
            Logger.getLogger(MyClass.class);

  public static void main(String[] args) {
    BasicConfigurator.configure();
    logger.info("hello world");
  }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Logging Practice: Use of a System Output Stream	137	137	0	274
Total	**137**	**137**	**0**	**274**

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat

HammerHead.java, line 245 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: dumpSession()
File: HammerHead.java:245

```
242 {
243 String name = (String) enumerator.nextElement();
244 Object value = session.getAttribute(name);
245 System.out.println("Name: " + name);
246 System.out.println("Value: " + value);
247 }
248 }
```

HammerHead.java, line 246 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat

HammerHead.java, line 246 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: dumpSession()
File: HammerHead.java:246

```
243 String name = (String) enumerator.nextElement();
244 Object value = session.getAttribute(name);
245 System.out.println("Name: " + name);
246 System.out.println("Value: " + value);
247 }
248 }
249
```

HammerHead.java, line 307 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: log()
File: HammerHead.java:307

```
304 String output = new Date() + " | " + request.getRemoteHost() + ":"
305 + request.getRemoteAddr() + " | " + message;
306 log(output);
307 System.out.println(output);
308 }
309
310
```

Package: org.owasp.webgoat.lessons

Encoding.java, line 769 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 769 (Poor Logging Practice: Use of a System Output Stream) — Low

File: Encoding.java:769

```
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
767 System.out.print( "Double unicode encoding is..." );
768 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
769 System.out.print( "Double URL encoding: " );
770 System.out.println( urlEncode( urlEncode( userInput ) ) + " : " + urlDecode( urlDecode( userInput ) ) );
771 }
772 catch ( Exception e )
```

CommandInjection.java, line 285 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: exec()
File: CommandInjection.java:285

```
282 */
283 private String exec(WebSession s, String[] command)
284 {
285 System.out.println("Executing OS command: " + Arrays.asList(command));
286 ExecResults er = Exec.execSimple(command);
287 if (!er.getError())
288 {
```

CommandInjection.java, line 264 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: exec()
File: CommandInjection.java:264

```
261 */
262 private String exec(WebSession s, String command)
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 264 (Poor Logging Practice: Use of a System Output Stream) — Low

```
263 {
264 System.out.println("Executing OS command: " + command);
265 ExecResults er = Exec.execSimple(command);
266 if ((command.indexOf("&") != -1 || command.indexOf(";") != -1)
267 && !er.getError())
```

Encoding.java, line 755 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:755

```
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
753 System.out.print( "MD5 hash: " );
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
755 System.out.print( "SHA-256 hash: " );
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
757 System.out.print( "Unicode encoding: " );
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
```

Encoding.java, line 770 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:770

```
767 System.out.print( "Double unicode encoding is..." );
768 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
769 System.out.print( "Double URL encoding: " );
770 System.out.println( urlEncode( urlEncode( userInput ) ) + " : " + urlDecode( urlDecode( userInput ) ) );
771 }
772 catch ( Exception e )
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 770 (Poor Logging Practice: Use of a System Output Stream) — Low

```
773 {
```

Encoding.java, line 749 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:749

```
746 System.out.println( "Working with: " + userInput );
747 System.out.print( "Base64 encoding: " );
748 System.out.println( base64Encode( userInput ) + " : " + base64Decode( userInput ) );
749 System.out.print( "Entity encoding: " );
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
751 System.out.print( "Password based encryption (PBE): " );
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
```

Encoding.java, line 763 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:763

```
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
761 System.out.print( "Hex encoding: " );
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
763 System.out.print( "Rot13 encoding: " );
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
765 System.out.print( "XOR with password: " );
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 400 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SqlNumericInjection.java:400

```
397 }
398 catch (Exception e)
399 {
400 System.out.println("Exception caught: " + e);
401 e.printStackTrace(System.out);
402 }
403 }
```

AbstractLesson.java, line 478 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: getFileText()
File: AbstractLesson.java:478

```
475 }
476 catch (Exception e)
477 {
478 System.out.println(e);
479 e.printStackTrace();
480 }
481
```

Encoding.java, line 767 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 767 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:767

```
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
765 System.out.print( "XOR with password: " );
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
767 System.out.print( "Double unicode encoding is..." );
768 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
769 System.out.print( "Double URL encoding: " );
770 System.out.println( urlEncode( urlEncode( userInput ) ) + " : " + urlDecode( urlDecode( userInput ) ) );
```

Encoding.java, line 751 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:751

```
748 System.out.println( base64Encode( userInput ) + " : " + base64Decode( userInput ) );
749 System.out.print( "Entity encoding: " );
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
751 System.out.print( "Password based encryption (PBE): " );
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
753 System.out.print( "MD5 hash: " );
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
```

Encoding.java, line 753 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 753 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:753

```
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
751 System.out.print( "Password based encryption (PBE): " );
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
753 System.out.print( "MD5 hash: " );
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
755 System.out.print( "SHA-256 hash: " );
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
```

Encoding.java, line 766 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:766

```
763 System.out.print( "Rot13 encoding: " );
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
765 System.out.print( "XOR with password: " );
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
767 System.out.print( "Double unicode encoding is..." );
768 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
769 System.out.print( "Double URL encoding: " );
```

Encoding.java, line 758 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:758

```
755 System.out.print( "SHA-256 hash: " );
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 758 (Poor Logging Practice: Use of a System Output Stream) — Low

```
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
757 System.out.print( "Unicode encoding: " );
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
759 System.out.print( "URL encoding: " );
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
761 System.out.print( "Hex encoding: " );
```

Encoding.java, line 768 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:768

```
765 System.out.print( "XOR with password: " );
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
767 System.out.print( "Double unicode encoding is..." );
768 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
769 System.out.print( "Double URL encoding: " );
770 System.out.println( urlEncode( urlEncode( userInput ) ) + " : " + urlDecode( urlDecode( userInput ) ) );
771 }
```

AbstractLesson.java, line 960 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: makeSourceDump_DELETEME()
File: AbstractLesson.java:960

```
957 }
958 catch (IOException e)
959 {
960 System.out.println("reading file EXCEPTION: " + filename);
961 s.setMessage("Could not find source file");
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 960 (Poor Logging Practice: Use of a System Output Stream) — Low

```
962 }
963
```

Encoding.java, line 750 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:750

```
747 System.out.print( "Base64 encoding: " );
748 System.out.println( base64Encode( userInput ) + " : " + base64Decode( userInput ) );
749 System.out.print( "Entity encoding: " );
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
751 System.out.print( "Password based encryption (PBE): " );
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
753 System.out.print( "MD5 hash: " );
```

BlindSqlInjection.java, line 343 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: BlindSqlInjection.java:343

```
340 }
341 catch (Exception e)
342 {
343 System.out.println("Exception caught: " + e);
344 e.printStackTrace(System.out);
345 }
346 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 746 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:746

```
743 {
744 String userInput = args[0];
745 String userKey = args[1];
746 System.out.println( "Working with: " + userInput );
747 System.out.print( "Base64 encoding: " );
748 System.out.println( base64Encode( userInput ) + " : " + base64Decode( userInput ) );
749 System.out.print( "Entity encoding: " );
```

Encoding.java, line 765 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:765

```
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
763 System.out.print( "Rot13 encoding: " );
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
765 System.out.print( "XOR with password: " );
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
767 System.out.print( "Double unicode encoding is..." );
768 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
```

ThreadSafetyProblem.java, line 216 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

ThreadSafetyProblem.java, line 216 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: ThreadSafetyProblem.java:216

```
213 }
214 catch (Exception e)
215 {
216 System.out.println("Exception caught: " + e);
217 e.printStackTrace(System.out);
218 }
219 }
```

Encoding.java, line 748 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:748

```
745 String userKey = args[1];
746 System.out.println( "Working with: " + userInput );
747 System.out.print( "Base64 encoding: " );
748 System.out.println( base64Encode( userInput ) + " : " + base64Decode( userInput ) );
749 System.out.print( "Entity encoding: " );
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
751 System.out.print( "Password based encryption (PBE): " );
```

Encoding.java, line 747 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 747 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:747

```
744 String userInput = args[0];
745 String userKey = args[1];
746 System.out.println( "Working with: " + userInput );
747 System.out.print( "Base64 encoding: " );
748 System.out.println( base64Encode( userInput ) + " : " + base64Decode( userInput ) );
749 System.out.print( "Entity encoding: " );
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
```

Encoding.java, line 759 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:759

```
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
757 System.out.print( "Unicode encoding: " );
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
759 System.out.print( "URL encoding: " );
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
761 System.out.print( "Hex encoding: " );
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
```

Encoding.java, line 754 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:754

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 754 (Poor Logging Practice: Use of a System Output Stream) — Low

```
751 System.out.print( "Password based encryption (PBE): " );
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
753 System.out.print( "MD5 hash: " );
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
755 System.out.print( "SHA-256 hash: " );
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
757 System.out.print( "Unicode encoding: " );
```

CommandInjection.java, line 306 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: exec()
File: CommandInjection.java:306

```
303 */
304 private Element exec(WebSession s, String command, String args)
305 {
306 System.out.println("Executing OS command: '" + command
307 + "' with args: '" + args + "'");
308 ExecResults er = Exec.execSimple(command, args);
309 if ((args.indexOf("&") != -1 || args.indexOf(";") != -1)
```

Encoding.java, line 761 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:761

```
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
759 System.out.print( "URL encoding: " );
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
761 System.out.print( "Hex encoding: " );
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 761 (Poor Logging Practice: Use of a System Output Stream) — Low

```
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
763 System.out.print( "Rot13 encoding: " );
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
```

CommandInjection.java, line 87 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createContent()
File: CommandInjection.java:87

```
84 }
85 index = index + 1;
86 int helpFileLen = helpFile.length() - 1; // subtract 1 for the closing quote
87 System.out.println("Command = ["
88 + helpFile.substring(index, helpFileLen).trim()
89 .toLowerCase() + "]");
90 if ((osName.indexOf("Windows") != -1 && (helpFile.substring(
```

Encoding.java, line 760 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:760

```
757 System.out.print( "Unicode encoding: " );
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
759 System.out.print( "URL encoding: " );
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
761 System.out.print( "Hex encoding: " );
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
763 System.out.print( "Rot13 encoding: " );
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 757 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: print
Enclosing Method: main()
File: Encoding.java:757

```
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
755 System.out.print( "SHA-256 hash: " );
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
757 System.out.print( "Unicode encoding: " );
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
759 System.out.print( "URL encoding: " );
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
```

Encoding.java, line 764 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:764

```
761 System.out.print( "Hex encoding: " );
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
763 System.out.print( "Rot13 encoding: " );
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
765 System.out.print( "XOR with password: " );
766 System.out.println( xorEncode( userInput, userKey ) + " : " + xorDecode( userInput, userKey ) );
767 System.out.print( "Double unicode encoding is..." );
```

AbstractLesson.java, line 1044 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 1044 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: readFromURL()
File: AbstractLesson.java:1044

```
1041 }
1042 catch (Exception e)
1043 {
1044 System.out.println(e);
1045 e.printStackTrace();
1046 }
1047
```

Encoding.java, line 756 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:756

```
753 System.out.print( "MD5 hash: " );
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
755 System.out.print( "SHA-256 hash: " );
756 System.out.println( hashSHA( userInput ) + " : " + "Cannot reverse a hash" );
757 System.out.print( "Unicode encoding: " );
758 System.out.println( "Not Implemented" + " : " + "Not Implemented" );
759 System.out.print( "URL encoding: " );
```

Encoding.java, line 762 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 762 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:762

```
759 System.out.print( "URL encoding: " );
760 System.out.println( urlEncode( userInput ) + " : " + urlDecode( userInput ) );
761 System.out.print( "Hex encoding: " );
762 System.out.println( hexEncode( userInput ) + " : " + hexDecode( userInput ) );
763 System.out.print( "Rot13 encoding: " );
764 System.out.println( rot13( userInput ) + " : " + rot13( userInput ) );
765 System.out.print( "XOR with password: " );
```

Encoding.java, line 752 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Encoding.java:752

```
749 System.out.print( "Entity encoding: " );
750 System.out.println( HtmlEncoder.encode( userInput ) + " : " + HtmlEncoder.decode( userInput ) );
751 System.out.print( "Password based encryption (PBE): " );
752 System.out.println( encryptString( userInput, userKey ) + " : " + decryptString( userInput, userKey ) );
753 System.out.print( "MD5 hash: " );
754 System.out.println( hashMD5( userInput ) + " : " + "Cannot reverse a hash" );
755 System.out.print( "SHA-256 hash: " );
```

LessonAdapter.java, line 136 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createStagedContent()
File: LessonAdapter.java:136

```
133 catch (Exception e)
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 136 (Poor Logging Practice: Use of a System Output Stream) — Low

```
134 {
135 s.setMessage("Error generating " + this.getClass().getName());
136 System.out.println(e);
137 e.printStackTrace();
138 }
139
```

AbstractLesson.java, line 422 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: getFileMethod()
File: AbstractLesson.java:422

```
419 }
420 catch (Exception e)
421 {
422 System.out.println(e);
423 e.printStackTrace();
424 }
425
```

SqlStringInjection.java, line 316 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SqlStringInjection.java:316

```
313 }
314 catch (Exception e)
315 {
316 System.out.println("Exception caught: " + e);
317 e.printStackTrace(System.out);
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 316 (Poor Logging Practice: Use of a System Output Stream) — Low

```
318 }
319 }
```

Challenge2Screen.java, line 728 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: sendMessage()
File: Challenge2Screen.java:728

```
725 }
726 catch (Exception e)
727 {
728 System.out.println("Couldn't write " + message + " to " + s);
729 e.printStackTrace();
730 }
731 }
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 359 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:359

```
356 }
357 catch (ParameterNotFoundException pnfe)
358 {
359 System.out.println("Missing parameter");
360 pnfe.printStackTrace();
361 setCurrentAction(s, ERROR_ACTION);
362 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 378 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:378

```
375 catch (UnauthorizedException ue2)
376 {
377 s.setMessage("You are not authorized to perform this function");
378 System.out.println("Authorization failure");
379 ue2.printStackTrace();
380 }
381 catch (Exception e)
```

CrossSiteScripting.java, line 384 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:384

```
381 catch (Exception e)
382 {
383 // All other errors send the user to the generic error page
384 System.out.println("handleRequest() error");
385 e.printStackTrace();
386 setCurrentAction(s, ERROR_ACTION);
387 }
```

CrossSiteScripting.java, line 365 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 365 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:365

```
362 }
363 catch (ValidationException ve)
364 {
365 System.out.println("Validation failed");
366 ve.printStackTrace();
367 setCurrentAction(s, ERROR_ACTION);
368 }
```

FindProfile.java, line 120 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: FindProfile.java:120

```
117 }
118 catch (UnauthenticatedException ue1)
119 {
120 System.out.println("Internal server error");
121 ue1.printStackTrace();
122 }
123 catch (UnauthorizedException ue2)
```

FindProfile.java, line 125 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

FindProfile.java, line 125 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: FindProfile.java:125

```
122 }
123 catch (UnauthorizedException ue2)
124 {
125 System.out.println("Internal server error");
126 ue2.printStackTrace();
127 }
128 }
```

CrossSiteScripting.java, line 372 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:372

```
369 catch (UnauthenticatedException ue)
370 {
371 s.setMessage("Login failed");
372 System.out.println("Authentication failure");
373 ue.printStackTrace();
374 }
375 catch (UnauthorizedException ue2)
```

UpdateProfile.java, line 109 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: UpdateProfile.java:109

```
106 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 109 (Poor Logging Practice: Use of a System Output Stream) — Low

```
107 catch (UnauthenticatedException ue1)
108 {
109 System.out.println("Internal server error");
110 ue1.printStackTrace();
111 }
112 catch (UnauthorizedException ue2)
```

UpdateProfile.java, line 114 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: UpdateProfile.java:114

```
111 }
112 catch (UnauthorizedException ue2)
113 {
114 System.out.println("Internal server error");
115 ue2.printStackTrace();
116 }
117 }
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 450 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:450

```
447 catch (UnauthorizedException ue2)
448 {
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 450 (Poor Logging Practice: Use of a System Output Stream) — Low

```
449 s.setMessage("You are not authorized to perform this function");
450 System.out.println("Authorization failure");
451 setCurrentAction(s, ERROR_ACTION);
452 ue2.printStackTrace();
453 }
```

RoleBasedAccessControl.java, line 437 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:437

```
434 }
435 catch (ValidationException ve)
436 {
437 System.out.println("Validation failed");
438 ve.printStackTrace();
439 setCurrentAction(s, ERROR_ACTION);
440 }
```

Logout.java, line 70 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: Logout.java:70

```
67 }
68 catch (UnauthenticatedException ue1)
69 {
70 System.out.println("Internal server error");
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Logout.java, line 70 (Poor Logging Practice: Use of a System Output Stream) — Low

```
71 ue1.printStackTrace();
72 }
73 catch (UnauthorizedException ue2)
```

Logout.java, line 75 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: Logout.java:75

```
72 }
73 catch (UnauthorizedException ue2)
74 {
75 System.out.println("Internal server error");
76 ue2.printStackTrace();
77 }
78
```

UpdateProfile.java, line 124 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: UpdateProfile.java:124

```
121 }
122 catch (UnauthenticatedException ue1)
123 {
124 System.out.println("Internal server error");
125 ue1.printStackTrace();
126 }
127 catch (UnauthorizedException ue2)
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 129 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: UpdateProfile.java:129

```
126 }
127 catch (UnauthorizedException ue2)
128 {
129 System.out.println("Internal server error");
130 ue2.printStackTrace();
131 }
132 }
```

RoleBasedAccessControl.java, line 350 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:350

```
347 catch (UnauthenticatedException ue)
348 {
349 s.setMessage("Login failed");
350 System.out.println("Authentication failure");
351 ue.printStackTrace();
352 }
353 catch (UnauthorizedException ue2)
```

RoleBasedAccessControl.java, line 431 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 431 (Poor Logging Practice: Use of a System Output Stream) — Low

Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:431

```
428 }
429 catch (ParameterNotFoundException pnfe)
430 {
431 System.out.println("Missing parameter");
432 pnfe.printStackTrace();
433 setCurrentAction(s, ERROR_ACTION);
434 }
```

RoleBasedAccessControl.java, line 343 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:343

```
340 }
341 catch (ValidationException ve)
342 {
343 System.out.println("Validation failed");
344 ve.printStackTrace();
345 setCurrentAction(s, ERROR_ACTION);
346 }
```

DeleteProfile.java, line 80 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 80 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: DeleteProfile.java:80

```
77 }
78 catch (UnauthenticatedException ue1)
79 {
80 System.out.println("Internal server error");
81 ue1.printStackTrace();
82 }
83 catch (UnauthorizedException ue2)
```

DeleteProfile.java, line 85 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: DeleteProfile.java:85

```
82 }
83 catch (UnauthorizedException ue2)
84 {
85 System.out.println("Internal server error");
86 ue2.printStackTrace();
87 }
88 }
```

Login.java, line 90 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Login.java, line 90 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: Login.java:90

```
87 }
88 catch (UnauthenticatedException ue1)
89 {
90 System.out.println("Internal server error");
91 ue1.printStackTrace();
92 }
93 catch (UnauthorizedException ue2)
```

Login.java, line 95 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: Login.java:95

```
92 }
93 catch (UnauthorizedException ue2)
94 {
95 System.out.println("Internal server error");
96 ue2.printStackTrace();
97 }
98 }
```

RoleBasedAccessControl.java, line 444 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:444

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 444 (Poor Logging Practice: Use of a System Output Stream) — Low

```
441 catch (UnauthenticatedException ue)
442 {
443 s.setMessage("Login failed");
444 System.out.println("Authentication failure");
445 ue.printStackTrace();
446 }
447 catch (UnauthorizedException ue2)
```

RoleBasedAccessControl.java, line 363 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:363

```
360 catch (Exception e)
361 {
362 // All other errors send the user to the generic error page
363 System.out.println("handleRequest() error");
364 e.printStackTrace();
365 setCurrentAction(s, ERROR_ACTION);
366 }
```

RoleBasedAccessControl.java, line 356 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:356

```
353 catch (UnauthorizedException ue2)
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 356 (Poor Logging Practice: Use of a System Output Stream) — Low

```
354 {
355 s.setMessage("You are not authorized to perform this function");
356 System.out.println("Authorization failure");
357 setCurrentAction(s, ERROR_ACTION);
358 ue2.printStackTrace();
359 }
```

FindProfile.java, line 83 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: FindProfile.java:83

```
80 }
81 catch (UnauthenticatedException ue1)
82 {
83 System.out.println("Internal server error");
84 ue1.printStackTrace();
85 }
86 catch (UnauthorizedException ue2)
```

FindProfile.java, line 88 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: FindProfile.java:88

```
85 }
86 catch (UnauthorizedException ue2)
87 {
88 System.out.println("Internal server error");
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 88 (Poor Logging Practice: Use of a System Output Stream) — Low

```
89 ue2.printStackTrace();
90 }
91 }
```

RoleBasedAccessControl.java, line 337 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:337

```
334 }
335 catch (ParameterNotFoundException pnfe)
336 {
337 System.out.println("Missing parameter");
338 pnfe.printStackTrace();
339 setCurrentAction(s, ERROR_ACTION);
340 }
```

RoleBasedAccessControl.java, line 457 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:457

```
454 catch (Exception e)
455 {
456 // All other errors send the user to the generic error page
457 System.out.println("handleRequest() error");
458 e.printStackTrace();
459 setCurrentAction(s, ERROR_ACTION);
```

Poor Logging Practice: Use of a System Output Stream	Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 457 (Poor Logging Practice: Use of a System Output Stream)	Low

```
460 }
```

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 94 (Poor Logging Practice: Use of a System Output Stream)	Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: Login.java:94

```
91 }
92 catch (UnauthenticatedException ue1)
93 {
94 System.out.println("Internal server error");
95 ue1.printStackTrace();
96 }
97 catch (UnauthorizedException ue2)
```

Login.java, line 99 (Poor Logging Practice: Use of a System Output Stream)	Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: Login.java:99

```
96 }
97 catch (UnauthorizedException ue2)
98 {
99 System.out.println("Internal server error");
100 ue2.printStackTrace();
101 }
102 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 354 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SQLInjection.java:354

```
351 catch (UnauthenticatedException ue)
352 {
353 s.setMessage("Login failed");
354 System.out.println("Authentication failure");
355 ue.printStackTrace();
356 }
357 catch (UnauthorizedException ue2)
```

SQLInjection.java, line 347 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SQLInjection.java:347

```
344 }
345 catch (ValidationException ve)
346 {
347 System.out.println("Validation failed");
348 ve.printStackTrace();
349 setCurrentAction(s, ERROR_ACTION);
350 }
```

SQLInjection.java, line 366 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 366 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SQLInjection.java:366

```
363 catch (Exception e)
364 {
365 // All other errors send the user to the generic error page
366 System.out.println("handleRequest() error");
367 e.printStackTrace();
368 setCurrentAction(s, ERROR_ACTION);
369 }
```

SQLInjection.java, line 341 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SQLInjection.java:341

```
338 }
339 catch (ParameterNotFoundException pnfe)
340 {
341 System.out.println("Missing parameter");
342 pnfe.printStackTrace();
343 setCurrentAction(s, ERROR_ACTION);
344 }
```

SQLInjection.java, line 360 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 360 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: handleRequest()
File: SQLInjection.java:360

```
357 catch (UnauthorizedException ue2)
358 {
359 s.setMessage("You are not authorized to perform this function");
360 System.out.println("Authorization failure");
361 ue2.printStackTrace();
362 }
363 catch (Exception e)
```

Package: org.owasp.webgoat.lessons.admin

RefreshDBScreen.java, line 163 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: refreshDB()
File: RefreshDBScreen.java:163

```
160
161 CreateDB db = new CreateDB();
162 db.makeDB(connection);
163 System.out.println("Successfully refreshed the database.");
164 }
165 catch (Exception e)
166 {
```

Package: org.owasp.webgoat.session

CreateDB.java, line 714 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 714 (Poor Logging Practice: Use of a System Output Stream) — Low

Enclosing Method: createOwnershipTable()
File: CreateDB.java:714

```
711 }
712 catch (SQLException e)
713 {
714 System.out.println("Error: unable to drop ownership");
715 }
716
717 try
```

CreateDB.java, line 254 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createUserAdminTable()
File: CreateDB.java:254

```
251 }
252 catch (SQLException e)
253 {
254 System.out.println("Error creating user admin database");
255 e.printStackTrace();
256 }
257
```

CreateDB.java, line 549 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createRolesTable()
File: CreateDB.java:549

```
546 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 549 (Poor Logging Practice: Use of a System Output Stream) — Low

```
547 catch (SQLException e)
548 {
549 System.out.println("Error: unable to drop roles");
550 }
551
552 try
```

CreateDB.java, line 120 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: CreateDB.java:120

```
117 //boolean allowed = answer_results.next();
118
119 if (allowed)
120 System.out.println("User is allowed");
121 else
122 System.out.println("User is NOT allowed");
123 }
```

CreateDB.java, line 202 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createProductTable()
File: CreateDB.java:202

```
199 }
200 catch (SQLException e)
201 {
202 System.out.println("Error creating product database");
203 e.printStackTrace();
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 202 (Poor Logging Practice: Use of a System Output Stream) — Low

```
204 }
205
```

WebSession.java, line 1089 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: update()
File: WebSession.java:1089

```
1086 isHackedAdmin = myParser.getBooleanParameter( ADMIN, isAdmin );
1087 if ( isHackedAdmin )
1088 {
1089 System.out.println("Hacked admin");
1090 hasHackedHackableAdmin = true;
1091 }
1092 isColor = myParser.getBooleanParameter( COLOR, isColor );
```

WebSession.java, line 439 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:439

```
436 try
437 {
438 String path = context.getRealPath( "/database" ).replace( '\\', '/' );
439 System.out.println( "PATH: " + path );
440 String realConnectionString = databaseConnectionString.replaceAll( "PATH", path );
441 System.out.println( "Database Connection String: " + realConnectionString );
442
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 95 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:95

```
92 if (os.toLowerCase().indexOf("window") != -1)
93 {
94 dbName = dbName.concat("webgoat.mdb");
95 System.out.println("DBName: " + dbName);
96 Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
97 return DriverManager
98 .getConnection("jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ="
```

CreateDB.java, line 77 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: CreateDB.java:77

```
74 }
75 catch (Exception e)
76 {
77 System.out.println("Driver Manager failed!");
78 e.printStackTrace();
79 }
80
```

CreateDB.java, line 96 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 96 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: CreateDB.java:96

```
93 int employeeId = answer_results.getInt("userid");
94 String firstName = answer_results.getString("first_name");
95 String lastName = answer_results.getString("last_name");
96 System.out.println("Query 1 Results: " + firstName + " " + lastName
97 + " " + employeeId);
98 }
99 catch (SQLException sqle)
```

CreateDB.java, line 240 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createUserAdminTable()
File: CreateDB.java:240

```
237 }
238 catch (SQLException e)
239 {
240 System.out.println("Error dropping user admin database");
241 }
242
243 // Create the new table
```

WebSession.java, line 447 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

WebSession.java, line 447 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:447

```
444 }
445 catch ( Exception e )
446 {
447 System.out.println( "Couldn't open database: check web.xml database parameters" );
448 e.printStackTrace();
449 }
450
```

CreateDB.java, line 867 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: makeDB()
File: CreateDB.java:867

```
864 createAuthTable(connection);
865 createOwnershipTable(connection);
866 createWeatherDataTable(connection);
867 System.out.println("Success: creating tables.");
868 }
869 }
870
```

CreateDB.java, line 405 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createWeatherDataTable()
File: CreateDB.java:405

```
402 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 405 (Poor Logging Practice: Use of a System Output Stream) — Low

```
403 catch (SQLException e)
404 {
405 System.out.println("Error creating weather database");
406 e.printStackTrace();
407 }
408
```

Course.java, line 71 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: Course()
File: Course.java:71

```
68 }
69 catch (IOException e)
70 {
71 System.out.println("Error loading WebGoat properties");
72 e.printStackTrace();
73 }
74 }
```

CreateDB.java, line 164 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createMessageTable()
File: CreateDB.java:164

```
161 }
162 catch (SQLException e)
163 {
164 System.out.println("Error creating message database");
165 e.printStackTrace();
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 164 (Poor Logging Practice: Use of a System Output Stream) — Low

```
166 }
167 }
```

CreateDB.java, line 150 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createMessageTable()
File: CreateDB.java:150

```
147 }
148 catch (SQLException e)
149 {
150 System.out.println("Error dropping message database");
151 }
152
153 // Create the new table
```

WebSession.java, line 283 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: WebSession()
File: WebSession.java:283

```
280
281 // FIXME: hack to save context for web service calls
282 DatabaseUtilities.servletContextRealPath = context.getRealPath("/");
283 System.out.println("Context Path: " + DatabaseUtilities.servletContextRealPath);
284 // FIXME: need to solve concurrency problem here -- make tables for this user
285 if ( !databaseBuilt )
286 {
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

WebgoatProperties.java, line 124 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: WebgoatProperties.java:124

```
121 System.out.println("Error loading properties");
122 e.printStackTrace();
123 }
124 System.out.println(properties.getProperty("CommandInjection.category"));
125 }
126
127 }
```

CreateDB.java, line 189 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createProductTable()
File: CreateDB.java:189

```
186 }
187 catch (SQLException e)
188 {
189 System.out.println("Error dropping product database");
190 }
191
192 // Create the new table
```

CreateDB.java, line 352 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 352 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createLoginTable()
File: CreateDB.java:352

```
349 }
350 catch (SQLException e)
351 {
352 System.out.println("Error dropping user_login table");
353 }
354
355 // Create the new table
```

WebgoatProperties.java, line 49 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: WebgoatProperties()
File: WebgoatProperties.java:49

```
46 catch (IOException e)
47 {
48 System.out
49 .println("Warning: Unable to open webgoat.properties file");
50 }
51 }
52
```

CreateDB.java, line 856 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 856 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: makeDB()
File: CreateDB.java:856

```
853 */
854 public void makeDB(Connection connection) throws SQLException
855 {
856 System.out.println("Successful connection to database");
857 createUserDataTable(connection);
858 createLoginTable(connection);
859 createUserAdminTable(connection);
```

CreateDB.java, line 472 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createEmployeeTable()
File: CreateDB.java:472

```
469 }
470 catch (SQLException e)
471 {
472 System.out.println("Error: unable to create employee table");
473 }
474
475 String insertData1 = "INSERT INTO employee VALUES (101, 'Larry', 'Stooge', '386-09-5451', 'larry','"
```

CreateDB.java, line 562 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createRolesTable()
File: CreateDB.java:562

```
559 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 562 (Poor Logging Practice: Use of a System Output Stream) — Low

```
560 catch (SQLException e)
561 {
562 System.out.println("Error: Unable to create role table");
563 }
564
565 String insertData1 = "INSERT INTO roles VALUES (101, 'employee')";
```

CreateDB.java, line 727 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createOwnershipTable()
File: CreateDB.java:727

```
724 }
725 catch (SQLException e)
726 {
727 System.out.println("Error: unable to create ownership table");
728 }
729
730 String inputData = "INSERT INTO ownership VALUES (112, 101)";
```

CreateDB.java, line 618 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createAuthTable()
File: CreateDB.java:618

```
615 }
616 catch (SQLException e)
617 {
618 System.out.println("Error: unable to create auth table");
619 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 618 (Poor Logging Practice: Use of a System Output Stream) — Low

```
620
621 String insertData1 = "INSERT INTO auth VALUES('employee', 'Logout')";
```

WebSession.java, line 801 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: openLessonSession()
File: WebSession.java:801

```
798
799 public void openLessonSession(AbstractLesson lesson)
800 {
801 System.out.println("Opening new lesson session for lesson " + lesson);
802 LessonSession lessonSession = new LessonSession();
803 lessonSessions.put(lesson, lessonSession);
804 }
```

CreateDB.java, line 604 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createAuthTable()
File: CreateDB.java:604

```
601 }
602 catch (SQLException e)
603 {
604 System.out.println("Error: unable to drop auth");
605 }
606
607 try
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

WebSession.java, line 988 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: update()
File: WebSession.java:988

```
985 // FIXME: doesn't work right -- no reauthentication
986 if ( myParser.getRawParameter( LOGOUT, null ) != null )
987 {
988 System.out.println( "Logout " + request.getUserPrincipal() );
989 eatCookies();
990 request.getSession().invalidate();
991 currentScreen = WELCOME;
```

CreateDB.java, line 291 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createUserDataTable()
File: CreateDB.java:291

```
288 }
289 catch (SQLException e)
290 {
291 System.out.println("Error dropping user database");
292 }
293
294 // Create the new table
```

ErrorScreen.java, line 159 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

ErrorScreen.java, line 159 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createContent()
File: ErrorScreen.java:159

```
156 */
157 protected Element createContent( WebSession s )
158 {
159 System.out.println( "errorscreen createContent Error:" + this.error + " message:" + this.message );
160
161 Element content;
162
```

LessonTracker.java, line 253 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: load()
File: LessonTracker.java:253

```
250 }
251 catch (Exception e)
252 {
253 System.out.println("Failed to load lesson state for " + screen);
254 e.printStackTrace();
255 }
256 finally
```

CreateDB.java, line 364 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 364 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: createLoginTable()
File: CreateDB.java:364

```
361 }
362 catch (SQLException e)
363 {
364 System.out.println("Error creating user database");
365 e.printStackTrace();
366 }
367
```

WebgoatProperties.java, line 121 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: WebgoatProperties.java:121

```
118 }
119 catch (IOException e)
120 {
121 System.out.println("Error loading properties");
122 e.printStackTrace();
123 }
124 System.out.println(properties.getProperty("CommandInjection.category"));
```

CreateDB.java, line 447 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createEmployeeTable()
File: CreateDB.java:447

```
444 }
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 447 (Poor Logging Practice: Use of a System Output Stream) — Low

```
445 catch (SQLException e)
446 {
447 System.out.println("Error: unable to drop employee table");
448 }
449
450 // Create Table
```

WebSession.java, line 1112 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: restartLesson()
File: WebSession.java:1112

```
1109
1110 private void restartLesson(int lessonId)
1111 {
1112 System.out.println("Restarting lesson: " + getLesson(lessonId));
1113 getCurrentLesson().getLessonTracker( this ).setStage(1);
1114 getCurrentLesson().getLessonTracker( this ).setCompleted(false);
1115 }
```

CreateDB.java, line 306 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createUserDataTable()
File: CreateDB.java:306

```
303 }
304 catch (SQLException e)
305 {
306 System.out.println("Error creating user database");
307 e.printStackTrace();
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 306 (Poor Logging Practice: Use of a System Output Stream) — Low

```
308 }
309
```

WebSession.java, line 441 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:441

```
438 String path = context.getRealPath( "/database" ).replace( '\\', '/' );
439 System.out.println( "PATH: " + path );
440 String realConnectionString = databaseConnectionString.replaceAll( "PATH", path );
441 System.out.println( "Database Connection String: " + realConnectionString );
442
443 return realConnectionString;
444 }
```

CreateDB.java, line 62 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: CreateDB.java:62

```
59 }
60 catch (Exception e)
61 {
62 System.out.println("Failed to load DB driver");
63 e.printStackTrace();
64 }
65
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 394 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: store()
File: LessonTracker.java:394

```
391 catch (Exception e)
392 {
393 // what do we want to do, I think nothing.
394 System.out.println("Warning User data for " + s.getUserName()
395 + " will not persist");
396 }
397 finally
```

CreateDB.java, line 122 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: CreateDB.java:122

```
119 if (allowed)
120 System.out.println("User is allowed");
121 else
122 System.out.println("User is NOT allowed");
123 }
124 catch (SQLException sqle)
125 {
```

CreateDB.java, line 391 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 391 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: createWeatherDataTable()
File: CreateDB.java:391

```
388 }
389 catch (SQLException e)
390 {
391 System.out.println("Error dropping weather database");
392 }
393
394 // Create the new table
```

Package: org.owasp.webgoat.util

Exec.java, line 506 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:506

```
503 {
504 ExecResults results;
505 String sep = System.getProperty("line.separator");
506 System.out.println("----------------------------------------" + sep
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509 System.out.println(results);
```

Exec.java, line 509 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.util

Exec.java, line 509 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:509

```
506 System.out.println("----------------------------------------" + sep
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509 System.out.println(results);
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
512 results = Exec.execSimple("netstat -r");
```

Exec.java, line 513 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:513

```
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
512 results = Exec.execSimple("netstat -r");
513 System.out.println(results);
514
515 if (results.outputContains("localhost:1031"))
516 {
```

Exec.java, line 524 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:524

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.util

Exec.java, line 524 (Poor Logging Practice: Use of a System Output Stream) — Low

```
521  + "TEST 3: execInput");
522  results = Exec.execInput("find \"cde\"",
523  "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524  System.out.println(results);
525  System.out.println("----------------------------------------" + sep
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
```

Exec.java, line 528 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:528

```
525  System.out.println("----------------------------------------" + sep
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528  System.out.println(results);
529  System.out.println("----------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
```

Exec.java, line 532 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:532

```
529  System.out.println("----------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.util

Exec.java, line 532 (Poor Logging Practice: Use of a System Output Stream) — Low

```
533 System.out.println("----------------------------------------" + sep
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
```

Exec.java, line 536 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:536

```
533 System.out.println("----------------------------------------" + sep
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536 System.out.println(results);
537 System.out.println("----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
```

Exec.java, line 540 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:540

```
537 System.out.println("----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540 System.out.println(results);
541 }
542 }
543
```

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.util

Exec.java, line 517 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:517

```
514
515 if (results.outputContains("localhost:1031"))
516 {
517 System.out.println("ERROR: listening on 1031");
518 }
519
520 System.out.println("----------------------------------------" + sep
```

Exec.java, line 529 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:529

```
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528 System.out.println(results);
529 System.out.println("----------------------------------------" + sep
530 + "TEST 5:execLazy");
531 results = Exec.execLazy("ping -t 127.0.0.1");
532 System.out.println(results);
```

Exec.java, line 537 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.util

Exec.java, line 537 (Poor Logging Practice: Use of a System Output Stream) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:537

```
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536 System.out.println(results);
537 System.out.println("----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540 System.out.println(results);
```

Exec.java, line 520 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:520

```
517 System.out.println("ERROR: listening on 1031");
518 }
519
520 System.out.println("----------------------------------------" + sep
521 + "TEST 3: execInput");
522 results = Exec.execInput("find \"cde\"",
523 "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
```

Exec.java, line 525 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Poor Logging Practice: Use of a System Output Stream — Low

Package: org.owasp.webgoat.util

Exec.java, line 525 (Poor Logging Practice: Use of a System Output Stream) — Low

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:525

```
522 results = Exec.execInput("find \"cde\"",
523 "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524 System.out.println(results);
525 System.out.println("----------------------------------------" + sep
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528 System.out.println(results);
```

Exec.java, line 510 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:510

```
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509 System.out.println(results);
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
512 results = Exec.execSimple("netstat -r");
513 System.out.println(results);
```

Exec.java, line 533 (Poor Logging Practice: Use of a System Output Stream) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: FunctionCall: println
Enclosing Method: main()
File: Exec.java:533

```
530 + "TEST 5:execLazy");
```

Poor Logging Practice: Use of a System Output Stream	Low

Package: org.owasp.webgoat.util

Exec.java, line 533 (Poor Logging Practice: Use of a System Output Stream)	Low

```
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
533  System.out.println("----------------------------------------" + sep
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536  System.out.println(results);
```

Poor Style: Confusing Naming (1 issue)

Abstract

The class contains a field and a method with the same name.

Explanation

It is confusing to have a member field and a method with the same name. It makes it easy for a programmer to accidentally call the method when attempting to access the field or vice versa.

Example 1:

```
public class Totaller {
  private int total;
  public int total() {
    ...
  }
}
```

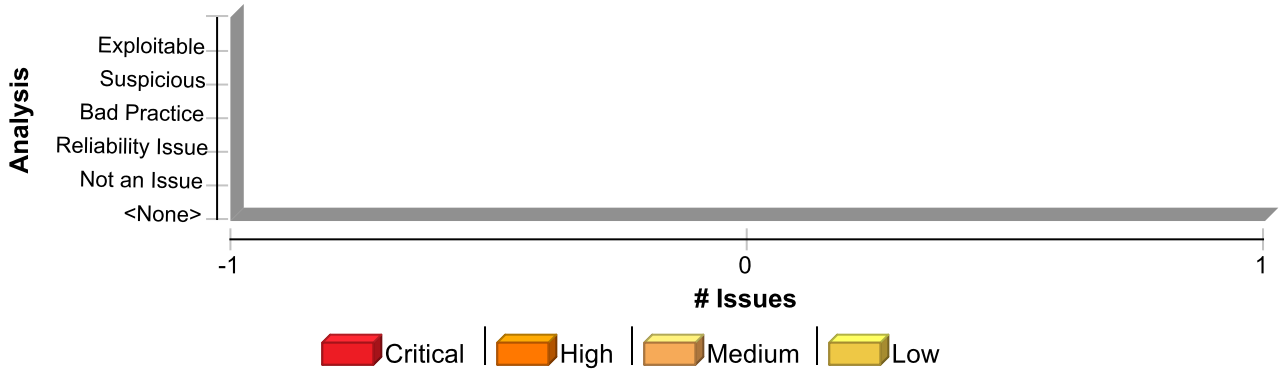
Recommendation

Rename either the method or the field. If the method returns the field, consider following the standard getter/setter naming convention.

Example 2: The code in Example 1 could be rewritten in the following way:

```
public class Totaller {
  private int total;
  public int getTotal() {
    ...
  }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Style: Confusing Naming	1	1	0	2
Total	**1**	**1**	**0**	**2**

Poor Style: Confusing Naming — Low

Package: org.owasp.webgoat.session

UserTracker.java, line 49 (Poor Style: Confusing Naming) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Field: instance
File: UserTracker.java:49

```
46 public class UserTracker
47 {
48
49  private static UserTracker instance;
50
51  // FIXME: persist this somehow!
52
```

Poor Style: Non-final Public Static Field (6 issues)

Abstract

Non-final public static fields can be changed by external classes.

Explanation

Typically, you do not want to provide external classes direct access to your object's member fields since a public field can be changed by any external class. Good object oriented designed uses encapsulation to prevent implementation details, such as member fields, from being exposed to other classes. Further, if the system assumes that this field cannot be changed, then malicious code might be able to adversely change the behavior of the system.

Example 1: In the following code, the field `ERROR_CODE` is declared as public and static, but not final:

```
public class MyClass
{
public static int ERROR_CODE = 100;
//...
}
```

In this case, malicious code might be able to change this error code and cause the program to behave in an unexpected manner. This category is from the Cigital Java Rulepack. http://www.cigital.com/

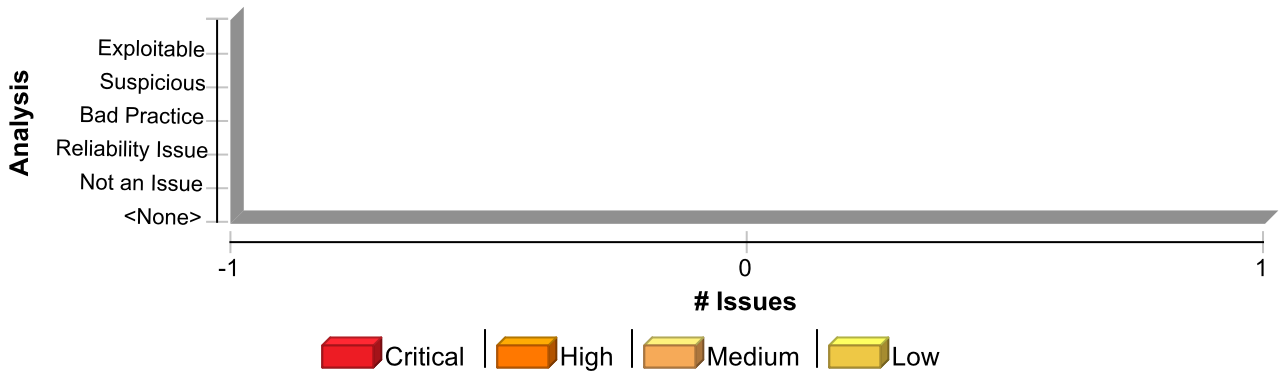
Recommendation

If you intend to expose a field as a constant value, the field should be declared as `public static final`, otherwise declare the field `private`.

Example 2:

```
public class MyClass
{
public static final int ERROR_CODE = 123;
//...
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Style: Non-final Public Static Field	6	6	0	12
Total	**6**	**6**	**0**	**12**

Poor Style: Non-final Public Static Field — Low

Package: org.owasp.webgoat

HammerHead.java, line 85 (Poor Style: Non-final Public Static Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Field: propertiesPath
File: HammerHead.java:85

```
82 /**
83 * Properties file path
84 */
85 public static String propertiesPath = null;
86
87
88 /**
```

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 74 (Poor Style: Non-final Public Static Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Field: connection
File: SoapRequest.java:74

```
71 */
72
73 //static boolean completed;
74 public static Connection connection = null;
75
76 public final static String firstName = "getFirstName";
77
```

Poor Style: Non-final Public Static Field — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 82 (Poor Style: Non-final Public Static Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Field: connection
File: WSDLScanning.java:82

```
79
80 static boolean beenRestartedYet = false;
81
82 public static Connection connection = null;
83
84 public final static String firstName = "getFirstName";
85
```

XMLInjection.java, line 61 (Poor Style: Non-final Public Static Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: Field: rewardsMap
File: XMLInjection.java:61

```
58
59 private final static String ACCOUNTID = "accountID";
60
61 public static HashMap rewardsMap = new HashMap();
62
63 private final static IMG MAC_LOGO = new IMG("images/logos/macadamian.gif").setAlt(
64 "Macadamian Technologies").setBorder(0).setHspace(0).setVspace(0);
```

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 50 (Poor Style: Non-final Public Static Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Poor Style: Non-final Public Static Field — Low

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 50 (Poor Style: Non-final Public Static Field) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Field: servletContextRealPath
File: DatabaseUtilities.java:50

```
47 public class DatabaseUtilities
48 {
49
50  public static String servletContextRealPath = null;
51
52
53  /**
```

Screen.java, line 54 (Poor Style: Non-final Public Static Field) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Field: MAIN_SIZE
File: Screen.java:54

```
51  /**
52  * Description of the Field
53  */
54  public static int MAIN_SIZE = 375;
55
56  //private Head head;
57  private Element content;
```

Poor Style: Redundant Initialization (4 issues)

Abstract

The variable's value is assigned but never used, making it a dead store.

Explanation

This variable's initial value is not used. After initialization, the variable is either assigned another value or goes out of scope.

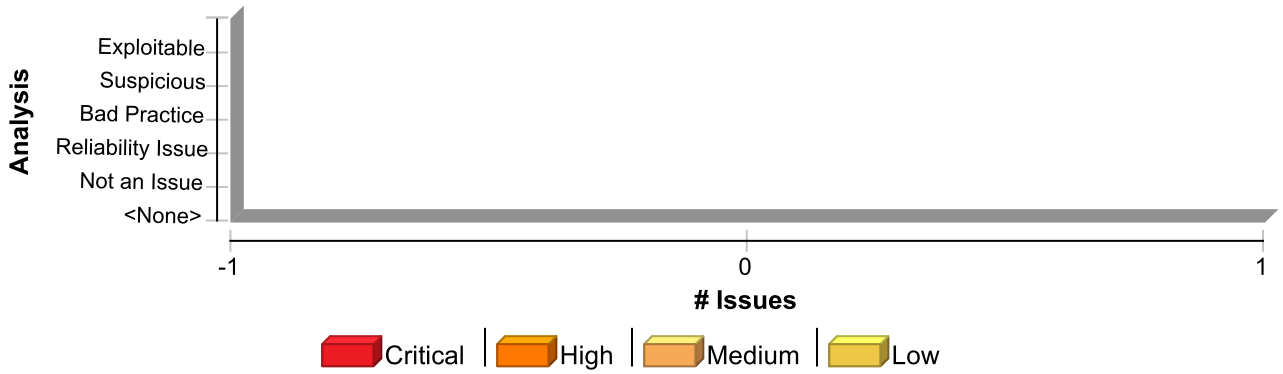
Example: The following code excerpt assigns to the variable `r` and then overwrites the value without using it.

```
int r = getNum();
r = getNewNum(buf);
```

Recommendation

Remove unnecessary assignments in order to make the code easier to understand and maintain.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Style: Redundant Initialization	4	4	0	8
Total	**4**	**4**	**0**	**8**

Poor Style: Redundant Initialization	**Low**
Package: org.owasp.webgoat.lessons	
ReflectedXSS.java, line 78 (Poor Style: Redundant Initialization)	**Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Poor Style: Redundant Initialization — Low

Package: org.owasp.webgoat.lessons

ReflectedXSS.java, line 78 (Poor Style: Redundant Initialization) — Low

Sink: VariableAccess: total
Enclosing Method: createContent()
File: ReflectedXSS.java:78

```
75 String param2 = HtmlEncoder.encode(s.getParser().getRawParameter(
76 "field2", "4128 3214 0002 1999"));
77 float quantity = 1.0f;
78 float total = 0.0f;
79 float runningTotal = 0.0f;
80
81 // test input field1
```

TraceXSS.java, line 78 (Poor Style: Redundant Initialization) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: VariableAccess: total
Enclosing Method: createContent()
File: TraceXSS.java:78

```
75 String param2 = HtmlEncoder.encode(s.getParser().getRawParameter(
76 "field2", "4128 3214 0002 1999"));
77 float quantity = 1.0f;
78 float total = 0.0f;
79 float runningTotal = 0.0f;
80
81 // test input field1
```

TraceXSS.java, line 77 (Poor Style: Redundant Initialization) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: VariableAccess: quantity
Enclosing Method: createContent()
File: TraceXSS.java:77

```
74 String param1 = s.getParser().getRawParameter("field1", "111");
```

Poor Style: Redundant Initialization — Low

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 77 (Poor Style: Redundant Initialization) — Low

```
75 String param2 = HtmlEncoder.encode(s.getParser().getRawParameter(
76 "field2", "4128 3214 0002 1999"));
77 float quantity = 1.0f;
78 float total = 0.0f;
79 float runningTotal = 0.0f;
80
```

ReflectedXSS.java, line 77 (Poor Style: Redundant Initialization) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: VariableAccess: quantity
Enclosing Method: createContent()
File: ReflectedXSS.java:77

```
74 String param1 = s.getParser().getRawParameter("field1", "111");
75 String param2 = HtmlEncoder.encode(s.getParser().getRawParameter(
76 "field2", "4128 3214 0002 1999"));
77 float quantity = 1.0f;
78 float total = 0.0f;
79 float runningTotal = 0.0f;
80
```

Poor Style: Value Never Read (4 issues)

Abstract

The variable's value is assigned but never used, making it a dead store.

Explanation

This variable's value is not used. After the assignment, the variable is either assigned another value or goes out of scope.

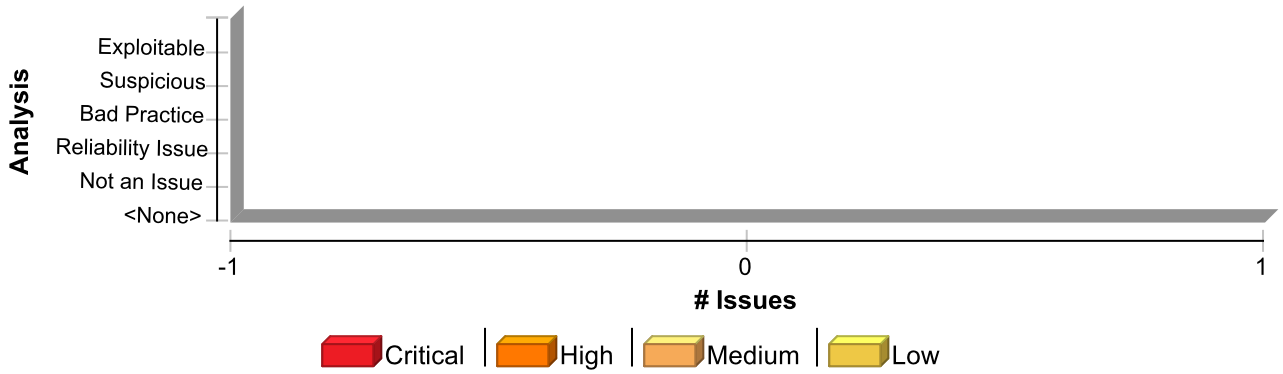
Example: The following code excerpt assigns to the variable `r` and then overwrites the value without using it.

```
r = getName();
r = getNewBuffer(buf);
```

Recommendation

Remove unnecessary assignments in order to make the code easier to understand and maintain.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Style: Value Never Read	4	4	0	8
Total	**4**	**4**	**0**	**8**

Poor Style: Value Never Read	**Low**
Package: org.owasp.webgoat.lessons	
CommandInjection.java, line 134 (Poor Style: Value Never Read)	**Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Poor Style: Value Never Read — Low

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 134 (Poor Style: Value Never Read) — Low

Sink: VariableAccess: safeDirName
Enclosing Method: createContent()
File: CommandInjection.java:134

```
131 if (upDirCount(helpFile) <= 3)
132 {
133 // FIXME: This value isn't used. What is the goal here?
134 safeDirName = s.getContext().getRealPath("/")
135 + helpFile;
136 illegalCommand = false;
137 }
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 114 (Poor Style: Value Never Read) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: VariableAccess: id
Enclosing Method: foundEmployee()
File: FindProfile.java:114

```
111 boolean found = false;
112 try
113 {
114 int id = getIntRequestAttribute(s, getLessonName() + "."
115 + RoleBasedAccessControl.EMPLOYEE_ID);
116 found = true;
117 }
```

UpdateProfile.java, line 176 (Poor Style: Value Never Read) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: VariableAccess: answer_results
Enclosing Method: changeEmployeeProfile()

Poor Style: Value Never Read — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 176 (Poor Style: Value Never Read) — Low

File: UpdateProfile.java:176

```
173 Statement answer_statement = WebSession.getConnection(s)
174 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
175 ResultSet.CONCUR_READ_ONLY);
176 ResultSet answer_results = answer_statement.executeQuery(query);
177 }
178 catch (SQLException sqle)
179 {
```

UpdateProfile.java, line 225 (Poor Style: Value Never Read) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: VariableAccess: answer_results
Enclosing Method: changeEmployeeProfile_BACKUP()
File: UpdateProfile.java:225

```
222 Statement answer_statement = WebSession.getConnection(s)
223 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
224 ResultSet.CONCUR_READ_ONLY);
225 ResultSet answer_results = answer_statement.executeQuery(query);
226 }
227 catch (SQLException sqle)
228 {
```

Portability Flaw: Locale Dependent Comparison (11 issues)

Abstract

Unexpected portability problems can be found when the locale is not specified.

Explanation

When comparing data that may be locale-dependent, an appropriate locale should be specified.

Example 1: The following example tries to perform validation to determine if user input includes a `<script>` tag.

```
  ...
  public String tagProcessor(String tag){
    if (tag.toUpperCase().equals("SCRIPT")){
      return null;
    }
    //does not contain SCRIPT tag, keep processing input
    ...
  }
  ...
```

The problem with the above code is that `java.lang.String.toUpperCase()` when used without a locale uses the rules of the default locale. Using the Turkish locale `"title".toUpperCase()` returns "T\u0130TLE", where "\u0130" is the "LATIN CAPITAL LETTER I WITH DOT ABOVE" character. This can lead to unexpected results, such as in Example 1 where this will prevent the word "script" from being caught by this validation, potentially leading to a Cross-Site Scripting vulnerability.

Recommendation

To prevent this from occurring, always make sure to either specify the default locale, or specify the locale with APIs that accept them such as `toUpperCase()`.

Example 2: The following specifies the locale manually as an argument to `toUpperCase()`.

```
import java.util.Locale;
  ...
  public String tagProcessor(String tag){
    if (tag.toUpperCase(Locale.ENGLISH).equals("SCRIPT")){
      return null;
    }
    //does not contain SCRIPT tag, keep processing input
    ...
  }
  ...
```

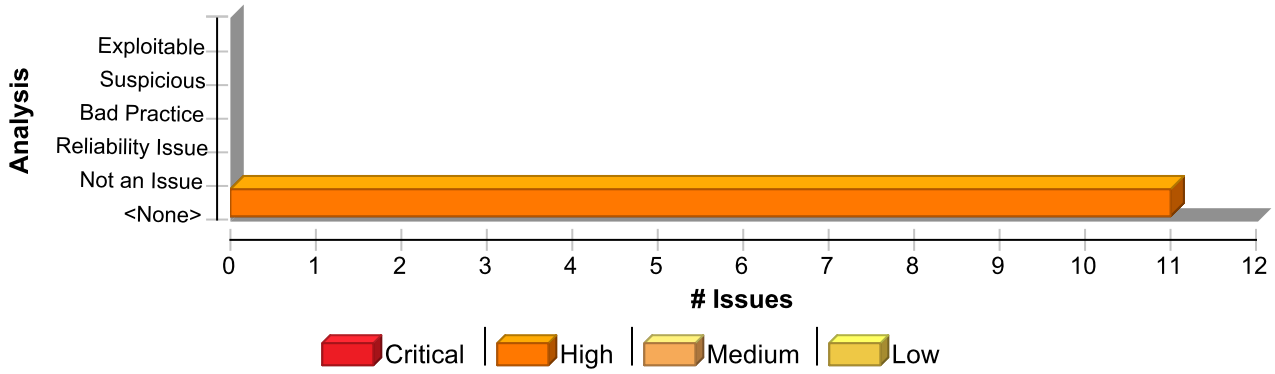
Example 3: The following uses the function `java.lang.String.equalsIgnoreCase()` API to prevent this issue.

```
  ...
  public String tagProcessor(String tag){
    if (tag.equalsIgnoreCase("SCRIPT")){
      return null;
    }
    //does not contain SCRIPT tag, keep processing input
```

```
  ...
}
...
```

This prevents the problem because `equalsIgnoreCase()` changes case similar to `Character.toLowerCase()` and `Character.toUpperCase()`. This involves creating temporary canonical forms of both strings using information from the `UnicodeData` file that is part of the Unicode Character Database maintained by the Unicode Consortium, and even though this may render them unreadable if they were to be read out, it makes comparison possible without being dependent upon locale.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Portability Flaw: Locale Dependent Comparison	11	11	0	22
Total	**11**	**11**	**0**	**22**

Portability Flaw: Locale Dependent Comparison — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 102 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:102

```
99  .substring(index, helpFileLen).trim().toLowerCase()
100 .equals("ipconfig")))
101 || (helpFile.substring(index, helpFileLen).trim()
102 .toLowerCase().equals("netstat -a #")
```

Portability Flaw: Locale Dependent Comparison — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 102 (Portability Flaw: Locale Dependent Comparison) — High

```
103  || helpFile.substring(index, helpFileLen)
104  .trim().toLowerCase().equals("dir #")
105  || helpFile.substring(index, helpFileLen)
```

CommandInjection.java, line 98 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:98

```
95  || helpFile.substring(index, helpFileLen).trim()
96  .toLowerCase().equals("ls")
97  || helpFile.substring(index, helpFileLen).trim()
98  .toLowerCase().equals("ifconfig") || helpFile
99  .substring(index, helpFileLen).trim().toLowerCase()
100 .equals("ipconfig")))
101 || (helpFile.substring(index, helpFileLen).trim()
```

CommandInjection.java, line 94 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:94

```
91  index, helpFileLen).trim().toLowerCase().equals(
92  "netstat -a")
93  || helpFile.substring(index, helpFileLen).trim()
94  .toLowerCase().equals("dir")
95  || helpFile.substring(index, helpFileLen).trim()
96  .toLowerCase().equals("ls")
```

Portability Flaw: Locale Dependent Comparison — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 94 (Portability Flaw: Locale Dependent Comparison) — High

```
97  || helpFile.substring(index, helpFileLen).trim()
```

CommandInjection.java, line 108 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:108

```
105  || helpFile.substring(index, helpFileLen)
106  .trim().toLowerCase().equals("ls #")
107  || helpFile.substring(index, helpFileLen)
108  .trim().toLowerCase().equals("ls -l #")
109  || helpFile.substring(index, helpFileLen)
110  .trim().toLowerCase().equals(
111  "ifconfig #") || helpFile
```

CommandInjection.java, line 100 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:100

```
97  || helpFile.substring(index, helpFileLen).trim()
98  .toLowerCase().equals("ifconfig") || helpFile
99  .substring(index, helpFileLen).trim().toLowerCase()
100  .equals("ipconfig")))
101  || (helpFile.substring(index, helpFileLen).trim()
102  .toLowerCase().equals("netstat -a #")
103  || helpFile.substring(index, helpFileLen)
```

Portability Flaw: Locale Dependent Comparison — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 110 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:110

```
107 || helpFile.substring(index, helpFileLen)
108 .trim().toLowerCase().equals("ls -l #")
109 || helpFile.substring(index, helpFileLen)
110 .trim().toLowerCase().equals(
111 "ifconfig #") || helpFile
112 .substring(index, helpFileLen).trim()
113 .toLowerCase().equals("ipconfig #")))
```

CommandInjection.java, line 104 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:104

```
101 || (helpFile.substring(index, helpFileLen).trim()
102 .toLowerCase().equals("netstat -a #")
103 || helpFile.substring(index, helpFileLen)
104 .trim().toLowerCase().equals("dir #")
105 || helpFile.substring(index, helpFileLen)
106 .trim().toLowerCase().equals("ls #")
107 || helpFile.substring(index, helpFileLen)
```

CommandInjection.java, line 113 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Portability Flaw: Locale Dependent Comparison — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 113 (Portability Flaw: Locale Dependent Comparison) — High

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:113

```
110 .trim().toLowerCase().equals(
111 "ifconfig #") || helpFile
112 .substring(index, helpFileLen).trim()
113 .toLowerCase().equals("ipconfig #")))
114 {
115 illegalCommand = false;
116 }
```

CommandInjection.java, line 96 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:96

```
93 || helpFile.substring(index, helpFileLen).trim()
94 .toLowerCase().equals("dir")
95 || helpFile.substring(index, helpFileLen).trim()
96 .toLowerCase().equals("ls")
97 || helpFile.substring(index, helpFileLen).trim()
98 .toLowerCase().equals("ifconfig") || helpFile
99 .substring(index, helpFileLen).trim().toLowerCase()
```

CommandInjection.java, line 91 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Portability Flaw: Locale Dependent Comparison — High

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 91 (Portability Flaw: Locale Dependent Comparison) — High

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:91

```
88  + helpFile.substring(index, helpFileLen).trim()
89  .toLowerCase() + "]");
90  if ((osName.indexOf("Windows") != -1 && (helpFile.substring(
91  index, helpFileLen).trim().toLowerCase().equals(
92  "netstat -a")
93  || helpFile.substring(index, helpFileLen).trim()
94  .toLowerCase().equals("dir")
```

CommandInjection.java, line 106 (Portability Flaw: Locale Dependent Comparison) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: helpFile.substring(index, helpFileLen).trim().toLowerCase().equals(...) : Comparison without checking locale
Enclosing Method: createContent()
File: CommandInjection.java:106

```
103  || helpFile.substring(index, helpFileLen)
104  .trim().toLowerCase().equals("dir #")
105  || helpFile.substring(index, helpFileLen)
106  .trim().toLowerCase().equals("ls #")
107  || helpFile.substring(index, helpFileLen)
108  .trim().toLowerCase().equals("ls -l #")
109  || helpFile.substring(index, helpFileLen)
```

Privacy Violation (78 issues)

Abstract

Mishandling private information, such as customer passwords or social security numbers, can compromise user privacy and is often illegal.

Explanation

Privacy violations occur when:

1. Private user information enters the program.

2. The data is written to an external location, such as the console, file system, or network.

Example 1: The following code contains a logging statement that tracks the contents of records added to a database by storing them in a log file. Among other values that are stored, the `getPassword()` function returns the user-supplied plain text password associated with the account.

```
pass = getPassword();
...
dbmsLog.println(id+":"+pass+":"+type+":"+tstamp);
```

The code in the example above logs a plain text password to the file system. Although many developers trust the file system as a safe storage location for data, it should not be trusted implicitly, particularly when privacy is a concern.

Privacy is one of the biggest concerns in the mobile world for a couple of reasons. One of them is a much higher chance of device loss. The other has to do with inter-process communication between mobile applications. The essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which is why application authors need to be careful about what information they include in messages addressed to other applications running on the device. Sensitive information should never be part of inter-process communication between mobile applications.

Example 2: The code below reads username and password for a given site from an Android WebView store and broadcasts them to all the registered receivers.

```
...
webview.setWebViewClient(new WebViewClient() {
  public void onReceivedHttpAuthRequest(WebView view,
        HttpAuthHandler handler, String host, String realm) {
    String[] credentials = view.getHttpAuthUsernamePassword(host, realm);
    String username = credentials[0];
    String password = credentials[1];
    Intent i = new Intent();
    i.setAction("SEND_CREDENTIALS");
    i.putExtra("username", username);
    i.putExtra("password", password);
    view.getContext().sendBroadcast(i);
  }
});
...
```

There are several problems with this example. First of all, by default, WebView credentials are stored in plain text and

are not hashed. So if a user has a rooted device (or uses an emulator), she is able to read stored passwords for given sites. Second, plain text credentials are broadcast to all the registered receivers, which means that any receiver registered to listen to intents with the `SEND_CREDENTIALS` action will receive the message. The broadcast is not even protected with a permission to limit the number of recipients, even though in this case, we do not recommend using permissions as a fix.

Private data can enter a program in a variety of ways:

- Directly from the user in the form of a password or personal information

- Accessed from a database or other data store by the application

- Indirectly from a partner or other third party

Typically, in the context of the mobile world, this private information would include (along with passwords, SSNs and other general personal information):

- Location

- Cell phone number

- Serial numbers and device IDs

- Network Operator information

- Voicemail information

Sometimes data that is not labeled as private can have a privacy implication in a different context. For example, student identification numbers are usually not considered private because there is no explicit and publicly-available mapping to an individual student's personal information. However, if a school generates identification numbers based on student social security numbers, then the identification numbers should be considered private.

Security and privacy concerns often seem to compete with each other. From a security perspective, you should record all important operations so that any anomalous activity can later be identified. However, when private data is involved, this practice can create risk.

Although there are many ways in which private data can be handled unsafely, a common risk stems from misplaced trust. Programmers often trust the operating environment in which a program runs, and therefore believe that it is acceptable to store private information on the file system, in the registry, or in other locally-controlled resources. However, even if access to certain resources is restricted, this does not guarantee that the individuals who do have access can be trusted. For example, in 2004, an unscrupulous employee at AOL sold approximately 92 million private customer e-mail addresses to a spammer marketing an offshore gambling web site [1].

In response to such high-profile exploits, the collection and management of private data is becoming increasingly regulated. Depending on its location, the type of business it conducts, and the nature of any private data it handles, an organization may be required to comply with one or more of the following federal and state regulations:

- Safe Harbor Privacy Framework [3]

- Gramm-Leach Bliley Act (GLBA) [4]

- Health Insurance Portability and Accountability Act (HIPAA) [5]

- California SB-1386 [6]

Despite these regulations, privacy violations continue to occur with alarming frequency.

Recommendation

When security and privacy demands clash, privacy should usually be given the higher priority. To accomplish this and still maintain required security information, cleanse any private information before it exits the program.

To enforce good privacy management, develop and strictly adhere to internal privacy guidelines. The guidelines should specifically describe how an application should handle private data. If your organization is regulated by federal or state law, ensure that your privacy guidelines are sufficiently strenuous to meet the legal requirements. Even if your organization is not regulated, you must protect private information or risk losing customer confidence.

The best policy with respect to private data is to minimize its exposure. Applications, processes, and employees should not be granted access to any private data unless the access is required for the tasks that they are to perform. Just as the principle of least privilege dictates that no operation should be performed with more than the necessary privileges, access to private data should be restricted to the smallest possible group.

For mobile applications, make sure they never communicate any sensitive data to other applications running on the device. When private data needs to be stored, it should always be encrypted. For Android, as well as any other platform that uses SQLite database, SQLCipher is a good alternative. SQLCipher is an extension to the SQLite database that provides transparent 256-bit AES encryption of database files. Thus, credentials can be stored in an encrypted database.

Example 3: The code below demonstrates how to integrate SQLCipher into an Android application after downloading the necessary binaries, and store credentials into the database file.

```
import net.sqlcipher.database.SQLiteDatabase;
...
  SQLiteDatabase.loadLibs(this);
  File dbFile = getDatabasePath("credentials.db");
  dbFile.mkdirs();
  dbFile.delete();
  SQLiteDatabase db = SQLiteDatabase.openOrCreateDatabase(dbFile,
"credentials", null);
  db.execSQL("create table credentials(u, p)");
  db.execSQL("insert into credentials(u, p) values(?, ?)", new Object[]
{username, password});
...
```

Note that references to `android.database.sqlite.SQLiteDatabase` are substituted with those of `net.sqlcipher.database.SQLiteDatabase`.

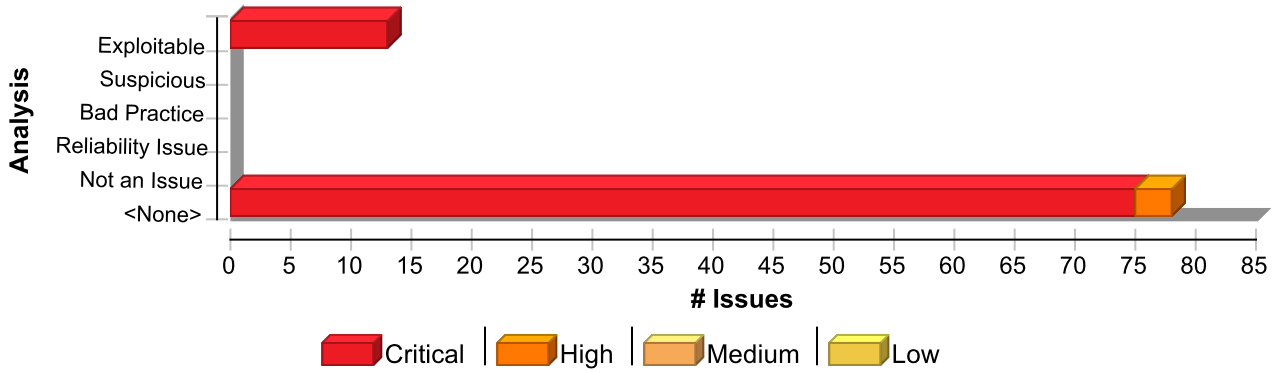
To enable encryption on the WebView store, WebKit has to be re-compiled with the `sqlcipher.so` library.

Example 4: The code below reads username and password for a given site from an Android WebView store and instead of broadcasting them to all the registered receivers, it only broadcasts internally so that the broadcast can only be seen by other parts of the same app.

```
...
webview.setWebViewClient(new WebViewClient() {
  public void onReceivedHttpAuthRequest(WebView view,
        HttpAuthHandler handler, String host, String realm) {
    String[] credentials = view.getHttpAuthUsernamePassword(host, realm);
    String username = credentials[0];
    String password = credentials[1];
    Intent i = new Intent();
    i.setAction("SEND_CREDENTIALS");
    i.putExtra("username", username);
```

```
    i.putExtra("password", password);
    LocalBroadcastManager.getInstance(view.getContext()).sendBroadcast(i);
  }
});
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Privacy Violation	78	78	0	156
Total	**78**	**78**	**0**	**156**

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 58 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Indeterminate (Below Exploitable threshold)
AA_Training	Include

Source Details

Source: Read this.ssn
From: org.owasp.webgoat.session.Employee.getSsn
File: JavaSource/org/owasp/webgoat/session/Employee.java:204

```
201
202  public String getSsn()
203  {
204  return ssn;
205  }
206
```

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 58 (Privacy Violation) — Critical

207

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:58
Taint Flags: PRIVATE

```
55 SSN:
56 </TD>
57 <TD>
58 <%=employee.getSsn()%>
59 </TD>
60 <TD>
61 Salary:
```

ViewProfile.jsp, line 71 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getCcn()
From: /lessons/CrossSiteScripting.ViewProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/ViewProfile.jsp:71

```
68 Credit Card:
69 </TD>
70 <TD>
71 <%=employee.getCcn()%>
72 </TD>
73 <TD>
74 Credit Card Limit:
```

Sink Details

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 71 (Privacy Violation) — Critical

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:71
Taint Flags: PRIVATE

```
68 Credit Card:
69 </TD>
70 <TD>
71 <%=employee.getCcn()%>
72 </TD>
73 <TD>
74 Credit Card Limit:
```

EditProfile.jsp, line 73 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.CrossSiteScripting.CrossSiteScripting.CCN_LI
MIT
From: /lessons/CrossSiteScripting.EditProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/EditProfile.jsp:73

```
70 Credit Card Limit:
71 </TD>
72 <TD>
73 <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN_LIMIT%>" type="text" value="<
%=employee.getCcnLimit()%>"/>
74 </TD>
75 </TR>
76 <TR><TD>
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:73
Taint Flags: PRIVATE

```
70 Credit Card Limit:
71 </TD>
72 <TD>
```

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 73 (Privacy Violation) — Critical

```
73 <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN_LIMIT%>" type="text" value="<%=employee.getCcnLimit()%>"/>
74 </TD>
75 </TR>
76 <TR><TD>
```

ViewProfile.jsp, line 58 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getSsn()
From: /lessons/CrossSiteScripting.ViewProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/ViewProfile.jsp:58

```
55 SSN:
56 </TD>
57 <TD>
58 <%=employee.getSsn()%>
59 </TD>
60 <TD>
61 Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:58
Taint Flags: PRIVATE

```
55 SSN:
56 </TD>
57 <TD>
58 <%=employee.getSsn()%>
59 </TD>
60 <TD>
```

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

ViewProfile.jsp, line 58 (Privacy Violation) — Critical

```
61 Salary:
```

ViewProfile.jsp, line 71 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ccn
From: org.owasp.webgoat.session.Employee.getCcn
File: JavaSource/org/owasp/webgoat/session/Employee.java:132

```
129
130  public String getCcn()
131  {
132  return ccn;
133  }
134
135
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:71
Taint Flags: PRIVATE

```
68 Credit Card:
69 </TD>
70 <TD>
71 <%=employee.getCcn()%>
72 </TD>
73 <TD>
74 Credit Card Limit:
```

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.CrossSiteScripting.CrossSiteScripting.SSN
From: /lessons/CrossSiteScripting.EditProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/EditProfile.jsp:54

```
51  SSN:
52  </TD>
53  <TD>
54  <input class="lesson_text_db" name="<%=CrossSiteScripting.SSN%>" type="text" value="<
%=employee.getSsn()%>"/>
55  </TD>
56  <TD>
57  Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:54
Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
54 <input class="lesson_text_db" name="<%=CrossSiteScripting.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
55 </TD>
56 <TD>
57 Salary:
```

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ccn
From: org.owasp.webgoat.session.Employee.getCcn
File: JavaSource/org/owasp/webgoat/session/Employee.java:132

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 67 (Privacy Violation) — Critical

```
129
130  public String getCcn()
131  {
132  return ccn;
133  }
134
135
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:67
Taint Flags: PRIVATE

```
64  Credit Card:
65  </TD>
66  <TD>
67  <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68  </TD>
69  <TD>
70  Credit Card Limit:
```

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getSsn()
From: /lessons/CrossSiteScripting.EditProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/EditProfile.jsp:54

```
51  SSN:
52  </TD>
53  <TD>
```

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 54 (Privacy Violation) — Critical

```
54  <input class="lesson_text_db" name="<%=CrossSiteScripting.SSN%>" type="text" value="<
%=employee.getSsn()%>"/>
55  </TD>
56  <TD>
57  Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:54
Taint Flags: PRIVATE

```
51  SSN:
52  </TD>
53  <TD>
54  <input class="lesson_text_db" name="<%=CrossSiteScripting.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
55  </TD>
56  <TD>
57  Salary:
```

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getCcn()
From: /lessons/CrossSiteScripting.EditProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/EditProfile.jsp:67

```
64  Credit Card:
65  </TD>
66  <TD>
67  <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN%>" type="text" value="<
%=employee.getCcn()%>"/>
```

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 67 (Privacy Violation) — Critical

```
68  </TD>
69  <TD>
70  Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:67
Taint Flags: PRIVATE

```
64  Credit Card:
65  </TD>
66  <TD>
67  <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68  </TD>
69  <TD>
70  Credit Card Limit:
```

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.CrossSiteScripting.CrossSiteScripting.CCN
From: /lessons/CrossSiteScripting.EditProfile.jsp._jspService
File: WebContent/lessons/CrossSiteScripting/EditProfile.jsp:67

```
64  Credit Card:
65  </TD>
66  <TD>
67  <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN%>" type="text" value="<
%=employee.getCcn()%>"/>
68  </TD>
69  <TD>
70  Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 67 (Privacy Violation) — Critical

File: EditProfile.jsp:67
Taint Flags: PRIVATE

```
64 Credit Card:
65 </TD>
66 <TD>
67 <input class="lesson_text_db" name="<%=CrossSiteScripting.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ssn
From: org.owasp.webgoat.session.Employee.getSsn
File: JavaSource/org/owasp/webgoat/session/Employee.java:204

```
201
202  public String getSsn()
203  {
204  return ssn;
205  }
206
207
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:54
Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
54 <input class="lesson_text_db" name="<%=CrossSiteScripting.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
55 </TD>
56 <TD>
```

Privacy Violation — Critical

Package: /lessons/CrossSiteScripting

EditProfile.jsp, line 54 (Privacy Violation) — Critical

```
57 Salary:
```

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 72 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.CCN_LIMIT
From: /lessons/RoleBasedAccessControl.EditProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/EditProfile.jsp:72

```
69 Credit Card Limit:
70 </TD>
71 <TD>
72 <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN_LIMIT%>" type="text" value="<
%=employee.getCcnLimit()%>"/>
73 </TD>
74 </TR>
75 <TR><TD>
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:72
Taint Flags: PRIVATE

```
69 Credit Card Limit:
70 </TD>
71 <TD>
72 <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN_LIMIT%>" type="text" value="<%=employee.getCcnLimit()
%>"/>
73 </TD>
74 </TR>
75 <TR><TD>
```

EditProfile.jsp, line 53 (Privacy Violation) — Critical

Issue Details

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 53 (Privacy Violation) — Critical

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getSsn()
From: /lessons/RoleBasedAccessControl.EditProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/EditProfile.jsp:53

```
50  SSN:
51  </TD>
52  <TD>
53  <input class="lesson_text_db" name="<%=RoleBasedAccessControl.SSN%>" type="text" value="<
%=employee.getSsn()%>"/>
54  </TD>
55  <TD>
56  Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:53
Taint Flags: PRIVATE

```
50 SSN:
51 </TD>
52 <TD>
53 <input class="lesson_text_db" name="<%=RoleBasedAccessControl.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
54 </TD>
55 <TD>
56 Salary:
```

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getSsn()
From: /lessons/RoleBasedAccessControl.ViewProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/ViewProfile.jsp:54

```
51 SSN:
52 </TD>
53 <TD>
54 <span class="lesson_text_db"><%=employee.getSsn()%></span>
55 </TD>
56 <TD>
57 Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:54
Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
54 <span class="lesson_text_db"><%=employee.getSsn()%></span>
55 </TD>
56 <TD>
57 Salary:
```

EditProfile.jsp, line 66 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction	Indeterminate (Below Exploitable threshold)

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 66 (Privacy Violation) — Critical

Source Details

Source: Read this.ccn
From: org.owasp.webgoat.session.Employee.getCcn
File: JavaSource/org/owasp/webgoat/session/Employee.java:132

```
129
130  public String getCcn()
131  {
132  return ccn;
133  }
134
135
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:66
Taint Flags: PRIVATE

```
63  Credit Card:
64  </TD>
65  <TD>
66  <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
67  </TD>
68  <TD>
69  Credit Card Limit:
```

ViewProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ccn
From: org.owasp.webgoat.session.Employee.getCcn
File: JavaSource/org/owasp/webgoat/session/Employee.java:132

```
129
130  public String getCcn()
131  {
132  return ccn;
```

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

ViewProfile.jsp, line 67 (Privacy Violation) — Critical

```
133  }
134
135
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:67
Taint Flags: PRIVATE

```
64  Credit Card:
65  </TD>
66  <TD>
67  <span class="lesson_text_db"><%=employee.getCcn()%></span>
68  </TD>
69  <TD>
70  Credit Card Limit:
```

EditProfile.jsp, line 66 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getCcn()
From: /lessons/RoleBasedAccessControl.EditProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/EditProfile.jsp:66

```
63  Credit Card:
64  </TD>
65  <TD>
66  <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN%>" type="text" value="<
%=employee.getCcn()%>"/>
67  </TD>
68  <TD>
```

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 66 (Privacy Violation) — Critical

```
69  Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:66
Taint Flags: PRIVATE

```
63  Credit Card:
64  </TD>
65  <TD>
66  <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
67  </TD>
68  <TD>
69  Credit Card Limit:
```

EditProfile.jsp, line 53 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ssn
From: org.owasp.webgoat.session.Employee.getSsn
File: JavaSource/org/owasp/webgoat/session/Employee.java:204

```
201
202  public String getSsn()
203  {
204  return ssn;
205  }
206
207
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:53
Taint Flags: PRIVATE

```
50  SSN:
```

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 53 (Privacy Violation) — Critical

```
51 </TD>
52 <TD>
53 <input class="lesson_text_db" name="<%=RoleBasedAccessControl.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
54 </TD>
55 <TD>
56 Salary:
```

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ssn
From: org.owasp.webgoat.session.Employee.getSsn
File: JavaSource/org/owasp/webgoat/session/Employee.java:204

```
201
202  public String getSsn()
203  {
204  return ssn;
205  }
206
207
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:54
Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
54 <span class="lesson_text_db"><%=employee.getSsn()%></span>
55 </TD>
56 <TD>
57 Salary:
```

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

ViewProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getCcn()
From: /lessons/RoleBasedAccessControl.ViewProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/ViewProfile.jsp:67

```
64 Credit Card:
65 </TD>
66 <TD>
67 <span class="lesson_text_db"><%=employee.getCcn()%></span>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:67
Taint Flags: PRIVATE

```
64 Credit Card:
65 </TD>
66 <TD>
67 <span class="lesson_text_db"><%=employee.getCcn()%></span>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

EditProfile.jsp, line 53 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 53 (Privacy Violation) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessControl.SSN
From: /lessons/RoleBasedAccessControl.EditProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/EditProfile.jsp:53

```
50 SSN:
51 </TD>
52 <TD>
53 <input class="lesson_text_db" name="<%=RoleBasedAccessControl.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
54 </TD>
55 <TD>
56 Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:53
Taint Flags: PRIVATE

```
50 SSN:
51 </TD>
52 <TD>
53 <input class="lesson_text_db" name="<%=RoleBasedAccessControl.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
54 </TD>
55 <TD>
56 Salary:
```

EditProfile.jsp, line 66 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessControl.CCN

Privacy Violation — Critical

Package: /lessons/RoleBasedAccessControl

EditProfile.jsp, line 66 (Privacy Violation) — Critical

From: /lessons/RoleBasedAccessControl.EditProfile.jsp._jspService
File: WebContent/lessons/RoleBasedAccessControl/EditProfile.jsp:66

```
63  Credit Card:
64  </TD>
65  <TD>
66  <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN%>" type="text" value="<
%=employee.getCcn()%>"/>
67  </TD>
68  <TD>
69  Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:66
Taint Flags: PRIVATE

```
63  Credit Card:
64  </TD>
65  <TD>
66  <input class="lesson_text_db" name="<%=RoleBasedAccessControl.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
67  </TD>
68  <TD>
69  Credit Card Limit:
```

Package: /lessons/SQLInjection

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ssn
From: org.owasp.webgoat.session.Employee.getSsn
File: JavaSource/org/owasp/webgoat/session/Employee.java:204

```
201
202  public String getSsn()
203  {
```

Privacy Violation — Critical

Package: /lessons/SQLInjection

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

```
204  return ssn;
205  }
206
207
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:54
Taint Flags: PRIVATE

```
51  SSN:
52  </TD>
53  <TD>
54  <%=employee.getSsn()%>
55  </TD>
56  <TD>
57  Salary:
```

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getSsn()
From: /lessons/SQLInjection.EditProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/EditProfile.jsp:54

```
51  SSN:
52  </TD>
53  <TD>
54  <input class="lesson_text_db" name="<%=SQLInjection.SSN%>" type="text" value="<%=employee.getSsn()
%>"/>
55  </TD>
```

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 54 (Privacy Violation) — Critical

```
56  <TD>
57  Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:54
Taint Flags: PRIVATE

```
51  SSN:
52  </TD>
53  <TD>
54  <input class="lesson_text_db" name="<%=SQLInjection.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
55  </TD>
56  <TD>
57  Salary:
```

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.SSN
From: /lessons/SQLInjection.EditProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/EditProfile.jsp:54

```
51  SSN:
52  </TD>
53  <TD>
54  <input class="lesson_text_db" name="<%=SQLInjection.SSN%>" type="text" value="<%=employee.getSsn()
%>"/>
55  </TD>
56  <TD>
57  Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:54

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
54 <input class="lesson_text_db" name="<%=SQLInjection.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
55 </TD>
56 <TD>
57 Salary:
```

EditProfile.jsp, line 73 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.CCN_LIMIT
From: /lessons/SQLInjection.EditProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/EditProfile.jsp:73

```
70 Credit Card Limit:
71 </TD>
72 <TD>
73 <input class="lesson_text_db" name="<%=SQLInjection.CCN_LIMIT%>" type="text" value="<
%=employee.getCcnLimit()%>"/>
74 </TD>
75 </TR>
76 <TR><TD>
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:73
Taint Flags: PRIVATE

```
70 Credit Card Limit:
71 </TD>
72 <TD>
73 <input class="lesson_text_db" name="<%=SQLInjection.CCN_LIMIT%>" type="text" value="<%=employee.getCcnLimit()%>"/>
74 </TD>
75 </TR>
```

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 73 (Privacy Violation) — Critical

```
76 <TR><TD>
```

ViewProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ccn
From: org.owasp.webgoat.session.Employee.getCcn
File: JavaSource/org/owasp/webgoat/session/Employee.java:132

```
129
130 public String getCcn()
131 {
132 return ccn;
133 }
134
135
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:67
Taint Flags: PRIVATE

```
64 Credit Card:
65 </TD>
66 <TD>
67 <%=employee.getCcn()%>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.CCN
From: /lessons/SQLInjection.EditProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/EditProfile.jsp:67

```
64 Credit Card:
65 </TD>
66 <TD>
67 <input class="lesson_text_db" name="<%=SQLInjection.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:67
Taint Flags: PRIVATE

```
64 Credit Card:
65 </TD>
66 <TD>
67 <input class="lesson_text_db" name="<%=SQLInjection.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ccn
From: org.owasp.webgoat.session.Employee.getCcn
File: JavaSource/org/owasp/webgoat/session/Employee.java:132

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 67 (Privacy Violation) — Critical

```
129
130  public String getCcn()
131  {
132  return ccn;
133  }
134
135
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:67
Taint Flags: PRIVATE

```
64  Credit Card:
65  </TD>
66  <TD>
67  <input class="lesson_text_db" name="<%=SQLInjection.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68  </TD>
69  <TD>
70  Credit Card Limit:
```

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read this.ssn
From: org.owasp.webgoat.session.Employee.getSsn
File: JavaSource/org/owasp/webgoat/session/Employee.java:204

```
201
202  public String getSsn()
203  {
204  return ssn;
205  }
206
207
```

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 54 (Privacy Violation) — Critical

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: EditProfile.jsp:54
Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
54 <input class="lesson_text_db" name="<%=SQLInjection.SSN%>" type="text" value="<%=employee.getSsn()%>"/>
55 </TD>
56 <TD>
57 Salary:
```

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getCcn()
From: /lessons/SQLInjection.EditProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/EditProfile.jsp:67

```
64 Credit Card:
65 </TD>
66 <TD>
67 <input class="lesson_text_db" name="<%=SQLInjection.CCN%>" type="text" value="<%=employee.getCcn()
%>"/>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()

Privacy Violation — Critical

Package: /lessons/SQLInjection

EditProfile.jsp, line 67 (Privacy Violation) — Critical

Enclosing Method: _jspService()
File: EditProfile.jsp:67
Taint Flags: PRIVATE

```
64 Credit Card:
65 </TD>
66 <TD>
67 <input class="lesson_text_db" name="<%=SQLInjection.CCN%>" type="text" value="<%=employee.getCcn()%>"/>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

ViewProfile.jsp, line 67 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getCcn()
From: /lessons/SQLInjection.ViewProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/ViewProfile.jsp:67

```
64 Credit Card:
65 </TD>
66 <TD>
67 <%=employee.getCcn()%>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:67
Taint Flags: PRIVATE

Privacy Violation — Critical

Package: /lessons/SQLInjection

ViewProfile.jsp, line 67 (Privacy Violation) — Critical

```
64 Credit Card:
65 </TD>
66 <TD>
67 <%=employee.getCcn()%>
68 </TD>
69 <TD>
70 Credit Card Limit:
```

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: org.owasp.webgoat.session.Employee.getSsn()
From: /lessons/SQLInjection.ViewProfile.jsp._jspService
File: WebContent/lessons/SQLInjection/ViewProfile.jsp:54

```
51 SSN:
52 </TD>
53 <TD>
54 <%=employee.getSsn()%>
55 </TD>
56 <TD>
57 Salary:
```

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: ViewProfile.jsp:54
Taint Flags: PRIVATE

```
51 SSN:
52 </TD>
53 <TD>
```

Privacy Violation — Critical

Package: /lessons/SQLInjection

ViewProfile.jsp, line 54 (Privacy Violation) — Critical

```
54 <%=employee.getSsn()%>
55 </TD>
56 <TD>
57 Salary:
```

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.CCN_LIMIT
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav
a:95

```
92 String ccn = s.getParser().getStringParameter(
93 RoleBasedAccessControl.CCN);
94 int ccnLimit = s.getParser().getIntParameter(
95 RoleBasedAccessControl.CCN_LIMIT);
96 String disciplinaryActionDate = s.getParser().getStringParameter(
97 RoleBasedAccessControl.DISCIPLINARY_DATE);
98 String disciplinaryActionNotes = s.getParser().getStringParameter(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693 if (values == null)
694 {
695 throw new ParameterNotFoundException(name + " not found");
696 }
697 else if (values[0].length() == 0)
698 {
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.PASSWORD
From: org.owasp.webgoat.lessons.SQLInjection.Login.updateLessonStatus
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:272

```
269  String employeeId = s.getParser().getStringParameter(
270  SQLInjection.EMPLOYEE_ID);
271  String password = s.getParser().getRawParameter(
272  SQLInjection.PASSWORD);
273  switch (getStage(s))
274  {
275  case 1:
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629  if (values == null)
630  {
631  throw new ParameterNotFoundException(name + " not found");
632  }
633  else if (values[0].length() == 0)
634  {
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 699 (Privacy Violation) — Critical

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessControl.CCN
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.java:93

```
90 int salary = s.getParser().getIntParameter(
91 RoleBasedAccessControl.SALARY);
92 String ccn = s.getParser().getStringParameter(
93 RoleBasedAccessControl.CCN);
94 int ccnLimit = s.getParser().getIntParameter(
95 RoleBasedAccessControl.CCN_LIMIT);
96 String disciplinaryActionDate = s.getParser().getStringParameter(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696 }
697 else if (values[0].length() == 0)
698 {
699 throw new ParameterNotFoundException(name + " was empty");
700 }
701 else
702 {
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.HtmlClues.PASSWORD
From: org.owasp.webgoat.lessons.HtmlClues.backdoor
File: JavaSource/org/owasp/webgoat/lessons/HtmlClues.java:80

```
77 private boolean backdoor(WebSession s)
78 {
79 String username = s.getParser().getRawParameter(USERNAME, "");
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 635 (Privacy Violation) — Critical

```
80  String password = s.getParser().getRawParameter(PASSWORD, "");
81
82  //<START_OMIT_SOURCE>
83  return (username.equals("admin") && password.equals("adminpw"));
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632  }
633  else if (values[0].length() == 0)
634  {
635  throw new ParameterNotFoundException(name + " was empty");
636  }
637
638  return (values[0]);
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakSessionID.PASSWORD
From: org.owasp.webgoat.lessons.WeakSessionID.makeLogin
File: JavaSource/org/owasp/webgoat/lessons/WeakSessionID.java:232

```
229  {}
230  try
231  {
232  password = s.getParser().getStringParameter(PASSWORD);
233  }
234  catch (ParameterNotFoundException pnfe)
235  {}
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 699 (Privacy Violation) — Critical

Enclosing Method: getStringParameter()
File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696 }
697 else if (values[0].length() == 0)
698 {
699 throw new ParameterNotFoundException(name + " was empty");
700 }
701 else
702 {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.SSN
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav
a:77

```
74 String lastName = s.getParser().getStringParameter(
75 RoleBasedAccessControl.LAST_NAME);
76 String ssn = s.getParser().getStringParameter(
77 RoleBasedAccessControl.SSN);
78 String title = s.getParser().getStringParameter(
79 RoleBasedAccessControl.TITLE);
80 String phone = s.getParser().getStringParameter(
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

```
55 }
56 }
57
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.DOS_Login.PASSWORD
From: org.owasp.webgoat.lessons.DOS_Login.createContent
File: JavaSource/org/owasp/webgoat/lessons/DOS_Login.java:89

```
86 String username = "";
87 String password = "";
88 username = s.getParser().getRawParameter(USERNAME);
89 password = s.getParser().getRawParameter(PASSWORD);
90
91 // don;t allow user name from other lessons. it would be too simple.
92 if (username.equals("jeff") || username.equals("dave"))
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632 }
633 else if (values[0].length() == 0)
634 {
635 throw new ParameterNotFoundException(name + " was empty");
636 }
637
638 return (values[0]);
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 635 (Privacy Violation) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.PASSWORD
From: org.owasp.webgoat.lessons.SQLInjection.Login.updateLessonStatus
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:272

```
269 String employeeId = s.getParser().getStringParameter(
270 SQLInjection.EMPLOYEE_ID);
271 String password = s.getParser().getRawParameter(
272 SQLInjection.PASSWORD);
273 switch (getStage(s))
274 {
275 case 1:
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632 }
633 else if (values[0].length() == 0)
634 {
635 throw new ParameterNotFoundException(name + " was empty");
636 }
637
638 return (values[0]);
```

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.PASSWORD
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.Login.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/Login.java:78

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

```
75  employeeId = s.getParser().getIntParameter(
76  RoleBasedAccessControl.EMPLOYEE_ID);
77  String password = s.getParser().getStringParameter(
78  RoleBasedAccessControl.PASSWORD);
79
80  // Attempt authentication
81  if (login(s, employeeId, password))
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693  if (values == null)
694  {
695  throw new ParameterNotFoundException(name + " not found");
696  }
697  else if (values[0].length() == 0)
698  {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.HtmlClues.PASSWORD
From: org.owasp.webgoat.lessons.HtmlClues.backdoor
File: JavaSource/org/owasp/webgoat/lessons/HtmlClues.java:80

```
77  private boolean backdoor(WebSession s)
78  {
79  String username = s.getParser().getRawParameter(USERNAME, "");
80  String password = s.getParser().getRawParameter(PASSWORD, "");
81
82  //<START_OMIT_SOURCE>
83  return (username.equals("admin") && password.equals("adminpw"));
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakAuthenticationCookie.PASSWORD
From: org.owasp.webgoat.lessons.FailOpenAuthentication.createContent
File: JavaSource/org/owasp/webgoat/lessons/FailOpenAuthentication.java:73

```
70 try
71 {
72 username = s.getParser().getRawParameter(USERNAME);
73 password = s.getParser().getRawParameter(PASSWORD);
74
75 // if credentials are bad, send the login page
76 if (!"webgoat".equals(username) || !password.equals("webgoat"))
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

```
54 super(s);
55 }
56 }
57
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.PASSWORD
From: org.owasp.webgoat.lessons.SQLInjection.Login.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:78

```
75 employeeId = s.getParser().getStringParameter(
76 SQLInjection.EMPLOYEE_ID);
77 String password = s.getParser().getRawParameter(
78 SQLInjection.PASSWORD);
79
80 // Attempt authentication
81 boolean authenticated = login(s, employeeId, password);
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632 }
633 else if (values[0].length() == 0)
634 {
635 throw new ParameterNotFoundException(name + " was empty");
636 }
637
638 return (values[0]);
```

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessControl.SSN
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.java:77

```
74 String lastName = s.getParser().getStringParameter(
75 RoleBasedAccessControl.LAST_NAME);
76 String ssn = s.getParser().getStringParameter(
77 RoleBasedAccessControl.SSN);
78 String title = s.getParser().getStringParameter(
79 RoleBasedAccessControl.TITLE);
80 String phone = s.getParser().getStringParameter(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693 if (values == null)
694 {
695 throw new ParameterNotFoundException(name + " not found");
696 }
697 else if (values[0].length() == 0)
698 {
```

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

Source: Read org.owasp.webgoat.lessons.WeakSessionID.PASSWORD
From: org.owasp.webgoat.lessons.WeakSessionID.makeLogin
File: JavaSource/org/owasp/webgoat/lessons/WeakSessionID.java:232

```
229 {}
230 try
231 {
232 password = s.getParser().getStringParameter(PASSWORD);
233 }
234 catch (ParameterNotFoundException pnfe)
235 {}
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693 if (values == null)
694 {
695 throw new ParameterNotFoundException(name + " not found");
696 }
697 else if (values[0].length() == 0)
698 {
```

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakAuthenticationCookie.PASSWORD
From: org.owasp.webgoat.lessons.WeakAuthenticationCookie.checkParams
File: JavaSource/org/owasp/webgoat/lessons/WeakAuthenticationCookie.java:127

```
124 protected String checkParams(WebSession s) throws Exception
125 {
126 String username = s.getParser().getStringParameter(USERNAME, "");
127 String password = s.getParser().getStringParameter(PASSWORD, "");
128
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

```
129  if ((username.length() > 0) && (password.length() > 0))
130  {
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693  if (values == null)
694  {
695  throw new ParameterNotFoundException(name + " not found");
696  }
697  else if (values[0].length() == 0)
698  {
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.SSN
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav
a:77

```
74  String lastName = s.getParser().getStringParameter(
75  RoleBasedAccessControl.LAST_NAME);
76  String ssn = s.getParser().getStringParameter(
77  RoleBasedAccessControl.SSN);
78  String title = s.getParser().getStringParameter(
79  RoleBasedAccessControl.TITLE);
80  String phone = s.getParser().getStringParameter(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 699 (Privacy Violation) — Critical

File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696 }
697 else if (values[0].length() == 0)
698 {
699 throw new ParameterNotFoundException(name + " was empty");
700 }
701 else
702 {
```

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakAuthenticationCookie.PASSWORD
From: org.owasp.webgoat.lessons.FailOpenAuthentication.createContent
File: JavaSource/org/owasp/webgoat/lessons/FailOpenAuthentication.java:73

```
70 try
71 {
72 username = s.getParser().getRawParameter(USERNAME);
73 password = s.getParser().getRawParameter(PASSWORD);
74
75 // if credentials are bad, send the login page
76 if (!"webgoat".equals(username) || !password.equals("webgoat"))
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629 if (values == null)
630 {
631 throw new ParameterNotFoundException(name + " not found");
632 }
633 else if (values[0].length() == 0)
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 631 (Privacy Violation) — Critical

```
634 {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.PASSWORD
From: org.owasp.webgoat.lessons.SQLInjection.Login.updateLessonStatus
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:272

```
269 String employeeId = s.getParser().getStringParameter(
270 SQLInjection.EMPLOYEE_ID);
271 String password = s.getParser().getRawParameter(
272 SQLInjection.PASSWORD);
273 switch (getStage(s))
274 {
275 case 1:
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 631 (Privacy Violation) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.HtmlClues.PASSWORD
From: org.owasp.webgoat.lessons.HtmlClues.backdoor
File: JavaSource/org/owasp/webgoat/lessons/HtmlClues.java:80

```
77 private boolean backdoor(WebSession s)
78 {
79 String username = s.getParser().getRawParameter(USERNAME, "");
80 String password = s.getParser().getRawParameter(PASSWORD, "");
81
82 //<START_OMIT_SOURCE>
83 return (username.equals("admin") && password.equals("adminpw"));
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629 if (values == null)
630 {
631 throw new ParameterNotFoundException(name + " not found");
632 }
633 else if (values[0].length() == 0)
634 {
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.CCN_LIMIT
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 699 (Privacy Violation) — Critical

a:95

```
92  String ccn = s.getParser().getStringParameter(
93  RoleBasedAccessControl.CCN);
94  int ccnLimit = s.getParser().getIntParameter(
95  RoleBasedAccessControl.CCN_LIMIT);
96  String disciplinaryActionDate = s.getParser().getStringParameter(
97  RoleBasedAccessControl.DISCIPLINARY_DATE);
98  String disciplinaryActionNotes = s.getParser().getStringParameter(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696  }
697  else if (values[0].length() == 0)
698  {
699  throw new ParameterNotFoundException(name + " was empty");
700  }
701  else
702  {
```

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WsSAXInjection.PASSWORD
From: org.owasp.webgoat.lessons.WsSAXInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/WsSAXInjection.java:149

```
146  {
147  ec.addElement(makeInputLine(s));
148
149  password = s.getParser().getRawParameter(PASSWORD, null);
150
151  PRE pre = new PRE();
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 631 (Privacy Violation) — Critical

```
152  String xml = template1;
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629  if (values == null)
630  {
631  throw new ParameterNotFoundException(name + " not found");
632  }
633  else if (values[0].length() == 0)
634  {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.CCN
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav
a:93

```
90  int salary = s.getParser().getIntParameter(
91  RoleBasedAccessControl.SALARY);
92  String ccn = s.getParser().getStringParameter(
93  RoleBasedAccessControl.CCN);
94  int ccnLimit = s.getParser().getIntParameter(
95  RoleBasedAccessControl.CCN_LIMIT);
96  String disciplinaryActionDate = s.getParser().getStringParameter(
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.XPATHInjection.PASSWORD
From: org.owasp.webgoat.lessons.XPATHInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/XPATHInjection.java:142

```
139 return ec;
140 }
141
142 String password = s.getParser().getRawParameter(PASSWORD, "");
143 if (password == null || password.length() == 0)
144 {
145 ec.addElement(new P().addElement(new StringElement(
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.XPATHInjection.PASSWORD
From: org.owasp.webgoat.lessons.XPATHInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/XPATHInjection.java:142

```
139  return ec;
140  }
141
142  String password = s.getParser().getRawParameter(PASSWORD, "");
143  if (password == null || password.length() == 0)
144  {
145  ec.addElement(new P().addElement(new StringElement(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629  if (values == null)
630  {
631  throw new ParameterNotFoundException(name + " not found");
632  }
633  else if (values[0].length() == 0)
634  {
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 699 (Privacy Violation) — Critical

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.Challenge2Screen.PASSWORD
From: org.owasp.webgoat.lessons.Challenge2Screen.doStage1
File: JavaSource/org/owasp/webgoat/lessons/Challenge2Screen.java:151

```
148  setStage(s, 1);
149
150  String username = s.getParser().getStringParameter(USERNAME, "");
151  String password = s.getParser().getStringParameter(PASSWORD, "");
152
153  if (username.equals(user) && password.equals(pass))
154  {
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696  }
697  else if (values[0].length() == 0)
698  {
699  throw new ParameterNotFoundException(name + " was empty");
700  }
701  else
702  {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.Challenge2Screen.PASSWORD
From: org.owasp.webgoat.lessons.Challenge2Screen.doStage1
File: JavaSource/org/owasp/webgoat/lessons/Challenge2Screen.java:151

```
148  setStage(s, 1);
149
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

```
150 String username = s.getParser().getStringParameter(USERNAME, "");
151 String password = s.getParser().getStringParameter(PASSWORD, "");
152
153 if (username.equals(user) && password.equals(pass))
154 {
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakAuthenticationCookie.PASSWORD
From: org.owasp.webgoat.lessons.FailOpenAuthentication.createContent
File: JavaSource/org/owasp/webgoat/lessons/FailOpenAuthentication.java:73

```
70 try
71 {
72 username = s.getParser().getRawParameter(USERNAME);
73 password = s.getParser().getRawParameter(PASSWORD);
74
75 // if credentials are bad, send the login page
76 if (!"webgoat".equals(username) || !password.equals("webgoat"))
```

Sink Details

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 635 (Privacy Violation) — Critical

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632 }
633 else if (values[0].length() == 0)
634 {
635 throw new ParameterNotFoundException(name + " was empty");
636 }
637
638 return (values[0]);
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.PASSWORD
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.Login.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/Login.java:78

```
75 employeeId = s.getParser().getIntParameter(
76 RoleBasedAccessControl.EMPLOYEE_ID);
77 String password = s.getParser().getStringParameter(
78 RoleBasedAccessControl.PASSWORD);
79
80 // Attempt authentication
81 if (login(s, employeeId, password))
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

```
55 }
56 }
57
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.PASSWORD
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.Login.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/Login.java:78

```
75 employeeId = s.getParser().getIntParameter(
76 RoleBasedAccessControl.EMPLOYEE_ID);
77 String password = s.getParser().getStringParameter(
78 RoleBasedAccessControl.PASSWORD);
79
80 // Attempt authentication
81 if (login(s, employeeId, password))
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696 }
697 else if (values[0].length() == 0)
698 {
699 throw new ParameterNotFoundException(name + " was empty");
700 }
701 else
702 {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.DOS_Login.PASSWORD
From: org.owasp.webgoat.lessons.DOS_Login.createContent
File: JavaSource/org/owasp/webgoat/lessons/DOS_Login.java:89

```
86 String username = "";
87 String password = "";
88 username = s.getParser().getRawParameter(USERNAME);
89 password = s.getParser().getRawParameter(PASSWORD);
90
91 // don;t allow user name from other lessons. it would be too simple.
92 if (username.equals("jeff") || username.equals("dave"))
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.CCN_LIMIT

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav
a:95

```
92 String ccn = s.getParser().getStringParameter(
93 RoleBasedAccessControl.CCN);
94 int ccnLimit = s.getParser().getIntParameter(
95 RoleBasedAccessControl.CCN_LIMIT);
96 String disciplinaryActionDate = s.getParser().getStringParameter(
97 RoleBasedAccessControl.DISCIPLINARY_DATE);
98 String disciplinaryActionNotes = s.getParser().getStringParameter(
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.RoleBasedAccessControl.RoleBasedAccessContro
l.CCN
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.UpdateProfile.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/UpdateProfile.jav
a:93

```
90 int salary = s.getParser().getIntParameter(
91 RoleBasedAccessControl.SALARY);
92 String ccn = s.getParser().getStringParameter(
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

```
93  RoleBasedAccessControl.CCN);
94  int ccnLimit = s.getParser().getIntParameter(
95  RoleBasedAccessControl.CCN_LIMIT);
96  String disciplinaryActionDate = s.getParser().getStringParameter(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693  if (values == null)
694  {
695  throw new ParameterNotFoundException(name + " not found");
696  }
697  else if (values[0].length() == 0)
698  {
```

ParameterParser.java, line 699 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakAuthenticationCookie.PASSWORD
From: org.owasp.webgoat.lessons.WeakAuthenticationCookie.checkParams
File: JavaSource/org/owasp/webgoat/lessons/WeakAuthenticationCookie.java:127

```
124  protected String checkParams(WebSession s) throws Exception
125  {
126  String username = s.getParser().getStringParameter(USERNAME, "");
127  String password = s.getParser().getStringParameter(PASSWORD, "");
128
129  if ((username.length() > 0) && (password.length() > 0))
130  {
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 699 (Privacy Violation) — Critical

Enclosing Method: getStringParameter()
File: ParameterParser.java:699
Taint Flags: PRIVATE

```
696 }
697 else if (values[0].length() == 0)
698 {
699 throw new ParameterNotFoundException(name + " was empty");
700 }
701 else
702 {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakAuthenticationCookie.PASSWORD
From: org.owasp.webgoat.lessons.WeakAuthenticationCookie.checkParams
File: JavaSource/org/owasp/webgoat/lessons/WeakAuthenticationCookie.java:127

```
124 protected String checkParams(WebSession s) throws Exception
125 {
126 String username = s.getParser().getStringParameter(USERNAME, "");
127 String password = s.getParser().getStringParameter(PASSWORD, "");
128
129 if ((username.length() > 0) && (password.length() > 0))
130 {
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

```
56 }
57
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WsSAXInjection.PASSWORD
From: org.owasp.webgoat.lessons.WsSAXInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/WsSAXInjection.java:149

```
146 {
147 ec.addElement(makeInputLine(s));
148
149 password = s.getParser().getRawParameter(PASSWORD, null);
150
151 PRE pre = new PRE();
152 String xml = template1;
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632 }
633 else if (values[0].length() == 0)
634 {
635 throw new ParameterNotFoundException(name + " was empty");
636 }
637
638 return (values[0]);
```

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 631 (Privacy Violation) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.PASSWORD
From: org.owasp.webgoat.lessons.SQLInjection.Login.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:78

```
75 employeeId = s.getParser().getStringParameter(
76 SQLInjection.EMPLOYEE_ID);
77 String password = s.getParser().getRawParameter(
78 SQLInjection.PASSWORD);
79
80 // Attempt authentication
81 boolean authenticated = login(s, employeeId, password);
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629 if (values == null)
630 {
631 throw new ParameterNotFoundException(name + " not found");
632 }
633 else if (values[0].length() == 0)
634 {
```

ParameterParser.java, line 695 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.Challenge2Screen.PASSWORD
From: org.owasp.webgoat.lessons.Challenge2Screen.doStage1
File: JavaSource/org/owasp/webgoat/lessons/Challenge2Screen.java:151

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 695 (Privacy Violation) — Critical

```
148  setStage(s, 1);
149
150  String username = s.getParser().getStringParameter(USERNAME, "");
151  String password = s.getParser().getStringParameter(PASSWORD, "");
152
153  if (username.equals(user) && password.equals(pass))
154  {
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getStringParameter()
File: ParameterParser.java:695
Taint Flags: PRIVATE

```
692
693  if (values == null)
694  {
695  throw new ParameterNotFoundException(name + " not found");
696  }
697  else if (values[0].length() == 0)
698  {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WsSAXInjection.PASSWORD
From: org.owasp.webgoat.lessons.WsSAXInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/WsSAXInjection.java:149

```
146  {
147  ec.addElement(makeInputLine(s));
148
149  password = s.getParser().getRawParameter(PASSWORD, null);
150
151  PRE pre = new PRE();
152  String xml = template1;
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterParser.java, line 635 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.XPATHInjection.PASSWORD
From: org.owasp.webgoat.lessons.XPATHInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/XPATHInjection.java:142

```
139 return ec;
140 }
141
142 String password = s.getParser().getRawParameter(PASSWORD, "");
143 if (password == null || password.length() == 0)
144 {
145 ec.addElement(new P().addElement(new StringElement(
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:635
Taint Flags: PRIVATE

```
632 }
633 else if (values[0].length() == 0)
634 {
```

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterParser.java, line 635 (Privacy Violation) — Critical

```
635 throw new ParameterNotFoundException(name + " was empty");
636 }
637
638 return (values[0]);
```

ParameterParser.java, line 631 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.DOS_Login.PASSWORD
From: org.owasp.webgoat.lessons.DOS_Login.createContent
File: JavaSource/org/owasp/webgoat/lessons/DOS_Login.java:89

```
86 String username = "";
87 String password = "";
88 username = s.getParser().getRawParameter(USERNAME);
89 password = s.getParser().getRawParameter(PASSWORD);
90
91 // don;t allow user name from other lessons. it would be too simple.
92 if (username.equals("jeff") || username.equals("dave"))
```

Sink Details

Sink: org.owasp.webgoat.session.ParameterNotFoundException.ParameterNotFoundException()
Enclosing Method: getRawParameter()
File: ParameterParser.java:631
Taint Flags: PRIVATE

```
628
629 if (values == null)
630 {
631 throw new ParameterNotFoundException(name + " not found");
632 }
633 else if (values[0].length() == 0)
634 {
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.WeakSessionID.PASSWORD
From: org.owasp.webgoat.lessons.WeakSessionID.makeLogin
File: JavaSource/org/owasp/webgoat/lessons/WeakSessionID.java:232

```
229 {}
230 try
231 {
232 password = s.getParser().getStringParameter(PASSWORD);
233 }
234 catch (ParameterNotFoundException pnfe)
235 {}
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read org.owasp.webgoat.lessons.SQLInjection.SQLInjection.PASSWORD

Privacy Violation — Critical

Package: org.owasp.webgoat.session

ParameterNotFoundException.java, line 54 (Privacy Violation) — Critical

From: org.owasp.webgoat.lessons.SQLInjection.Login.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:78

```
75 employeeId = s.getParser().getStringParameter(
76 SQLInjection.EMPLOYEE_ID);
77 String password = s.getParser().getRawParameter(
78 SQLInjection.PASSWORD);
79
80 // Attempt authentication
81 boolean authenticated = login(s, employeeId, password);
```

Sink Details

Sink: java.lang.Exception.Exception()
Enclosing Method: ParameterNotFoundException()
File: ParameterNotFoundException.java:54
Taint Flags: PRIVATE

```
51 */
52 public ParameterNotFoundException(String s)
53 {
54 super(s);
55 }
56 }
57
```

Privacy Violation — High

Package: org.owasp.webgoat.session

LessonTracker.java, line 187 (Privacy Violation) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read
From: org.owasp.webgoat.lessons.ForgotPassword.getTitle
File: JavaSource/org/owasp/webgoat/lessons/ForgotPassword.java:330

```
327 */
328 public String getTitle()
329 {
```

Privacy Violation — High

Package: org.owasp.webgoat.session

LessonTracker.java, line 187 (Privacy Violation) — High

```
330  return ("Forgot Password");
331  }
332
333
```

Sink Details

Sink: java.util.Properties.getProperty()
Enclosing Method: setProperties()
File: LessonTracker.java:187
Taint Flags: PASSWORD_STRING

```
184  + ".maxHintLevel"));
185  currentStage = Integer.parseInt(props.getProperty(screen.getTitle()
186  + ".currentStage"));
187  numVisits = Integer.parseInt(props.getProperty(screen.getTitle()
188  + ".numVisits"));
189  viewedCookies = Boolean.valueOf(
190  props.getProperty(screen.getTitle() + ".viewedCookies"))
```

LessonTracker.java, line 183 (Privacy Violation) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read
From: org.owasp.webgoat.lessons.ForgotPassword.getTitle
File: JavaSource/org/owasp/webgoat/lessons/ForgotPassword.java:330

```
327  */
328  public String getTitle()
329  {
330  return ("Forgot Password");
331  }
332
333
```

Sink Details

Sink: java.util.Properties.getProperty()

Privacy Violation — High

Package: org.owasp.webgoat.session

LessonTracker.java, line 183 (Privacy Violation) — High

Enclosing Method: setProperties()
File: LessonTracker.java:183
Taint Flags: PASSWORD_STRING

```
180 completed = Boolean.valueOf(
181 props.getProperty(screen.getTitle() + ".completed"))
182 .booleanValue();
183 maxHintLevel = Integer.parseInt(props.getProperty(screen.getTitle()
184 + ".maxHintLevel"));
185 currentStage = Integer.parseInt(props.getProperty(screen.getTitle()
186 + ".currentStage"));
```

LessonTracker.java, line 185 (Privacy Violation) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: Read
From: org.owasp.webgoat.lessons.ForgotPassword.getTitle
File: JavaSource/org/owasp/webgoat/lessons/ForgotPassword.java:330

```
327 */
328 public String getTitle()
329 {
330 return ("Forgot Password");
331 }
332
333
```

Sink Details

Sink: java.util.Properties.getProperty()
Enclosing Method: setProperties()
File: LessonTracker.java:185
Taint Flags: PASSWORD_STRING

```
182 .booleanValue();
183 maxHintLevel = Integer.parseInt(props.getProperty(screen.getTitle()
184 + ".maxHintLevel"));
185 currentStage = Integer.parseInt(props.getProperty(screen.getTitle()
186 + ".currentStage"));
```

Privacy Violation	High
Package: org.owasp.webgoat.session	
LessonTracker.java, line 185 (Privacy Violation)	**High**

```
187  numVisits = Integer.parseInt(props.getProperty(screen.getTitle()
188  + ".numVisits"));
```

Race Condition: Singleton Member Field (1 issue)

Abstract

Servlet member fields might allow one user to see another user's data.

Explanation

Many Servlet developers do not understand that a Servlet is a singleton. There is only one instance of the Servlet, and that single instance is used and re-used to handle multiple requests that are processed simultaneously by different threads.

A common result of this misunderstanding is that developers use Servlet member fields in such a way that one user may inadvertently see another user's data. In other words, storing user data in Servlet member fields introduces a data access race condition.

Example 1: The following Servlet stores the value of a request parameter in a member field and then later echoes the parameter value to the response output stream.

```
public class GuestBook extends HttpServlet {

   String name;

   protected void doPost (HttpServletRequest req, HttpServletResponse res) {
     name = req.getParameter("name");
     ...
     out.println(name + ", thanks for visiting!");
   }
}
```

While this code will work perfectly in a single-user environment, if two users access the Servlet at approximately the same time, it is possible for the two request handler threads to interleave in the following way:

Thread 1: assign "`Dick`" to `name` Thread 2: assign "`Jane`" to `name` Thread 1: print "`Jane, thanks for visiting!`" Thread 2: print "`Jane, thanks for visiting!`"

Thereby showing the first user the second user's name.

Recommendation

Do not use Servlet member fields for anything but constants. (i.e. make all member fields `static final`).

Developers are often tempted to use Servlet member fields for user data when they need to transport data from one region of code to another. If this is your aim, consider declaring a separate class and using the Servlet only to "wrap" this new class.

Example 2: The bug in the example above can be corrected in the following way:

```
public class GuestBook extends HttpServlet {

   protected void doPost (HttpServletRequest req, HttpServletResponse res) {
    GBRequestHandler handler = new GBRequestHandler();
    handler.handle(req, res);
   }
```

```
}

public class GBRequestHandler {

   String name;

   public void handle(HttpServletRequest req, HttpServletResponse res) {
     name = req.getParameter("name");
     ...
     out.println(name + ", thanks for visiting!");
   }

}
```

Alternatively, a Servlet can utilize synchronized blocks to access servlet instance variables but using synchronized blocks may cause significant performance problems.

Please notice that wrapping the field access within a synchronized block will only prevent the issue if all read and write operations on that member are performed within the same synchronized block or method.

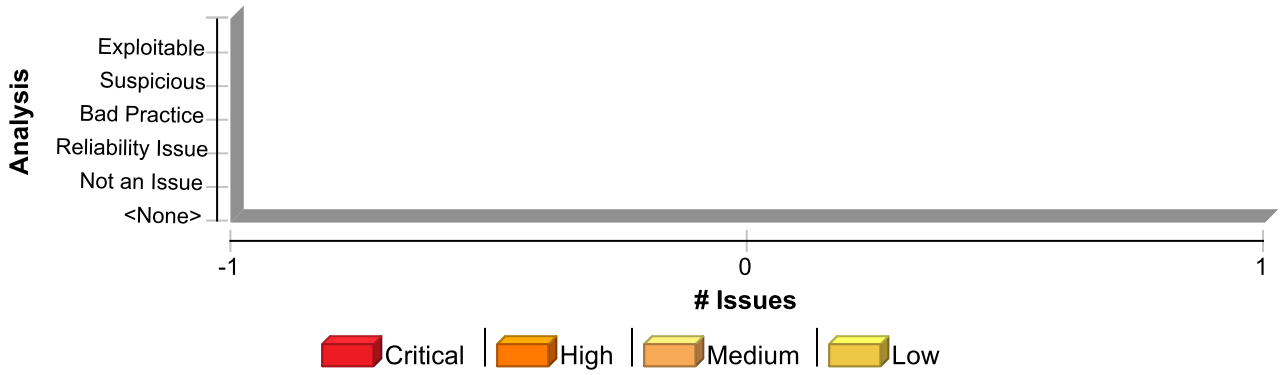
Example 3: Wrapping the Example 1 write operation (assignment) in a synchronized block will not fix the problem since the threads will have to get a lock to modify `name` field, but they will release the lock afterwards, allowing a second thread to change the value again. If, after changing the `name` value, the first thread resumes execution, the value printed will be the one assigned by the second thread:

```
public class GuestBook extends HttpServlet {

   String name;

   protected void doPost (HttpServletRequest req, HttpServletResponse res) {
     synchronized(name) {
        name = req.getParameter("name");
     }
     ...
     out.println(name + ", thanks for visiting!");
   }
}
```

In order to fix the race condition, all the write and read operations on the shared member field should be run atomically within the same synchronized block:

```
public class GuestBook extends HttpServlet {

   String name;

   protected void doPost (HttpServletRequest req, HttpServletResponse res) {
     synchronized(name) {
        name = req.getParameter("name");
        ...
        out.println(name + ", thanks for visiting!");
     }
   }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Race Condition: Singleton Member Field	1	1	0	2
Total	**1**	**1**	**0**	**2**

Race Condition: Singleton Member Field — High

Package: org.owasp.webgoat

HammerHead.java, line 135 (Race Condition: Singleton Member Field) — High

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: AssignmentStatement
Enclosing Method: doPost()
File: HammerHead.java:135

```
132
133  // FIXME: If a response is written by updateSession(), do not
134  // call makeScreen() and writeScreen()
135  mySession = updateSession(request, response, context);
136  if (response.isCommitted())
137  return;
138
```

Race Condition: Static Database Connection (2 issues)

Abstract

Database connections stored in static fields will be shared between threads.

Explanation

A transactional resource object such as database connection can only be associated with one transaction at a time. For this reason, a connection should not be shared between threads and should not be stored in a static field. See Section 4.2.3 of the J2EE Specification for more details.

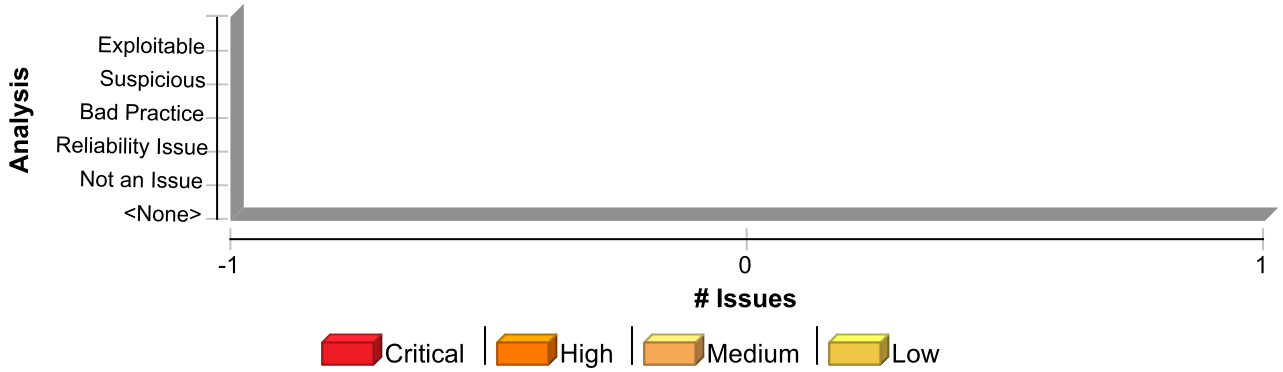
Example 1:

```
public class ConnectionManager {

private static Connection conn = initDbConn();
...
}
```

Recommendation

Rather than storing the database connection in a static field, use a connection pool to cache connection objects. Most modern J2EE and Servlet containers provide built-in connection pooling facilities.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Race Condition: Static Database Connection	2	2	0	4
Total	**2**	**2**	**0**	**4**

Race Condition: Static Database Connection	**High**
Package: org.owasp.webgoat.lessons	
SoapRequest.java, line 74 (Race Condition: Static Database Connection)	**High**

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Structural)

Race Condition: Static Database Connection — High

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 74 (Race Condition: Static Database Connection) — High

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Field: connection
File: SoapRequest.java:74

```
71 */
72
73 //static boolean completed;
74 public static Connection connection = null;
75
76 public final static String firstName = "getFirstName";
77
```

WSDLScanning.java, line 82 (Race Condition: Static Database Connection) — High

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Field: connection
File: WSDLScanning.java:82

```
79
80 static boolean beenRestartedYet = false;
81
82 public static Connection connection = null;
83
84 public final static String firstName = "getFirstName";
85
```

Redundant Null Check (3 issues)

Abstract

The program can dereference a null pointer, thereby causing a null pointer exception.

Explanation

Null pointer exceptions usually occur when one or more of the programmer's assumptions is violated. Specifically, dereference-after-check errors occur when a program makes an explicit check for null, but proceeds to dereference the object when it is known to be null. Errors of this type are often the result of a typo or programmer oversight.

Most null pointer issues result in general software reliability problems, but if attackers can intentionally cause the program to dereference a null pointer, they can use the resulting exception to mount a denial of service attack or to cause the application to reveal debugging information that will be valuable in planning subsequent attacks.

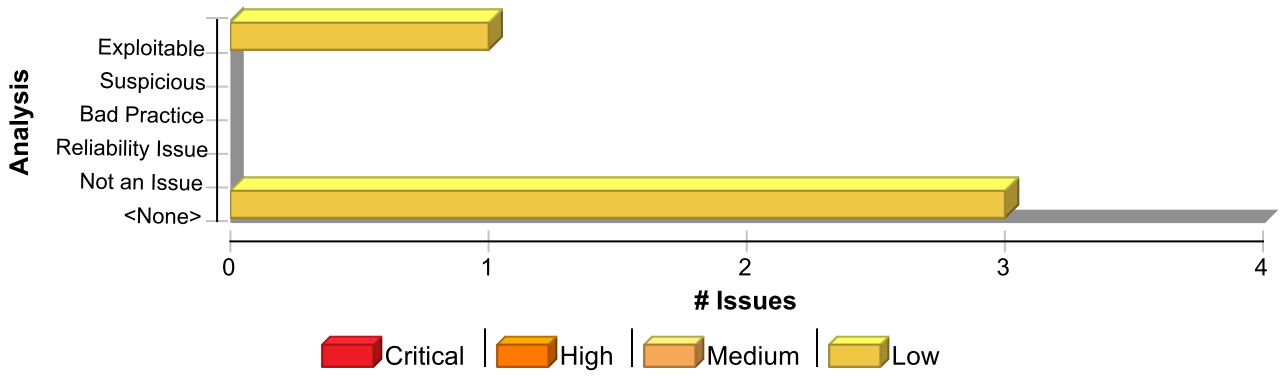
Example 1: In the following code, the programmer confirms that the variable `foo` is `null` and subsequently dereferences it erroneously. If `foo` is `null` when it is checked in the `if` statement, then a null dereference will occur, thereby causing a null pointer exception.

```
if (foo == null) {
    foo.setBar(val);
    ...
}
```

Recommendation

Implement careful checks before dereferencing objects that might be `null`. When possible, abstract null checks into wrappers around code that manipulates resources to ensure that they are applied in all cases and to minimize the places where mistakes can occur.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Redundant Null Check	3	3	0	6
Total	**3**	**3**	**0**	**6**

Redundant Null Check — Low

Package: org.owasp.webgoat.lessons

FailOpenAuthentication.java, line 104 (Redundant Null Check) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: Dereferenced : username
Enclosing Method: createContent()
File: FailOpenAuthentication.java:104

```
101 // We make sure the username was submitted to avoid telling the user an invalid
102 // username/password was entered when they first enter the lesson via the side menu.
103 // This also suppresses the error if they just hit the login and both fields are empty.
104 if (username.length() != 0)
105 {
106 s.setMessage("Invalid username and password entered.");
107 }
```

PathBasedAccessControl.java, line 110 (Redundant Null Check) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: Dereferenced : file
Enclosing Method: createContent()
File: PathBasedAccessControl.java:110

```
107 {
108 // allow them to look at any file in the webgoat hierachy. Don't allow them
109 // to look about the webgoat root, except to see the LICENSE file
110 if (upDirCount(file) == 3 && !file.endsWith("LICENSE"))
111 {
112 s.setMessage("Access denied");
113 s
```

FailOpenAuthentication.java, line 86 (Redundant Null Check) — Low

Issue Details

Redundant Null Check — Low

Package: org.owasp.webgoat.lessons

FailOpenAuthentication.java, line 86 (Redundant Null Check) — Low

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: Dereferenced : username
Enclosing Method: createContent()
File: FailOpenAuthentication.java:86

```
83 catch (Exception e)
84 {
85 // The parameter was omitted. set fail open status complete
86 if (username.length() > 0
87 && e.getMessage().indexOf("not found") != -1)
88 {
89 if ((username != null) && (username.length() > 0))
```

Resource Injection (2 issues)

Abstract

Allowing user input to control resource identifiers could enable an attacker to access or modify otherwise protected system resources.

Explanation

A resource injection issue occurs when the following two conditions are met:

1. An attacker is able to specify the identifier used to access a system resource.

For example, an attacker may be able to specify a port number to be used to connect to a network resource.

2. By specifying the resource, the attacker gains a capability that would not otherwise be permitted.

For example, the program may give the attacker the ability to transmit sensitive information to a third-party server.

Note: Resource injections involving resources stored on the file system are reported in a separate category named path manipulation. See the path manipulation description for further details of this vulnerability.

Example 1: The following code uses a port number read from an HTTP request to create a socket.

```
String remotePort = request.getParameter("remotePort");
...
ServerSocket srvr = new ServerSocket(remotePort);
Socket skt = srvr.accept();
...
```

Some think that in the mobile world, classic web application vulnerabilities, such as resource injection, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 2: The following code uses a URL read from an Android intent to load the page in `WebView`.

```
...
    WebView webview = new WebView(this);
    setContentView(webview);
        String url = this.getIntent().getExtras().getString("url");
    webview.loadUrl(url);
...
```

The kind of resource affected by user input indicates the kind of content that may be dangerous. For example, data containing special characters like period, slash, and backslash are risky when used in methods that interact with the file system. Similarly, data that contains URLs and URIs is risky for functions that create remote connections.

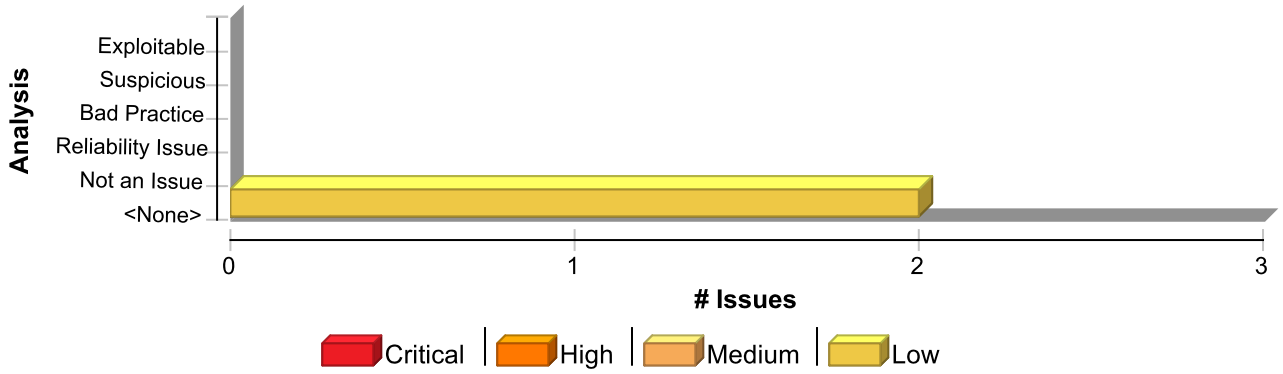
Recommendation

The best way to prevent resource injection is with a level of indirection: create a list of legitimate resource names that a user is allowed to specify, and only allow the user to select from the list. With this approach the input provided by

the user is never used directly to specify the resource name.

In some situations this approach is impractical because the set of legitimate resource names is too large or too hard to keep track of. Programmers often resort to blacklisting in these situations. Blacklisting selectively rejects or escapes potentially dangerous characters before using the input. However, any such list of unsafe characters is likely to be incomplete and will almost certainly become out of date. A better approach is to create a whitelist of characters that are allowed to appear in the resource name and accept input composed exclusively of characters in the approved set.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Resource Injection	2	2	0	4
Total	**2**	**2**	**0**	**4**

Resource Injection — Low

Package: org.owasp.webgoat.util

Interceptor.java, line 94 (Resource Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletContext.getInitParameter()
From: org.owasp.webgoat.util.Interceptor.doFilter
File: JavaSource/org/owasp/webgoat/util/Interceptor.java:85

```
82 String osgServerName = req.getSession().getServletContext()
83 .getInitParameter(OSG_SERVER_NAME);
84 String osgServerPort = req.getSession().getServletContext()
85 .getInitParameter(OSG_SERVER_PORT);
86
87 try
```

Resource Injection — Low

Package: org.owasp.webgoat.util

Interceptor.java, line 94 (Resource Injection) — Low

```
88 {
```

Sink Details

Sink: java.net.Socket.Socket()
Enclosing Method: doFilter()
File: Interceptor.java:94
Taint Flags: NUMBER, PROPERTY

```
91 && osgServerPort != null && osgServerPort.length() != 0)
92 {
93 osgSocket = new Socket(osgServerName, Integer
94 .parseInt(osgServerPort));
95 if (osgSocket != null)
96 {
97 out = new PrintWriter(osgSocket.getOutputStream(), true);
```

Interceptor.java, line 93 (Resource Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletContext.getInitParameter()
From: org.owasp.webgoat.util.Interceptor.doFilter
File: JavaSource/org/owasp/webgoat/util/Interceptor.java:83

```
80 PrintWriter out = null;
81 BufferedReader in = null;
82 String osgServerName = req.getSession().getServletContext()
83 .getInitParameter(OSG_SERVER_NAME);
84 String osgServerPort = req.getSession().getServletContext()
85 .getInitParameter(OSG_SERVER_PORT);
86
```

Sink Details

Sink: java.net.Socket.Socket()
Enclosing Method: doFilter()
File: Interceptor.java:93
Taint Flags: PROPERTY

```
90 if (osgServerName != null && osgServerName.length() != 0
```

Resource Injection	Low

Package: org.owasp.webgoat.util

Interceptor.java, line 93 (Resource Injection) — Low

```
91  && osgServerPort != null && osgServerPort.length() != 0)
92  {
93  osgSocket = new Socket(osgServerName, Integer
94  .parseInt(osgServerPort));
95  if (osgSocket != null)
96  {
```

SQL Injection (62 issues)

Abstract

Constructing a dynamic SQL statement with input coming from an untrusted source could allow an attacker to modify the statement's meaning or to execute arbitrary SQL commands.

Explanation

SQL injection errors occur when:

1. Data enters a program from an untrusted source.

2. The data is used to dynamically construct a SQL query.

Example 1: The following code dynamically constructs and executes a SQL query that searches for items matching a specified name. The query restricts the items displayed to those where the owner matches the user name of the currently-authenticated user.

```
...
    String userName = ctx.getAuthenticatedUserName();
    String itemName = request.getParameter("itemName");
    String query = "SELECT * FROM items WHERE owner = '"
                + userName + "' AND itemname = '"
                + itemName + "'";
    ResultSet rs = stmt.execute(query);
...
```

The query that this code intends to execute follows:

```
    SELECT * FROM items
    WHERE owner = <userName>
    AND itemname = <itemName>;
```

However, because the query is constructed dynamically by concatenating a constant base query string and a user input string, the query only behaves correctly if `itemName` does not contain a single-quote character. If an attacker with the user name `wiley` enters the string "`name' OR 'a'='a`" for `itemName`, then the query becomes the following:

```
    SELECT * FROM items
    WHERE owner = 'wiley'
    AND itemname = 'name' OR 'a'='a';
```

The addition of the `OR 'a'='a'` condition causes the where clause to always evaluate to true, so the query becomes logically equivalent to the much simpler query:

```
    SELECT * FROM items;
```

This simplification of the query allows the attacker to bypass the requirement that the query only return items owned by the authenticated user; the query now returns all entries stored in the `items` table, regardless of their specified owner.

Example 2: This example examines the effects of a different malicious value passed to the query constructed and executed in Example 1. If an attacker with the user name `wiley` enters the string "`name'; DELETE FROM items; --`" for `itemName`, then the query becomes the following two queries:

```
    SELECT * FROM items
    WHERE owner = 'wiley'
    AND itemname = 'name';

  DELETE FROM items;

  --'
```

Many database servers, including Microsoft(R) SQL Server 2000, allow multiple SQL statements separated by semicolons to be executed at once. While this attack string results in an error on Oracle and other database servers that do not allow the batch-execution of statements separated by semicolons, on databases that do allow batch execution, this type of attack allows the attacker to execute arbitrary commands against the database.

Notice the trailing pair of hyphens (--), which specifies to most database servers that the remainder of the statement is to be treated as a comment and not executed [4]. In this case the comment character serves to remove the trailing single-quote left over from the modified query. On a database where comments are not allowed to be used in this way, the general attack could still be made effective using a trick similar to the one shown in Example 1. If an attacker enters the string "`name'); DELETE FROM items; SELECT * FROM items WHERE 'a'='a`", the following three valid statements will be created:

```
    SELECT * FROM items
    WHERE owner = 'wiley'
    AND itemname = 'name';

  DELETE FROM items;

  SELECT * FROM items WHERE 'a'='a';
```

Some think that in the mobile world, classic web application vulnerabilities, such as SQL injection, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 3: The following code adapts Example 1 to the Android platform.

```
...
        PasswordAuthentication pa = authenticator.getPasswordAuthentication();
        String userName = pa.getUserName();
        String itemName = this.getIntent().getExtras().getString("itemName");
        String query = "SELECT * FROM items WHERE owner = '"
                               + userName + "' AND itemname = '"
                               + itemName + "'";
        SQLiteDatabase db = this.openOrCreateDatabase("DB", MODE_PRIVATE,
null);
        Cursor c = db.rawQuery(query, null);
...
```

One traditional approach to preventing SQL injection attacks is to handle them as an input validation problem and

either accept only characters from a whitelist of safe values or identify and escape a blacklist of potentially malicious values. Whitelisting can be a very effective means of enforcing strict input validation rules, but parameterized SQL statements require less maintenance and can offer more guarantees with respect to security. As is almost always the case, blacklisting is riddled with loopholes that make it ineffective at preventing SQL injection attacks. For example, attackers may:

- Target fields that are not quoted - Find ways to bypass the need for certain escaped meta-characters - Use stored procedures to hide the injected meta-characters

Manually escaping characters in input to SQL queries can help, but it will not make your application secure from SQL injection attacks.

Another solution commonly proposed for dealing with SQL injection attacks is to use stored procedures. Although stored procedures prevent some types of SQL injection attacks, they fail to protect against many others. Stored procedures typically help prevent SQL injection attacks by limiting the types of statements that can be passed to their parameters. However, there are many ways around the limitations and many interesting statements that can still be passed to stored procedures. Again, stored procedures can prevent some exploits, but they will not make your application secure against SQL injection attacks.

Recommendation

The root cause of a SQL injection vulnerability is the ability of an attacker to change context in the SQL query, causing a value that the programmer intended to be interpreted as data to be interpreted as a command instead. When a SQL query is constructed, the programmer knows what should be interpreted as part of the command and what should be interpreted as data. Parameterized SQL statements can enforce this behavior by disallowing data-directed context changes and preventing nearly all SQL injection attacks. Parameterized SQL statements are constructed using strings of regular SQL, but where user-supplied data needs to be included, they include bind parameters, which are placeholders for data that is subsequently inserted. In other words, bind parameters allow the programmer to explicitly specify to the database what should be treated as a command and what should be treated as data. When the program is ready to execute a statement, it specifies to the database the runtime values to use for each of the bind parameters without the risk that the data will be interpreted as a modification to the command.

Example 1 can be rewritten to use parameterized SQL statements (instead of concatenating user supplied strings) as follows:

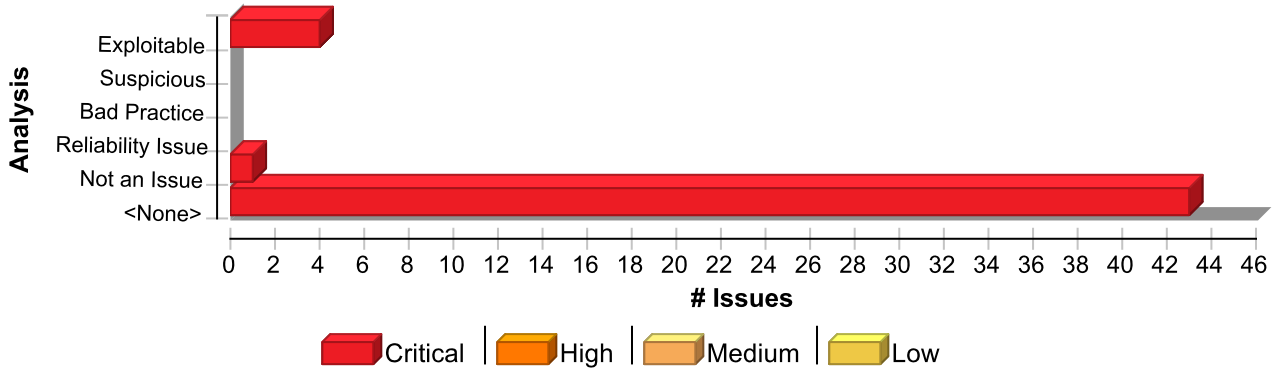
```
...
    String userName = ctx.getAuthenticatedUserName();
    String itemName = request.getParameter("itemName");
    String query = "SELECT * FROM items WHERE itemname=? AND owner=?";
    PreparedStatement stmt = conn.prepareStatement(query);
    stmt.setString(1, itemName);
    stmt.setString(2, userName);
    ResultSet results = stmt.execute();
...
```

And here is an Android equivalent:

```
...
    PasswordAuthentication pa = authenticator.getPasswordAuthentication();
    String userName = pa.getUserName();
    String itemName = this.getIntent().getExtras().getString("itemName");
    String query = "SELECT * FROM items WHERE itemname=? AND owner=?";
    SQLiteDatabase db = this.openOrCreateDatabase("DB", MODE_PRIVATE, null);
    Cursor c = db.rawQuery(query, new Object[]{itemName, userName});
...
```

More complicated scenarios, often found in report generation code, require that user input affect the structure of the SQL statement, for instance by adding a dynamic constraint in the `WHERE` clause. Do not use this requirement to justify concatenating user input to create a query string. Prevent SQL injection attacks where user input must affect command structure with a level of indirection: create a set of legitimate strings that correspond to different elements you might include in a SQL statement. When constructing a statement, use input from the user to select from this set of application-controlled values.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
SQL Injection	62	62	0	124
Total	**62**	**62**	**0**	**124**

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 240 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: getCreditCard(0)
From: org.owasp.webgoat.lessons.WsSqlInjection.getCreditCard
File: JavaSource/org/owasp/webgoat/lessons/WsSqlInjection.java:252

```
249  }
250
251
252  public String[] getCreditCard(String id)
253  {
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 240 (SQL Injection) — Critical

```
254  ResultSet results = getResults(id);
255  if ((results != null))
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: getResults()
File: WsSqlInjection.java:240
Taint Flags: WEBSERVICE, XSS

```
237  Statement statement = connection.createStatement(
238  ResultSet.TYPE_SCROLL_INSENSITIVE,
239  ResultSet.CONCUR_READ_ONLY);
240  ResultSet results = statement.executeQuery(query);
241  return results;
242  }
243  catch (SQLException sqle)
```

ThreadSafetyProblem.java, line 103 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: createContent()
File: ThreadSafetyProblem.java:103
Taint Flags: WEB, XSS

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

ThreadSafetyProblem.java, line 103 (SQL Injection) — Critical

```
100 Statement statement = connection.createStatement(
101 ResultSet.TYPE_SCROLL_INSENSITIVE,
102 ResultSet.CONCUR_READ_ONLY);
103 ResultSet results = statement.executeQuery(query);
104
105 if ((results != null) && (results.first() == true))
106 {
```

BlindSqlInjection.java, line 122 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: createContent()
File: BlindSqlInjection.java:122
Taint Flags: PRIMARY_KEY, WEB, XSS

```
119 Statement statement = connection.createStatement(
120 ResultSet.TYPE_SCROLL_INSENSITIVE,
121 ResultSet.CONCUR_READ_ONLY);
122 ResultSet results = statement.executeQuery(query);
123
124 if ((results != null) && (results.first() == true))
125 {
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 134 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createContent()
File: DOS_Login.java:134
Taint Flags: WEB, XSS

```
131 + "', '"
132 + s.getUserName()
133 + "' )";
134 statement.executeUpdate(insertData1);
135 }
136 // check the total count of logins
137 query = "SELECT * FROM user_login WHERE webgoat_user = '"
```

Challenge2Screen.java, line 220 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 220 (SQL Injection) — Critical

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: org.owasp.webgoat.lessons.Challenge2Screen.getCookie
File: JavaSource/org/owasp/webgoat/lessons/Challenge2Screen.java:801

```
798 */
799 protected String getCookie(WebSession s)
800 {
801 Cookie[] cookies = s.getRequest().getCookies();
802
803 for (int i = 0; i < cookies.length; i++)
804 {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: doStage2()
File: Challenge2Screen.java:220
Taint Flags: WEB, XSS

```
217 Vector<String> v = new Vector<String>();
218 try
219 {
220 ResultSet results = statement3.executeQuery(query);
221
222 while (results.next())
223 {
```

BackDoors.java, line 106 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction	Indeterminate (Below Not An Issue threshold)

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 106 (SQL Injection) — Critical

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: concept1()
File: BackDoors.java:106
Taint Flags: WEB, XSS

```
103 ResultSet.CONCUR_READ_ONLY);
104 if (arrSQL.length == 2)
105 {
106 statement.executeUpdate(arrSQL[1]);
107
108 getLessonTracker(s).setStage(2);
109 s
```

SqlStringInjection.java, line 112 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 112 (SQL Injection) — Critical

```
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: injectableQuery()
File: SqlStringInjection.java:112
Taint Flags: WEB, XSS

```
109  Statement statement = connection.createStatement(
110  ResultSet.TYPE_SCROLL_INSENSITIVE,
111  ResultSet.CONCUR_READ_ONLY);
112  ResultSet results = statement.executeQuery(query);
113
114  if ((results != null) && (results.first() == true))
115  {
```

BackDoors.java, line 113 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: concept1()
File: BackDoors.java:113

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

BackDoors.java, line 113 (SQL Injection) — Critical

Taint Flags: WEB, XSS

```
110  .setMessage("You have succeeded in exploiting the vulnerable query and created another SQL statement. Now move to stage 2 to learn how to create a backdoor or a DB worm");
111  }
112
113  ResultSet rs = statement.executeQuery(arrSQL[0]);
114  if (rs.next())
115  {
116  Table t = new Table(0).setCellSpacing(0).setCellPadding(0)
```

DOS_Login.java, line 114 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: createContent()
File: DOS_Login.java:114
Taint Flags: WEB, XSS

```
111  Statement statement = connection.createStatement(
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 114 (SQL Injection) — Critical

```
112 ResultSet.TYPE_SCROLL_INSENSITIVE,
113 ResultSet.CONCUR_READ_ONLY);
114 ResultSet results = statement.executeQuery(query);
115 if ((results != null) && (results.first() == true))
116 {
117 ResultSetMetaData resultsMetaData = results.getMetaData();
```

WsSqlInjection.java, line 240 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Not an Issue
AA_Prediction	Not an Issue

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Not an Issue] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Not an Issue]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: getResults()
File: WsSqlInjection.java:240
Taint Flags: PRIMARY_KEY, WEB, XSS

```
237 Statement statement = connection.createStatement(
238 ResultSet.TYPE_SCROLL_INSENSITIVE,
239 ResultSet.CONCUR_READ_ONLY);
240 ResultSet results = statement.executeQuery(query);
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 240 (SQL Injection) — Critical

```
241 return results;
242 }
243 catch (SQLException sqle)
```

SqlNumericInjection.java, line 130 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: injectableQuery()
File: SqlNumericInjection.java:130
Taint Flags: WEB, XSS

```
127 Statement statement = connection.createStatement(
128 ResultSet.TYPE_SCROLL_INSENSITIVE,
129 ResultSet.CONCUR_READ_ONLY);
130 ResultSet results = statement.executeQuery(query);
131
132 if ((results != null) && (results.first() == true))
133 {
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:134

```
131 String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132 String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136 String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137 String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:140

```
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
139  .getParameter(CrossSiteScripting.MANAGER));
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141  int salary = Integer.parseInt(request
142  .getParameter(CrossSiteScripting.SALARY));
143  String ccn = request.getParameter(CrossSiteScripting.CCN);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:151

```
148  String disciplinaryActionNotes = request
149  .getParameter(CrossSiteScripting.DISCIPLINARY_NOTES);
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

```
150  String personalDescription = request
151  .getParameter(CrossSiteScripting.DESCRIPTION);
152
153  Employee employee = new Employee(subjectId, firstName, lastName, ssn,
154  title, phone, address1, address2, manager, startDate, salary,
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
5

```
132  String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133  String ssn = request.getParameter(CrossSiteScripting.SSN);
134  String title = request.getParameter(CrossSiteScripting.TITLE);
135  String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: PRIMARY_KEY, WEB, XSS

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:143

```
140 String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141 int salary = Integer.parseInt(request
142 .getParameter(CrossSiteScripting.SALARY));
143 String ccn = request.getParameter(CrossSiteScripting.CCN);
144 int ccnLimit = Integer.parseInt(request
145 .getParameter(CrossSiteScripting.CCN_LIMIT));
146 String disciplinaryActionDate = request
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337 {
338 Statement statement = WebSession.getConnection(s)
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

```
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:143

```
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141  int salary = Integer.parseInt(request
142  .getParameter(CrossSiteScripting.SALARY));
143  String ccn = request.getParameter(CrossSiteScripting.CCN);
144  int ccnLimit = Integer.parseInt(request
145  .getParameter(CrossSiteScripting.CCN_LIMIT));
146  String disciplinaryActionDate = request
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:133

```
130 HttpServletRequest request = s.getRequest();
131 String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132 String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136 String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245 Statement answer_statement = WebSession.getConnection(s)
246 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247 ResultSet.CONCUR_READ_ONLY);
248 answer_statement.executeUpdate(query);
249 }
250 catch (SQLException sqle)
251 {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:137

```
134  String title = request.getParameter(CrossSiteScripting.TITLE);
135  String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
139  .getParameter(CrossSiteScripting.MANAGER));
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:134

```
131  String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

```
132 String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136 String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137 String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245 Statement answer_statement = WebSession.getConnection(s)
246 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247 ResultSet.CONCUR_READ_ONLY);
248 answer_statement.executeUpdate(query);
249 }
250 catch (SQLException sqle)
251 {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:149

```
146 String disciplinaryActionDate = request
147 .getParameter(CrossSiteScripting.DISCIPLINARY_DATE);
148 String disciplinaryActionNotes = request
149 .getParameter(CrossSiteScripting.DISCIPLINARY_NOTES);
150 String personalDescription = request
151 .getParameter(CrossSiteScripting.DESCRIPTION);
152
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:136

```
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136 String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137 String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138 int manager = Integer.parseInt(request
139 .getParameter(CrossSiteScripting.MANAGER));
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
5

```
132 String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136 String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137 String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138 int manager = Integer.parseInt(request
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: PRIMARY_KEY, WEB, XSS

```
245 Statement answer_statement = WebSession.getConnection(s)
246 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247 ResultSet.CONCUR_READ_ONLY);
248 answer_statement.executeUpdate(query);
249 }
250 catch (SQLException sqle)
251 {
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:131

```
128  // is the better solution.
129
130  HttpServletRequest request = s.getRequest();
131  String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132  String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133  String ssn = request.getParameter(CrossSiteScripting.SSN);
134  String title = request.getParameter(CrossSiteScripting.TITLE);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
1

```
128  // is the better solution.
129
130  HttpServletRequest request = s.getRequest();
131  String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132  String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133  String ssn = request.getParameter(CrossSiteScripting.SSN);
134  String title = request.getParameter(CrossSiteScripting.TITLE);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
7

```
134  String title = request.getParameter(CrossSiteScripting.TITLE);
135  String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
139  .getParameter(CrossSiteScripting.MANAGER));
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:14
0

```
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
139  .getParameter(CrossSiteScripting.MANAGER));
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

```
140  String startDate = request.getParameter(CrossSiteScripting.START_DATE);
141  int salary = Integer.parseInt(request
142  .getParameter(CrossSiteScripting.SALARY));
143  String ccn = request.getParameter(CrossSiteScripting.CCN);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:133

```
130  HttpServletRequest request = s.getRequest();
131  String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132  String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133  String ssn = request.getParameter(CrossSiteScripting.SSN);
134  String title = request.getParameter(CrossSiteScripting.TITLE);
135  String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
```

Sink Details

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:132

```
129
130 HttpServletRequest request = s.getRequest();
131 String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132 String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337 {
338 Statement statement = WebSession.getConnection(s)
339 .createStatement();
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

```
340 statement.executeUpdate(query);
341 }
342 catch (SQLException sqle)
343 {
```

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:132

```
129
130 HttpServletRequest request = s.getRequest();
131 String firstName = request.getParameter(CrossSiteScripting.FIRST_NAME);
132 String lastName = request.getParameter(CrossSiteScripting.LAST_NAME);
133 String ssn = request.getParameter(CrossSiteScripting.SSN);
134 String title = request.getParameter(CrossSiteScripting.TITLE);
135 String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245 Statement answer_statement = WebSession.getConnection(s)
246 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247 ResultSet.CONCUR_READ_ONLY);
248 answer_statement.executeUpdate(query);
249 }
250 catch (SQLException sqle)
251 {
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 248 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:13
6

```
133  String ssn = request.getParameter(CrossSiteScripting.SSN);
134  String title = request.getParameter(CrossSiteScripting.TITLE);
135  String phone = request.getParameter(CrossSiteScripting.PHONE_NUMBER);
136  String address1 = request.getParameter(CrossSiteScripting.ADDRESS1);
137  String address2 = request.getParameter(CrossSiteScripting.ADDRESS2);
138  int manager = Integer.parseInt(request
139  .getParameter(CrossSiteScripting.MANAGER));
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:248
Taint Flags: WEB, XSS

```
245  Statement answer_statement = WebSession.getConnection(s)
246  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
247  ResultSet.CONCUR_READ_ONLY);
248  answer_statement.executeUpdate(query);
249  }
250  catch (SQLException sqle)
251  {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:151

```
148  String disciplinaryActionNotes = request
149  .getParameter(CrossSiteScripting.DISCIPLINARY_NOTES);
150  String personalDescription = request
151  .getParameter(CrossSiteScripting.DESCRIPTION);
152
153  Employee employee = new Employee(subjectId, firstName, lastName, ssn,
154  title, phone, address1, address2, manager, startDate, salary,
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

UpdateProfile.java, line 340 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: org.owasp.webgoat.lessons.CrossSiteScripting.UpdateProfile.parseEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/UpdateProfile.java:147

```
144  int ccnLimit = Integer.parseInt(request
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 340 (SQL Injection) — Critical

```
145  .getParameter(CrossSiteScripting.CCN_LIMIT));
146  String disciplinaryActionDate = request
147  .getParameter(CrossSiteScripting.DISCIPLINARY_DATE);
148  String disciplinaryActionNotes = request
149  .getParameter(CrossSiteScripting.DISCIPLINARY_NOTES);
150  String personalDescription = request
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:340
Taint Flags: WEB, XSS

```
337  {
338  Statement statement = WebSession.getConnection(s)
339  .createStatement();
340  statement.executeUpdate(query);
341  }
342  catch (SQLException sqle)
343  {
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Login.java, line 148 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Login.java, line 148 (SQL Injection) — Critical

```
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: login()
File: Login.java:148
Taint Flags: WEB, XSS

```
145  Statement answer_statement = WebSession.getConnection(s)
146  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
147  ResultSet.CONCUR_READ_ONLY);
148  ResultSet answer_results = answer_statement.executeQuery(query);
149  if (answer_results.first())
150  {
151  setSessionAttribute(s,
```

UpdateProfile.java, line 176 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeQuery()

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 176 (SQL Injection) — Critical

Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:176
Taint Flags: WEB, XSS

```
173 Statement answer_statement = WebSession.getConnection(s)
174 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
175 ResultSet.CONCUR_READ_ONLY);
176 ResultSet answer_results = answer_statement.executeQuery(query);
177 }
178 catch (SQLException sqle)
179 {
```

UpdateProfile.java, line 295 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687 public String getStringParameter(String name)
688 throws ParameterNotFoundException
689 {
690 String[] values = request.getParameterValues(name);
691 String value;
692
693 if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:295
Taint Flags: WEB, XSS

```
292 {
293 Statement statement = WebSession.getConnection(s)
294 .createStatement();
295 statement.executeUpdate(query);
296 }
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 295 (SQL Injection) — Critical

```
297 catch (SQLException sqle)
298 {
```

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 149 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: login()
File: Login.java:149
Taint Flags: WEB, XSS

```
146 Statement answer_statement = WebSession.getConnection(s)
147 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
148 ResultSet.CONCUR_READ_ONLY);
149 ResultSet answer_results = answer_statement.executeQuery(query);
150 if (answer_results.first())
151 {
152 setSessionAttribute(s,
```

ViewProfile.java, line 118 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 118 (SQL Injection) — Critical

Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:118
Taint Flags: PRIMARY_KEY, WEB, XSS

```
115 Statement answer_statement = WebSession.getConnection(s)
116 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
117 ResultSet.CONCUR_READ_ONLY);
118 ResultSet answer_results = answer_statement.executeQuery(query);
119 if (answer_results.next())
120 {
121 // Note: Do NOT get the password field.
```

Login.java, line 149 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 149 (SQL Injection) — Critical

File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: login()
File: Login.java:149
Taint Flags: PRIMARY_KEY, WEB, XSS

```
146  Statement answer_statement = WebSession.getConnection(s)
147  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
148  ResultSet.CONCUR_READ_ONLY);
149  ResultSet answer_results = answer_statement.executeQuery(query);
150  if (answer_results.first())
151  {
152  setSessionAttribute(s,
```

Login.java, line 191 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 191 (SQL Injection) — Critical

```
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: login_BACKUP()
File: Login.java:191
Taint Flags: WEB, XSS

```
188  Statement answer_statement = WebSession.getConnection(s)
189  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
190  ResultSet.CONCUR_READ_ONLY);
191  ResultSet answer_results = answer_statement.executeQuery(query);
192  if (answer_results.first())
193  {
194  setSessionAttribute(s,
```

ViewProfile.java, line 178 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:178
Taint Flags: PRIMARY_KEY, WEB, XSS

```
175  Statement answer_statement = WebSession.getConnection(s)
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 178 (SQL Injection) — Critical

```
176  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
177  ResultSet.CONCUR_READ_ONLY);
178  ResultSet answer_results = answer_statement.executeQuery(query);
179  if (answer_results.next())
180  {
181  // Note: Do NOT get the password field.
```

Login.java, line 191 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: login_BACKUP()
File: Login.java:191
Taint Flags: PRIMARY_KEY, WEB, XSS

```
188  Statement answer_statement = WebSession.getConnection(s)
189  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
190  ResultSet.CONCUR_READ_ONLY);
191  ResultSet answer_results = answer_statement.executeQuery(query);
192  if (answer_results.first())
193  {
194  setSessionAttribute(s,
```

SQL Injection — Critical

Package: org.owasp.webgoat.lessons.admin

ViewDatabase.java, line 90 (SQL Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: java.sql.Statement.executeQuery()
Enclosing Method: createContent()
File: ViewDatabase.java:90
Taint Flags: WEB, XSS

```
87  ResultSet.TYPE_SCROLL_INSENSITIVE,
88  ResultSet.CONCUR_READ_ONLY);
89  ResultSet results = statement.executeQuery(sqlStatement
90  .toString());
91
92  if ((results != null) && (results.first() == true))
93  {
```

SQL Injection — Low

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 162 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

SQL Injection — Low

Package: org.owasp.webgoat.lessons

DOS_Login.java, line 162 (SQL Injection) — Low

Sink Details

Sink: executeQuery()
Enclosing Method: createContent()
File: DOS_Login.java:162

```
159 // check the total count of logins
160 query = "SELECT * FROM user_login WHERE webgoat_user = '"
161 + s.getUserName() + "'";
162 results = statement.executeQuery(query);
163 results.last();
164 ec.addElement(new H2("Successfull login count: "
165 + results.getRow()));
```

DOS_Login.java, line 139 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: createContent()
File: DOS_Login.java:139

```
136 // check the total count of logins
137 query = "SELECT * FROM user_login WHERE webgoat_user = '"
138 + s.getUserName() + "'";
139 results = statement.executeQuery(query);
140 results.last();
141 // If they get back more than one user they succeeded
142 if (results.getRow() >= 3)
```

CSRF.java, line 181 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: makeList()
File: CSRF.java:181

SQL Injection — Low

Package: org.owasp.webgoat.lessons

CSRF.java, line 181 (SQL Injection) — Low

```
178
179  Statement statement = connection.createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY );
180
181  ResultSet results = statement.executeQuery( STANDARD_QUERY + " WHERE user_name LIKE '" + getNameroot( s.getUserName() ) + "%'" );
182
183  if ( ( results != null ) && ( results.first() == true ) )
184  {
```

DOS_Login.java, line 147 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeUpdate()
Enclosing Method: createContent()
File: DOS_Login.java:147

```
144  makeSuccess(s);
145  String deleteData1 = "DELETE from user_login WHERE webgoat_user = '"
146  + s.getUserName() + "'";
147  statement.executeUpdate(deleteData1);
148  return (new H1("Congratulations! Lesson Completed"));
149  }
150
```

DefaultLessonAction.java, line 265 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: isAuthorized()
File: DefaultLessonAction.java:265

```
262  query = "SELECT * FROM ownership WHERE employer_id = " + Integer.parseInt(employer_id) +
263  " AND employee_id = " + employeeId;
264  answer_statement = WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
```

SQL Injection — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 265 (SQL Injection) — Low

```
ResultSet.CONCUR_READ_ONLY );
265  answer_results = answer_statement.executeQuery( query );
266  authorized = answer_results.first();
267  }
268  }
```

DefaultLessonAction.java, line 256 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: isAuthorized()
File: DefaultLessonAction.java:256

```
253  try
254  {
255  Statement answer_statement = WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
256  ResultSet answer_results = answer_statement.executeQuery( query );
257  authorized = answer_results.first();
258
259  /* User is validated for function, but can the user perform that function on the specified user? */
```

DefaultLessonAction.java, line 203 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getUserName()
File: DefaultLessonAction.java:203

```
200  try
201  {
202  Statement answer_statement = WebSession.getConnection(s).createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY );
203  ResultSet answer_results = answer_statement.executeQuery( query );
```

SQL Injection — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 203 (SQL Injection) — Low

```
204 if (answer_results.next())
205 name = answer_results.getString("first_name");
206 }
```

StoredXss.java, line 343 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: makeList()
File: StoredXss.java:343

```
340 // but not anyone elses. This allows users to try out XSS to grab another user's
341 // cookies, but not get confused by other users scripts
342
343 ResultSet results = statement.executeQuery(STANDARD_QUERY
344 + " WHERE user_name LIKE '" + getNameroot(s.getUserName())
345 + "%'");
346
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 172 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:172

```
169 Statement answer_statement = WebSession.getConnection(s)
170 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
171 ResultSet.CONCUR_READ_ONLY);
172 ResultSet answer_results = answer_statement.executeQuery(query);
173 if (answer_results.next())
174 {
```

SQL Injection — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 172 (SQL Injection) — Low

```
175 // Note: Do NOT get the password field.
```

UpdateProfile.java, line 382 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: executeUpdate()
Enclosing Method: createEmployeeProfile_BACKUP()
File: UpdateProfile.java:382

```
379 {
380 Statement statement = WebSession.getConnection(s)
381 .createStatement();
382 statement.executeUpdate(query);
383 }
384 catch (SQLException sqle)
385 {
```

UpdateProfile.java, line 297 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: executeUpdate()
Enclosing Method: doChangeEmployeeProfile_BACKUP()
File: UpdateProfile.java:297

```
294 Statement answer_statement = WebSession.getConnection(s)
295 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
296 ResultSet.CONCUR_READ_ONLY);
297 answer_statement.executeUpdate(query);
298 }
299 catch (SQLException sqle)
300 {
```

SQL Injection — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 144 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeUpdate()
Enclosing Method: deleteEmployeeProfile_BACKUP()
File: DeleteProfile.java:144

```
141 Statement statement = WebSession.getConnection(s)
142 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
143 ResultSet.CONCUR_READ_ONLY);
144 statement.executeUpdate(query);
145 }
146 catch (SQLException sqle)
147 {
```

ListStaff.java, line 98 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getAllEmployees()
File: ListStaff.java:98

```
95 Statement answer_statement = WebSession.getConnection(s)
96 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
97 ResultSet.CONCUR_READ_ONLY);
98 ResultSet answer_results = answer_statement.executeQuery(query);
99 answer_results.beforeFirst();
100 while (answer_results.next())
101 {
```

UpdateProfile.java, line 311 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

SQL Injection — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 311 (SQL Injection) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeUpdate()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:311

```
308 {
309 Statement statement = WebSession.getConnection(s)
310 .createStatement();
311 statement.executeUpdate(query);
312 }
313 catch (SQLException sqle)
314 {
```

ListStaff.java, line 146 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:146

```
143 Statement answer_statement = WebSession.getConnection(s)
144 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
145 ResultSet.CONCUR_READ_ONLY);
146 ResultSet answer_results = answer_statement.executeQuery(query);
147 answer_results.beforeFirst();
148 while (answer_results.next())
149 {
```

UpdateProfile.java, line 225 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

SQL Injection — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 225 (SQL Injection) — Low

Sink: executeQuery()
Enclosing Method: changeEmployeeProfile_BACKUP()
File: UpdateProfile.java:225

```
222 Statement answer_statement = WebSession.getConnection(s)
223 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
224 ResultSet.CONCUR_READ_ONLY);
225 ResultSet answer_results = answer_statement.executeQuery(query);
226 }
227 catch (SQLException sqle)
228 {
```

ViewProfile.java, line 192 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:192

```
189 Statement answer_statement = WebSession.getConnection(s)
190 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
191 ResultSet.CONCUR_READ_ONLY);
192 ResultSet answer_results = answer_statement.executeQuery(query);
193 if (answer_results.next())
194 {
195 // Note: Do NOT get the password field.
```

Package: org.owasp.webgoat.lessons.SQLInjection

ListStaff.java, line 146 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getAllEmployees_BACKUP()

SQL Injection — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ListStaff.java, line 146 (SQL Injection) — Low

File: ListStaff.java:146

```
143 Statement answer_statement = WebSession.getConnection(s)
144 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
145 ResultSet.CONCUR_READ_ONLY);
146 ResultSet answer_results = answer_statement.executeQuery(query);
147 answer_results.beforeFirst();
148 while (answer_results.next())
149 {
```

ListStaff.java, line 98 (SQL Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: executeQuery()
Enclosing Method: getAllEmployees()
File: ListStaff.java:98

```
95 Statement answer_statement = WebSession.getConnection(s)
96 .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
97 ResultSet.CONCUR_READ_ONLY);
98 ResultSet answer_results = answer_statement.executeQuery(query);
99 answer_results.beforeFirst();
100 while (answer_results.next())
101 {
```

System Information Leak (192 issues)

Abstract

Revealing system data or debugging information helps an adversary learn about the system and form a plan of attack.

Explanation

An information leak occurs when system data or debugging information leaves the program through an output stream or logging function.

Example 1: The following code prints an exception to the standard error stream:

```
try {
    ...
} catch (Exception e) {
    e.printStackTrace();
}
```

Depending upon the system configuration, this information can be dumped to a console, written to a log file, or exposed to a remote user. For example, with scripting mechanisms it is trivial to redirect output information from "Standard error" or "Standard output" into a file or another program. Alternatively the system that the program runs on could have a remote logging mechanism such as a "syslog" server that will send the logs to a remote device. During development you will have no way of knowing where this information may end up being displayed.

In some cases the error message tells the attacker precisely what sort of an attack the system is vulnerable to. For example, a database error message can reveal that the application is vulnerable to a SQL injection attack. Other error messages can reveal more oblique clues about the system. In the example above, the leaked information could imply information about the type of operating system, the applications installed on the system, and the amount of care that the administrators have put into configuring the program.

Here is another scenario, specific to the mobile world. Most mobile devices now implement a Near-Field Communication (NFC) protocol for quickly sharing information between devices using radio communication. It works by bringing devices to close proximity or simply having them touch each other. Even though the communication range of NFC is limited to just a few centimeters, eavesdropping, data modification and various other types of attacks are possible, since NFC alone does not ensure secure communication.

Example 2: The Android platform provides support for NFC. The following code creates a message that gets pushed to the other device within the range.

```
...
public static final String TAG = "NfcActivity";
private static final String DATA_SPLITTER = "__:DATA:__";
private static final String MIME_TYPE = "application/my.applications.mimetype";
...
public NdefMessage createNdefMessage(NfcEvent event) {
    TelephonyManager tm =
(TelephonyManager)Context.getSystemService(Context.TELEPHONY_SERVICE);
    String VERSION = tm.getDeviceSoftwareVersion();
    String text = TAG + DATA_SPLITTER + VERSION;
    NdefRecord record = new NdefRecord(NdefRecord.TNF_MIME_MEDIA,
            MIME_TYPE.getBytes(), new byte[0], text.getBytes());
    NdefRecord[] records = { record };
    NdefMessage msg = new NdefMessage(records);
    return msg;
```

```
}
...
```

NFC Data Exchange Format (NDEF) message contains typed data, a URI, or a custom application payload. If the message contains information about the application, such as its name, MIME type, or device software version, this information could be leaked to an eavesdropper. In the example above, Fortify Static Code Analyzer reports a System Information Leak vulnerability on the return statement.

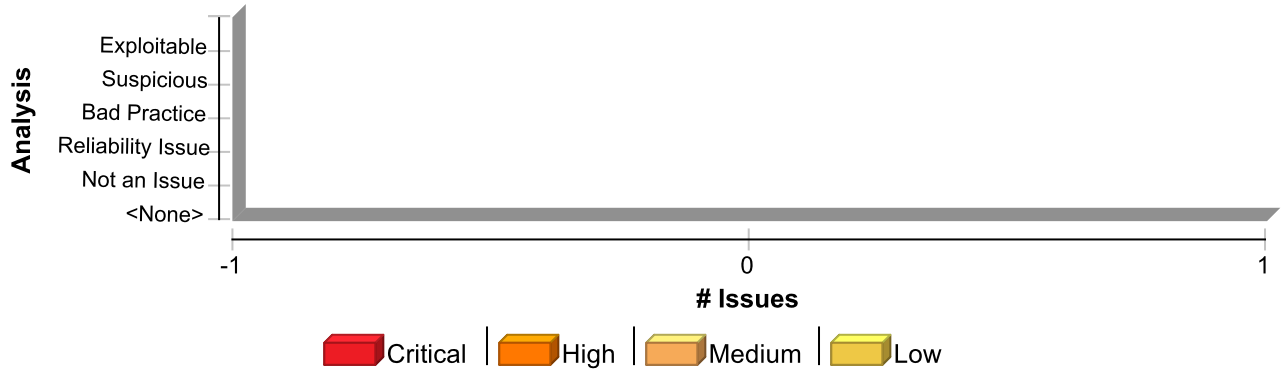
Recommendation

Write error messages with security in mind. In production environments, turn off detailed error information in favor of brief messages. Restrict the generation and storage of detailed output that can help administrators and programmers diagnose problems. Be careful, debugging traces can sometimes appear in non-obvious places (embedded in comments in the HTML for an error page, for example).

Even brief error messages that do not reveal stack traces or database dumps can potentially aid an attacker. For example, an "Access Denied" message can reveal that a file or user exists on the system.

If you are concerned about leaking system data via NFC on an Android device, you could do one of the following three things. Either do not include system data in the messages pushed to other devices in range, or encrypt the payload of the message, or establish secure communication channel at a higher layer.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
System Information Leak	192	192	0	384
Total	**192**	**192**	**0**	**384**

System Information Leak — **Low**

Package: org.owasp.webgoat

LessonSource.java, line 104 (System Information Leak) — **Low**

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

System Information Leak — Low

Package: org.owasp.webgoat

LessonSource.java, line 104 (System Information Leak) — Low

Sink Details

Sink: printStackTrace()
Enclosing Method: doPost()
File: LessonSource.java:104

```
101 }
102 catch (Throwable thr)
103 {
104 thr.printStackTrace();
105 log(request, "Could not write error screen: "
106 + thr.getMessage());
107 }
```

LessonSource.java, line 93 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doPost()
File: LessonSource.java:93

```
90 }
91 catch (Throwable t)
92 {
93 t.printStackTrace();
94 log("ERROR: " + t);
95 }
96 finally
```

HammerHead.java, line 192 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doPost()
File: HammerHead.java:192

System Information Leak — Low

Package: org.owasp.webgoat

HammerHead.java, line 192 (System Information Leak) — Low

```
189 }
190 catch (Throwable t)
191 {
192 t.printStackTrace();
193 log("ERROR: " + t);
194 screen = new ErrorScreen(mySession, t);
195 }
```

HammerHead.java, line 204 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doPost()
File: HammerHead.java:204

```
201 }
202 catch (Throwable thr)
203 {
204 thr.printStackTrace();
205 log(request, "Could not write error screen: "
206 + thr.getMessage());
207 }
```

Package: org.owasp.webgoat.lessons

Encoding.java, line 774 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: main()
File: Encoding.java:774

```
771 }
772 catch ( Exception e )
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 774 (System Information Leak) — Low

```
773 {
774 e.printStackTrace();
775 }
776 }
777
```

SilentTransactions.java, line 117 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SilentTransactions.java:117

```
114 }
115 catch (Exception ex)
116 {
117 ex.printStackTrace();
118 }
119
120 Form form = new Form(getFormAction(), Form.POST).setName("form")
```

WSDLScanning.java, line 264 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: WSDLScanning.java:264

```
261 catch (Exception e)
262 {
263 s.setMessage("Error generating " + this.getClass().getName());
264 e.printStackTrace();
265 }
266 return (ec);
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 264 (System Information Leak) — Low

```
267 }
```

BasicAuthentication.java, line 251 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doStage2()
File: BasicAuthentication.java:251

```
248 catch (Exception e)
249 {
250 s.setMessage("Error generating " + this.getClass().getName());
251 e.printStackTrace();
252 }
253
254 return (ec);
```

DefaultLessonAction.java, line 272 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isAuthorized()
File: DefaultLessonAction.java:272

```
269 catch ( SQLException sqle )
270 {
271 s.setMessage( "Error authorizing" );
272 sqle.printStackTrace();
273 }
274 }
275 catch ( Exception e )
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

BlindSqlInjection.java, line 344 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: BlindSqlInjection.java:344

```
341 catch (Exception e)
342 {
343 System.out.println("Exception caught: " + e);
344 e.printStackTrace(System.out);
345 }
346 }
347 }
```

HiddenFieldTampering.java, line 165 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: HiddenFieldTampering.java:165

```
162 catch (Exception e)
163 {
164 s.setMessage("Error generating " + this.getClass().getName());
165 e.printStackTrace();
166 }
167
168 return (ec);
```

SoapRequest.java, line 330 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 330 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: determineType()
File: SoapRequest.java:330

```
327 catch (Exception e)
328 {
329 s.setMessage("Error generating " + this.getClass().getName());
330 e.printStackTrace();
331 }
332
333 //DEVNOTE: Conditionally display Stage2 content depending on whether stage is completed or not
```

CommandInjection.java, line 211 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: CommandInjection.java:211

```
208 catch (Exception e)
209 {
210 s.setMessage("Error generating " + this.getClass().getName());
211 e.printStackTrace();
212 }
213
214 return (ec);
```

LessonAdapter.java, line 137 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 137 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: createStagedContent()
File: LessonAdapter.java:137

```
134 {
135 s.setMessage("Error generating " + this.getClass().getName());
136 System.out.println(e);
137 e.printStackTrace();
138 }
139
140 return (new StringElement(""));
```

Encoding.java, line 646 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: hashMD5()
File: Encoding.java:646

```
643 catch ( NoSuchAlgorithmException e )
644 {
645 // it's got to be there
646 e.printStackTrace();
647 }
648 return ( base64Encode( md.digest() ) );
649 }
```

WsSqlInjection.java, line 219 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: WsSqlInjection.java:219

```
216 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 219 (System Information Leak) — Low

```
217 {
218 s.setMessage("Error generating " + this.getClass().getName());
219 e.printStackTrace();
220 }
221 return (ec);
222 }
```

ThreadSafetyProblem.java, line 217 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: ThreadSafetyProblem.java:217

```
214 catch (Exception e)
215 {
216 System.out.println("Exception caught: " + e);
217 e.printStackTrace(System.out);
218 }
219 }
220
```

WsSAXInjection.java, line 166 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: WsSAXInjection.java:166

```
163 catch (Exception e)
164 {
165 s.setMessage("Error generating " + this.getClass().getName());
166 e.printStackTrace();
167 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 166 (System Information Leak) — Low

```
168 return (ec);
169 }
```

StoredXss.java, line 274 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: makeCurrent()
File: StoredXss.java:274

```
271 catch (Exception e)
272 {
273 s.setMessage("Error generating " + this.getClass().getName());
274 e.printStackTrace();
275 }
276
277 return (ec);
```

DefaultLessonAction.java, line 320 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isAuthorizedForEmployee()
File: DefaultLessonAction.java:320

```
317 catch ( SQLException sqle )
318 {
319 s.setMessage( "Error authorizing" );
320 sqle.printStackTrace();
321 }
322 }
323 catch ( Exception e )
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 729 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: sendMessage()
File: Challenge2Screen.java:729

```
726 catch (Exception e)
727 {
728 System.out.println("Couldn't write " + message + " to " + s);
729 e.printStackTrace();
730 }
731 }
732
```

DefaultLessonAction.java, line 216 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getUserName()
File: DefaultLessonAction.java:216

```
213 catch ( Exception e )
214 {
215 s.setMessage( "Error getting user name" );
216 e.printStackTrace();
217 }
218
219 return name;
```

XMLInjection.java, line 128 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons

XMLInjection.java, line 128 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: XMLInjection.java:128

```
125 }
126 catch (Exception ex)
127 {
128 ex.printStackTrace();
129 }
130
131 Form form = new Form(getFormAction(), Form.POST).setName("form")
```

SqlNumericInjection.java, line 323 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getStations()
File: SqlNumericInjection.java:323

```
320 }
321 catch (SQLException sqle)
322 {
323 sqle.printStackTrace();
324 }
325
326 return stations;
```

Encoding.java, line 672 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 672 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: hashSHA()
File: Encoding.java:672

```
669 catch ( NoSuchAlgorithmException e )
670 {
671 // it's got to be there
672 e.printStackTrace();
673 }
674 return ( base64Encode( md.digest() ) );
675 }
```

AccessControlMatrix.java, line 114 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: AccessControlMatrix.java:114

```
111 catch (Exception e)
112 {
113 s.setMessage("Error generating " + this.getClass().getName());
114 e.printStackTrace();
115 }
116
117 return (ec);
```

AbstractLesson.java, line 813 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isAuthorized()
File: AbstractLesson.java:813

```
810 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 813 (System Information Leak) — Low

```
811 {
812 s.setMessage("Error authorizing");
813 e.printStackTrace();
814 }
815 return authorized;
816 }
```

SqlNumericInjection.java, line 242 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: parameterizedQuery()
File: SqlNumericInjection.java:242

```
239 catch (Exception e)
240 {
241 s.setMessage("Error generating " + this.getClass().getName());
242 e.printStackTrace();
243 }
244
245 return (ec);
```

DefaultLessonAction.java, line 210 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getUserName()
File: DefaultLessonAction.java:210

```
207 catch ( SQLException sqle )
208 {
209 s.setMessage( "Error getting user name" );
210 sqle.printStackTrace();
211 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 210 (System Information Leak) — Low

```
212 }
213 catch ( Exception e )
```

HttpSplitting.java, line 137 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doHTTPSplitting()
File: HttpSplitting.java:137

```
134 catch (Exception e)
135 {
136 s.setMessage("Error generating " + this.getClass().getName());
137 e.printStackTrace();
138 }
139 return (ec);
140 }
```

UncheckedEmail.java, line 196 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: UncheckedEmail.java:196

```
193 catch (Exception e)
194 {
195 s.setMessage("Error generating " + this.getClass().getName());
196 e.printStackTrace();
197 }
198 return (ec);
199 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

HttpBasics.java, line 78 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: HttpBasics.java:78

```
75 catch (Exception e)
76 {
77 s.setMessage("Error generating " + this.getClass().getName());
78 e.printStackTrace();
79 }
80
81 if (!person.toString().equals(""")
```

SqlStringInjection.java, line 222 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: parameterizedQuery()
File: SqlStringInjection.java:222

```
219 catch (Exception e)
220 {
221 s.setMessage("Error generating " + this.getClass().getName());
222 e.printStackTrace();
223 }
224
225 return (ec);
```

XPATHInjection.java, line 207 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons

XPATHInjection.java, line 207 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: XPATHInjection.java:207

```
204 catch (IllegalArgumentException e)
205 {
206 s.setMessage("Error generating " + this.getClass().getName());
207 e.printStackTrace();
208 }
209 catch (XPathExpressionException e)
210 {
```

XPATHInjection.java, line 202 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: XPATHInjection.java:202

```
199 catch (IOException e)
200 {
201 s.setMessage("Error generating " + this.getClass().getName());
202 e.printStackTrace();
203 }
204 catch (IllegalArgumentException e)
205 {
```

AbstractLesson.java, line 1045 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 1045 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: readFromURL()
File: AbstractLesson.java:1045

```
1042 catch (Exception e)
1043 {
1044 System.out.println(e);
1045 e.printStackTrace();
1046 }
1047
1048 return (ec);
```

SqlStringInjection.java, line 152 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: injectableQuery()
File: SqlStringInjection.java:152

```
149 catch (Exception e)
150 {
151 s.setMessage("Error generating " + this.getClass().getName());
152 e.printStackTrace();
153 }
154
155 return (ec);
```

LogSpoofing.java, line 128 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: LogSpoofing.java:128

```
125 catch (UnsupportedEncodingException e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

LogSpoofing.java, line 128 (System Information Leak) — Low

```
126 {
127 s.setMessage("Error generating " + this.getClass().getName());
128 e.printStackTrace();
129 }
130 return ec;
131 }
```

DefaultLessonAction.java, line 326 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isAuthorizedForEmployee()
File: DefaultLessonAction.java:326

```
323 catch ( Exception e )
324 {
325 s.setMessage( "Error authorizing" );
326 e.printStackTrace();
327 }
328
329 // Update lesson status if necessary.
```

Encoding.java, line 459 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: Encoding.java:459

```
456
457 s.setMessage( "Error generating " + this.getClass().getName() );
458
459 e.printStackTrace();
460
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 459 (System Information Leak) — Low

```
461 }
462
```

HttpOnly.java, line 183 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createCustomCookieValue()
File: HttpOnly.java:183

```
180 original = value;
181
182 } catch (Exception e) {
183 e.printStackTrace();
184 }
185
186 return value;
```

SqlNumericInjection.java, line 162 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: injectableQuery()
File: SqlNumericInjection.java:162

```
159 catch (Exception e)
160 {
161 s.setMessage("Error generating " + this.getClass().getName());
162 e.printStackTrace();
163 }
164
165 return (ec);
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 163 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: accessWGService()
File: WSDLScanning.java:163

```
160 }
161 catch (Exception e)
162 {
163 e.printStackTrace();
164 }
165 return null;
166 }
```

WSDLScanning.java, line 159 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: accessWGService()
File: WSDLScanning.java:159

```
156 }
157 catch (ServiceException e)
158 {
159 e.printStackTrace();
160 }
161 catch (Exception e)
162 {
```

WSDLScanning.java, line 155 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 155 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: accessWGService()
File: WSDLScanning.java:155

```
152 }
153 catch (RemoteException e)
154 {
155 e.printStackTrace();
156 }
157 catch (ServiceException e)
158 {
```

BlindSqlInjection.java, line 145 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: BlindSqlInjection.java:145

```
142 catch (Exception e)
143 {
144 s.setMessage("Error generating " + this.getClass().getName());
145 e.printStackTrace();
146 }
147
148 return (ec);
```

DOMInjection.java, line 92 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons

DOMInjection.java, line 92 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: createContent()
File: DOMInjection.java:92

```
89 catch (Exception e)
90 {
91 s.setMessage("Error generating " + this.getClass().getName());
92 e.printStackTrace();
93 }
94
95 String lineSep = System.getProperty("line.separator");
```

HttpOnly.java, line 138 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: HttpOnly.java:138

```
135 catch ( Exception e )
136 {
137 s.setMessage( "Error generating " + this.getClass().getName() );
138 e.printStackTrace();
139 }
140
141 return ( ec );
```

WeakSessionID.java, line 138 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: WeakSessionID.java:138

```
135 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WeakSessionID.java, line 138 (System Information Leak) — Low

```
136 {
137 s.setMessage("Error generating " + this.getClass().getName());
138 e.printStackTrace();
139 }
140
141 return (null);
```

BasicAuthentication.java, line 160 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doStage1()
File: BasicAuthentication.java:160

```
157 catch (Exception e)
158 {
159 s.setMessage("Error generating " + this.getClass().getName());
160 e.printStackTrace();
161 }
162
163 return (ec);
```

TraceXSS.java, line 223 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: TraceXSS.java:223

```
220 catch (Exception e)
221 {
222 s.setMessage("Error generating " + this.getClass().getName());
223 e.printStackTrace();
224 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

TraceXSS.java, line 223 (System Information Leak) — Low

```
225 return (ec);
226 }
```

DOS_Login.java, line 172 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: DOS_Login.java:172

```
169 catch (SQLException sqle)
170 {
171 ec.addElement(new P().addElement(sqle.getMessage()));
172 sqle.printStackTrace();
173 }
174 }
175 catch (Exception e)
```

ThreadSafetyProblem.java, line 128 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: ThreadSafetyProblem.java:128

```
125 catch (Exception e)
126 {
127 s.setMessage("Error generating " + this.getClass().getName());
128 e.printStackTrace();
129 }
130
131 return (ec);
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 401 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SqlNumericInjection.java:401

```
398 catch (Exception e)
399 {
400 System.out.println("Exception caught: " + e);
401 e.printStackTrace(System.out);
402 }
403 }
404 }
```

DefaultLessonAction.java, line 278 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isAuthorized()
File: DefaultLessonAction.java:278

```
275 catch ( Exception e )
276 {
277 s.setMessage( "Error authorizing" );
278 e.printStackTrace();
279 }
280
281 // Update lesson status if necessary.
```

SqlStringInjection.java, line 317 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 317 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SqlStringInjection.java:317

```
314 catch (Exception e)
315 {
316 System.out.println("Exception caught: " + e);
317 e.printStackTrace(System.out);
318 }
319 }
320 }
```

AbstractLesson.java, line 807 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isAuthorized()
File: AbstractLesson.java:807

```
804 catch (SQLException sqle)
805 {
806 s.setMessage("Error authorizing");
807 sqle.printStackTrace();
808 }
809 }
810 catch (Exception e)
```

AbstractLesson.java, line 479 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 479 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: getFileText()
File: AbstractLesson.java:479

```
476 catch (Exception e)
477 {
478 System.out.println(e);
479 e.printStackTrace();
480 }
481
482 return (sb.toString());
```

Challenge2Screen.java, line 390 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: isDefaced()
File: Challenge2Screen.java:390

```
387 }
388 catch (Exception e)
389 {
390 e.printStackTrace();
391 }
392 return defaced;
393 //<END_OMIT_SOURCE>
```

PathBasedAccessControl.java, line 217 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: PathBasedAccessControl.java:217

```
214 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

PathBasedAccessControl.java, line 217 (System Information Leak) — Low

```
215 {
216 s.setMessage("Error generating " + this.getClass().getName());
217 e.printStackTrace();
218 }
219
220 return (ec);
```

CSRF.java, line 275 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: makeCurrent()
File: CSRF.java:275

```
272 catch ( Exception e )
273 {
274 s.setMessage( "Error generating " + this.getClass().getName() );
275 e.printStackTrace();
276 }
277
278 return ( ec );
```

JSONInjection.java, line 98 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: JSONInjection.java:98

```
95 }
96 catch (Exception ex)
97 {
98 ex.printStackTrace();
99 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

JSONInjection.java, line 98 (System Information Leak) — Low

```
100
101 Form form = new Form(getFormAction(), Form.POST).setName("form")
```

XPATHInjection.java, line 212 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: XPATHInjection.java:212

```
209 catch (XPathExpressionException e)
210 {
211 s.setMessage("Error generating " + this.getClass().getName());
212 e.printStackTrace();
213 }
214 return ec;
215 }
```

Challenge2Screen.java, line 441 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: resetWebPage()
File: Challenge2Screen.java:441

```
438 }
439 catch (Exception e)
440 {
441 e.printStackTrace();
442 }
443 }
444
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 273 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: viewWsdl()
File: SoapRequest.java:273

```
270 catch (Exception e)
271 {
272 s.setMessage("Error generating " + this.getClass().getName());
273 e.printStackTrace();
274 }
275
276 //DEVNOTE: Conditionally display Stage1 content depending on whether stage is completed or not
```

AbstractLesson.java, line 423 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getFileMethod()
File: AbstractLesson.java:423

```
420 catch (Exception e)
421 {
422 System.out.println(e);
423 e.printStackTrace();
424 }
425
426 return (sb.toString());
```

WeakAuthenticationCookie.java, line 197 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons

WeakAuthenticationCookie.java, line 197 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: WeakAuthenticationCookie.java:197

```
194 catch (Exception e)
195 {
196 s.setMessage("Error generating " + this.getClass().getName());
197 e.printStackTrace();
198 }
199
200 return (makeLogin(s));
```

ReflectedXSS.java, line 222 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: ReflectedXSS.java:222

```
219 catch (Exception e)
220 {
221 s.setMessage("Error generating " + this.getClass().getName());
222 e.printStackTrace();
223 }
224 return (ec);
225 }
```

JavaScriptValidation.java, line 259 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons

JavaScriptValidation.java, line 259 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: createContent()
File: JavaScriptValidation.java:259

```
256 catch (Exception e)
257 {
258 s.setMessage("Error generating " + this.getClass().getName());
259 e.printStackTrace();
260 }
261
262 return (ec);
```

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 379 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:379

```
376 {
377 s.setMessage("You are not authorized to perform this function");
378 System.out.println("Authorization failure");
379 ue2.printStackTrace();
380 }
381 catch (Exception e)
382 {
```

CrossSiteScripting.java, line 373 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 373 (System Information Leak) — Low

File: CrossSiteScripting.java:373

```
370 {
371 s.setMessage("Login failed");
372 System.out.println("Authentication failure");
373 ue.printStackTrace();
374 }
375 catch (UnauthorizedException ue2)
376 {
```

EditProfile.java, line 189 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: EditProfile.java:189

```
186 catch (Exception e)
187 {
188 s.setMessage("Error getting employee profile");
189 e.printStackTrace();
190 }
191
192 return profile;
```

CrossSiteScripting.java, line 366 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:366

```
363 catch (ValidationException ve)
364 {
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 366 (System Information Leak) — Low

```
365 System.out.println("Validation failed");
366 ve.printStackTrace();
367 setCurrentAction(s, ERROR_ACTION);
368 }
369 catch (UnauthenticatedException ue)
```

UpdateProfile.java, line 253 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:253

```
250 catch (SQLException sqle)
251 {
252 s.setMessage("Error updating employee profile");
253 sqle.printStackTrace();
254 }
255
256 }
```

UpdateProfile.java, line 345 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:345

```
342 catch (SQLException sqle)
343 {
344 s.setMessage("Error updating employee profile");
345 sqle.printStackTrace();
346 }
347 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 345 (System Information Leak) — Low

```
348 catch (Exception e)
```

UpdateProfile.java, line 115 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: UpdateProfile.java:115

```
112 catch (UnauthorizedException ue2)
113 {
114 System.out.println("Internal server error");
115 ue2.printStackTrace();
116 }
117 }
118 else
```

FindProfile.java, line 219 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:219

```
216 catch (Exception e)
217 {
218 s.setMessage("Error finding employee profile");
219 e.printStackTrace();
220 }
221
222 return profile;
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 206 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:206

```
203 catch (Exception e)
204 {
205 s.setMessage("Error getting employee profile");
206 e.printStackTrace();
207 }
208
209 return profile;
```

UpdateProfile.java, line 351 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:351

```
348 catch (Exception e)
349 {
350 s.setMessage("Error updating employee profile");
351 e.printStackTrace();
352 }
353 }
354
```

ViewProfile.java, line 200 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

ViewProfile.java, line 200 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:200

```
197 catch (SQLException sqle)
198 {
199 s.setMessage("Error getting employee profile");
200 sqle.printStackTrace();
201 }
202 }
203 catch (Exception e)
```

ViewProfile.java, line 140 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:140

```
137 catch (SQLException sqle)
138 {
139 s.setMessage("Error getting employee profile");
140 sqle.printStackTrace();
141 }
142 }
143 catch (Exception e)
```

CrossSiteScripting.java, line 385 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

CrossSiteScripting.java, line 385 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:385

```
382 {
383 // All other errors send the user to the generic error page
384 System.out.println("handleRequest() error");
385 e.printStackTrace();
386 setCurrentAction(s, ERROR_ACTION);
387 }
388 }
```

ViewProfile.java, line 146 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:146

```
143 catch (Exception e)
144 {
145 s.setMessage("Error getting employee profile");
146 e.printStackTrace();
147 }
148
149 return profile;
```

FindProfile.java, line 213 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:213

```
210 catch (SQLException sqle)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

FindProfile.java, line 213 (System Information Leak) — Low

```
211 {
212 s.setMessage("Error finding employee profile");
213 sqle.printStackTrace();
214 }
215 }
216 catch (Exception e)
```

UpdateProfile.java, line 110 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: UpdateProfile.java:110

```
107 catch (UnauthenticatedException ue1)
108 {
109 System.out.println("Internal server error");
110 ue1.printStackTrace();
111 }
112 catch (UnauthorizedException ue2)
113 {
```

UpdateProfile.java, line 302 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doChangeEmployeeProfile_BACKUP()
File: UpdateProfile.java:302

```
299 catch (SQLException sqle)
300 {
301 s.setMessage("Error updating employee profile");
302 sqle.printStackTrace();
303 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 302 (System Information Leak) — Low

```
304
305 }
```

CrossSiteScripting.java, line 360 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: CrossSiteScripting.java:360

```
357 catch (ParameterNotFoundException pnfe)
358 {
359 System.out.println("Missing parameter");
360 pnfe.printStackTrace();
361 setCurrentAction(s, ERROR_ACTION);
362 }
363 catch (ValidationException ve)
```

EditProfile.java, line 131 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:131

```
128 catch (Exception e)
129 {
130 s.setMessage("Error getting employee profile");
131 e.printStackTrace();
132 }
133
134 return profile;
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

EditProfile.java, line 183 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: EditProfile.java:183

```
180 catch (SQLException sqle)
181 {
182 s.setMessage("Error getting employee profile");
183 sqle.printStackTrace();
184 }
185 }
186 catch (Exception e)
```

EditProfile.java, line 125 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:125

```
122 catch (SQLException sqle)
123 {
124 s.setMessage("Error getting employee profile");
125 sqle.printStackTrace();
126 }
127 }
128 catch (Exception e)
```

UpdateProfile.java, line 387 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 387 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile_BACKUP()
File: UpdateProfile.java:387

```
384 catch (SQLException sqle)
385 {
386 s.setMessage("Error updating employee profile");
387 sqle.printStackTrace();
388 }
389 }
390 catch (Exception e)
```

UpdateProfile.java, line 260 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:260

```
257 catch (Exception e)
258 {
259 s.setMessage("Error updating employee profile");
260 e.printStackTrace();
261 }
262 }
263
```

FindProfile.java, line 121 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

FindProfile.java, line 121 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: FindProfile.java:121

```
118 catch (UnauthenticatedException ue1)
119 {
120 System.out.println("Internal server error");
121 ue1.printStackTrace();
122 }
123 catch (UnauthorizedException ue2)
124 {
```

UpdateProfile.java, line 309 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doChangeEmployeeProfile_BACKUP()
File: UpdateProfile.java:309

```
306 catch (Exception e)
307 {
308 s.setMessage("Error updating employee profile");
309 e.printStackTrace();
310 }
311 }
312
```

UpdateProfile.java, line 393 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile_BACKUP()
File: UpdateProfile.java:393

```
390 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.CrossSiteScripting

UpdateProfile.java, line 393 (System Information Leak) — Low

```
391 {
392 s.setMessage("Error updating employee profile");
393 e.printStackTrace();
394 }
395 }
396
```

FindProfile.java, line 126 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: FindProfile.java:126

```
123 catch (UnauthorizedException ue2)
124 {
125 System.out.println("Internal server error");
126 ue2.printStackTrace();
127 }
128 }
129 }
```

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 181 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:181

```
178 catch (SQLException sqle)
179 {
180 s.setMessage("Error updating employee profile");
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 181 (System Information Leak) — Low

```
181 sqle.printStackTrace();
182 }
183
184 }
```

ListStaff.java, line 115 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: ListStaff.java:115

```
112 catch (SQLException sqle)
113 {
114 s.setMessage("Error getting employees");
115 sqle.printStackTrace();
116 }
117 }
118 catch (Exception e)
```

UpdateProfile.java, line 237 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: changeEmployeeProfile_BACKUP()
File: UpdateProfile.java:237

```
234 catch (Exception e)
235 {
236 s.setMessage("Error updating employee profile");
237 e.printStackTrace();
238 }
239 }
240
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 237 (System Information Leak) — Low

Login.java, line 163 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: login()
File: Login.java:163

```
160 catch (SQLException sqle)
161 {
162 s.setMessage("Error logging in");
163 sqle.printStackTrace();
164 }
165 }
166 catch (Exception e)
```

UpdateProfile.java, line 263 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getNextUID()
File: UpdateProfile.java:263

```
260 catch (ClassNotFoundException e)
261 {
262 // TODO Auto-generated catch block
263 e.printStackTrace();
264 }
265 return uid + 1;
266 }
```

EditProfile.java, line 131 (System Information Leak) — Low

Issue Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

EditProfile.java, line 131 (System Information Leak) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:131

```
128 catch (Exception e)
129 {
130 s.setMessage("Error getting employee profile");
131 e.printStackTrace();
132 }
133
134 return profile;
```

DeleteProfile.java, line 126 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: deleteEmployeeProfile()
File: DeleteProfile.java:126

```
123 catch (Exception e)
124 {
125 s.setMessage("Error deleting employee profile");
126 e.printStackTrace();
127 }
128 }
129
```

ViewProfile.java, line 226 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

ViewProfile.java, line 226 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:226

```
223 catch (Exception e)
224 {
225 s.setMessage("Error getting employee profile");
226 e.printStackTrace();
227 }
228
229 return profile;
```

RoleBasedAccessControl.java, line 344 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:344

```
341 catch (ValidationException ve)
342 {
343 System.out.println("Validation failed");
344 ve.printStackTrace();
345 setCurrentAction(s, ERROR_ACTION);
346 }
347 catch (UnauthenticatedException ue)
```

RoleBasedAccessControl.java, line 432 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 432 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:432

```
429 catch (ParameterNotFoundException pnfe)
430 {
431 System.out.println("Missing parameter");
432 pnfe.printStackTrace();
433 setCurrentAction(s, ERROR_ACTION);
434 }
435 catch (ValidationException ve)
```

FindProfile.java, line 181 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:181

```
178 catch (SQLException sqle)
179 {
180 s.setMessage("Error finding employee profile");
181 sqle.printStackTrace();
182 }
183 }
184 catch (Exception e)
```

RoleBasedAccessControl.java, line 338 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:338

```
335 catch (ParameterNotFoundException pnfe)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 338 (System Information Leak) — Low

```
336 {
337 System.out.println("Missing parameter");
338 pnfe.printStackTrace();
339 setCurrentAction(s, ERROR_ACTION);
340 }
341 catch (ValidationException ve)
```

DeleteProfile.java, line 120 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: deleteEmployeeProfile()
File: DeleteProfile.java:120

```
117 catch (SQLException sqle)
118 {
119 s.setMessage("Error deleting employee profile");
120 sqle.printStackTrace();
121 }
122 }
123 catch (Exception e)
```

Logout.java, line 71 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: Logout.java:71

```
68 catch (UnauthenticatedException ue1)
69 {
70 System.out.println("Internal server error");
71 ue1.printStackTrace();
72 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Logout.java, line 71 (System Information Leak) — Low

```
73 catch (UnauthorizedException ue2)
74 {
```

Login.java, line 216 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: Login.java:216

```
213 catch (Exception e)
214 {
215 s.setMessage("Error getting employees");
216 e.printStackTrace();
217 }
218
219 return employees;
```

ViewProfile.java, line 220 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:220

```
217 catch (SQLException sqle)
218 {
219 s.setMessage("Error getting employee profile");
220 sqle.printStackTrace();
221 }
222 }
223 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 321 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:321

```
318 }
319 catch (Exception e)
320 {
321 e.printStackTrace();
322 s.setMessage("Error updating employee profile");
323 }
324 }
```

RoleBasedAccessControl.java, line 445 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:445

```
442 {
443 s.setMessage("Login failed");
444 System.out.println("Authentication failure");
445 ue.printStackTrace();
446 }
447 catch (UnauthorizedException ue2)
448 {
```

Login.java, line 96 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Login.java, line 96 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: Login.java:96

```
93 catch (UnauthorizedException ue2)
94 {
95 System.out.println("Internal server error");
96 ue2.printStackTrace();
97 }
98 }
99 else
```

RoleBasedAccessControl.java, line 458 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:458

```
455 {
456 // All other errors send the user to the generic error page
457 System.out.println("handleRequest() error");
458 e.printStackTrace();
459 setCurrentAction(s, ERROR_ACTION);
460 }
461 }
```

ListStaff.java, line 169 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

ListStaff.java, line 169 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:169

```
166 catch (Exception e)
167 {
168 s.setMessage("Error getting employees");
169 e.printStackTrace();
170 }
171
172 return employees;
```

DeleteProfile.java, line 81 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: DeleteProfile.java:81

```
78 catch (UnauthenticatedException ue1)
79 {
80 System.out.println("Internal server error");
81 ue1.printStackTrace();
82 }
83 catch (UnauthorizedException ue2)
84 {
```

EditProfile.java, line 186 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: EditProfile.java:186

```
183 catch (SQLException sqle)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

EditProfile.java, line 186 (System Information Leak) — Low

```
184 {
185 s.setMessage("Error getting employee profile");
186 sqle.printStackTrace();
187 }
188 }
189 catch (Exception e)
```

UpdateProfile.java, line 315 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:315

```
312 }
313 catch (SQLException sqle)
314 {
315 sqle.printStackTrace();
316 s.setMessage("Error updating employee profile");
317 }
318 }
```

UpdateProfile.java, line 299 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createEmployeeProfile()
File: UpdateProfile.java:299

```
296 }
297 catch (SQLException sqle)
298 {
299 sqle.printStackTrace();
300 s.setMessage("Error updating employee profile");
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 299 (System Information Leak) — Low

```
301 }
302
```

ListStaff.java, line 121 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: ListStaff.java:121

```
118 catch (Exception e)
119 {
120 s.setMessage("Error getting employees");
121 e.printStackTrace();
122 }
123
124 return employees;
```

Logout.java, line 76 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: Logout.java:76

```
73 catch (UnauthorizedException ue2)
74 {
75 System.out.println("Internal server error");
76 ue2.printStackTrace();
77 }
78
79 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 188 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: changeEmployeeProfile()
File: UpdateProfile.java:188

```
185 catch (Exception e)
186 {
187 s.setMessage("Error updating employee profile");
188 e.printStackTrace();
189 }
190 }
191
```

RoleBasedAccessControl.java, line 358 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:358

```
355 s.setMessage("You are not authorized to perform this function");
356 System.out.println("Authorization failure");
357 setCurrentAction(s, ERROR_ACTION);
358 ue2.printStackTrace();
359 }
360 catch (Exception e)
361 {
```

UpdateProfile.java, line 130 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

UpdateProfile.java, line 130 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: UpdateProfile.java:130

```
127 catch (UnauthorizedException ue2)
128 {
129 System.out.println("Internal server error");
130 ue2.printStackTrace();
131 }
132 }
133 else
```

UpdateProfile.java, line 257 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getNextUID()
File: UpdateProfile.java:257

```
254 }
255 catch (SQLException sqle)
256 {
257 sqle.printStackTrace();
258 s.setMessage("Error updating employee profile");
259 }
260 catch (ClassNotFoundException e)
```

DeleteProfile.java, line 86 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 86 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: DeleteProfile.java:86

```
83 catch (UnauthorizedException ue2)
84 {
85 System.out.println("Internal server error");
86 ue2.printStackTrace();
87 }
88 }
89 else
```

EditProfile.java, line 192 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: EditProfile.java:192

```
189 catch (Exception e)
190 {
191 s.setMessage("Error getting employee profile");
192 e.printStackTrace();
193 }
194
195 return profile;
```

DeleteProfile.java, line 149 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: deleteEmployeeProfile_BACKUP()
File: DeleteProfile.java:149

```
146 catch (SQLException sqle)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

DeleteProfile.java, line 149 (System Information Leak) — Low

```
147 {
148 s.setMessage("Error deleting employee profile");
149 sqle.printStackTrace();
150 }
151 }
152 catch (Exception e)
```

FindProfile.java, line 187 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: findEmployeeProfile()
File: FindProfile.java:187

```
184 catch (Exception e)
185 {
186 s.setMessage("Error finding employee profile");
187 e.printStackTrace();
188 }
189
190 return profile;
```

Login.java, line 169 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: login()
File: Login.java:169

```
166 catch (Exception e)
167 {
168 s.setMessage("Error logging in");
169 e.printStackTrace();
170 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

Login.java, line 169 (System Information Leak) — Low

```
171
172 //System.out.println("Lesson login result: " + authenticated);
```

RoleBasedAccessControl.java, line 364 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:364

```
361 {
362 // All other errors send the user to the generic error page
363 System.out.println("handleRequest() error");
364 e.printStackTrace();
365 setCurrentAction(s, ERROR_ACTION);
366 }
367
```

ViewProfile.java, line 166 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:166

```
163 catch (Exception e)
164 {
165 s.setMessage("Error getting employee profile");
166 e.printStackTrace();
167 }
168
169 return profile;
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

EditProfile.java, line 125 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: EditProfile.java:125

```
122 catch (SQLException sqle)
123 {
124 s.setMessage("Error getting employee profile");
125 sqle.printStackTrace();
126 }
127 }
128 catch (Exception e)
```

Login.java, line 210 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: Login.java:210

```
207 catch (SQLException sqle)
208 {
209 s.setMessage("Error getting employees");
210 sqle.printStackTrace();
211 }
212 }
213 catch (Exception e)
```

RoleBasedAccessControl.java, line 351 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 351 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: RoleBasedAccessControl.java:351

```
348 {
349 s.setMessage("Login failed");
350 System.out.println("Authentication failure");
351 ue.printStackTrace();
352 }
353 catch (UnauthorizedException ue2)
354 {
```

ViewProfile.java, line 160 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:160

```
157 catch (SQLException sqle)
158 {
159 s.setMessage("Error getting employee profile");
160 sqle.printStackTrace();
161 }
162 }
163 catch (Exception e)
```

FindProfile.java, line 89 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 89 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: FindProfile.java:89

```
86 catch (UnauthorizedException ue2)
87 {
88 System.out.println("Internal server error");
89 ue2.printStackTrace();
90 }
91 }
92 }
```

DeleteProfile.java, line 155 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: deleteEmployeeProfile_BACKUP()
File: DeleteProfile.java:155

```
152 catch (Exception e)
153 {
154 s.setMessage("Error deleting employee profile");
155 e.printStackTrace();
156 }
157 }
158
```

ListStaff.java, line 163 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:163

```
160 catch (SQLException sqle)
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

ListStaff.java, line 163 (System Information Leak) — Low

```
161 {
162 s.setMessage("Error getting employees");
163 sqle.printStackTrace();
164 }
165 }
166 catch (Exception e)
```

UpdateProfile.java, line 125 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: UpdateProfile.java:125

```
122 catch (UnauthenticatedException ue1)
123 {
124 System.out.println("Internal server error");
125 ue1.printStackTrace();
126 }
127 catch (UnauthorizedException ue2)
128 {
```

FindProfile.java, line 84 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: FindProfile.java:84

```
81 catch (UnauthenticatedException ue1)
82 {
83 System.out.println("Internal server error");
84 ue1.printStackTrace();
85 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

FindProfile.java, line 84 (System Information Leak) — Low

```
86 catch (UnauthorizedException ue2)
87 {
```

RoleBasedAccessControl.java, line 438 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:438

```
435 catch (ValidationException ve)
436 {
437 System.out.println("Validation failed");
438 ve.printStackTrace();
439 setCurrentAction(s, ERROR_ACTION);
440 }
441 catch (UnauthenticatedException ue)
```

UpdateProfile.java, line 230 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: changeEmployeeProfile_BACKUP()
File: UpdateProfile.java:230

```
227 catch (SQLException sqle)
228 {
229 s.setMessage("Error updating employee profile");
230 sqle.printStackTrace();
231 }
232
233 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.RoleBasedAccessControl

RoleBasedAccessControl.java, line 452 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest_BACKUP()
File: RoleBasedAccessControl.java:452

```
449 s.setMessage("You are not authorized to perform this function");
450 System.out.println("Authorization failure");
451 setCurrentAction(s, ERROR_ACTION);
452 ue2.printStackTrace();
453 }
454 catch (Exception e)
455 {
```

Login.java, line 91 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: Login.java:91

```
88 catch (UnauthenticatedException ue1)
89 {
90 System.out.println("Internal server error");
91 ue1.printStackTrace();
92 }
93 catch (UnauthorizedException ue2)
94 {
```

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 100 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 100 (System Information Leak) — Low

Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: Login.java:100

```
97  catch (UnauthorizedException ue2)
98  {
99  System.out.println("Internal server error");
100 ue2.printStackTrace();
101 }
102 }
103 else
```

Login.java, line 95 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: Login.java:95

```
92  catch (UnauthenticatedException ue1)
93  {
94  System.out.println("Internal server error");
95  ue1.printStackTrace();
96  }
97  catch (UnauthorizedException ue2)
98  {
```

Login.java, line 258 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 258 (System Information Leak) — Low

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: Login.java:258

```
255 catch (Exception e)
256 {
257 s.setMessage("Error getting employees");
258 e.printStackTrace();
259 }
260
261 return employees;
```

ViewProfile.java, line 212 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:212

```
209 catch (Exception e)
210 {
211 s.setMessage("Error getting employee profile");
212 e.printStackTrace();
213 }
214
215 return profile;
```

Login.java, line 168 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: login()
File: Login.java:168

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 168 (System Information Leak) — Low

```
165 catch (Exception e)
166 {
167 s.setMessage("Error logging in");
168 e.printStackTrace();
169 }
170
171 //System.out.println("Lesson login result: " + authenticated);
```

SQLInjection.java, line 348 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SQLInjection.java:348

```
345 catch (ValidationException ve)
346 {
347 System.out.println("Validation failed");
348 ve.printStackTrace();
349 setCurrentAction(s, ERROR_ACTION);
350 }
351 catch (UnauthenticatedException ue)
```

ViewProfile.java, line 152 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:152

```
149 catch (Exception e)
150 {
151 s.setMessage("Error getting employee profile");
152 e.printStackTrace();
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 152 (System Information Leak) — Low

```
153 }
154
155 return profile;
```

ListStaff.java, line 115 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: ListStaff.java:115

```
112 catch (SQLException sqle)
113 {
114 s.setMessage("Error getting employees");
115 sqle.printStackTrace();
116 }
117 }
118 catch (Exception e)
```

Login.java, line 211 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: login_BACKUP()
File: Login.java:211

```
208 catch (Exception e)
209 {
210 s.setMessage("Error logging in");
211 e.printStackTrace();
212 }
213
214 //System.out.println("Lesson login result: " + authenticated);
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 252 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: Login.java:252

```
249 catch (SQLException sqle)
250 {
251 s.setMessage("Error getting employees");
252 sqle.printStackTrace();
253 }
254 }
255 catch (Exception e)
```

SQLInjection.java, line 367 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SQLInjection.java:367

```
364 {
365 // All other errors send the user to the generic error page
366 System.out.println("handleRequest() error");
367 e.printStackTrace();
368 setCurrentAction(s, ERROR_ACTION);
369 }
370 }
```

Login.java, line 162 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

Login.java, line 162 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: login()
File: Login.java:162

```
159 catch (SQLException sqle)
160 {
161 s.setMessage("Error logging in");
162 sqle.printStackTrace();
163 }
164 }
165 catch (Exception e)
```

ListStaff.java, line 163 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:163

```
160 catch (SQLException sqle)
161 {
162 s.setMessage("Error getting employees");
163 sqle.printStackTrace();
164 }
165 }
166 catch (Exception e)
```

ViewProfile.java, line 146 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 146 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile()
File: ViewProfile.java:146

```
143 catch (SQLException sqle)
144 {
145 s.setMessage("Error getting employee profile");
146 sqle.printStackTrace();
147 }
148 }
149 catch (Exception e)
```

SQLInjection.java, line 355 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SQLInjection.java:355

```
352 {
353 s.setMessage("Login failed");
354 System.out.println("Authentication failure");
355 ue.printStackTrace();
356 }
357 catch (UnauthorizedException ue2)
358 {
```

SQLInjection.java, line 361 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SQLInjection.java:361

```
358 {
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

SQLInjection.java, line 361 (System Information Leak) — Low

```
359 s.setMessage("You are not authorized to perform this function");
360 System.out.println("Authorization failure");
361 ue2.printStackTrace();
362 }
363 catch (Exception e)
364 {
```

SQLInjection.java, line 342 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: handleRequest()
File: SQLInjection.java:342

```
339 catch (ParameterNotFoundException pnfe)
340 {
341 System.out.println("Missing parameter");
342 pnfe.printStackTrace();
343 setCurrentAction(s, ERROR_ACTION);
344 }
345 catch (ValidationException ve)
```

ListStaff.java, line 169 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees_BACKUP()
File: ListStaff.java:169

```
166 catch (Exception e)
167 {
168 s.setMessage("Error getting employees");
169 e.printStackTrace();
170 }
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ListStaff.java, line 169 (System Information Leak) — Low

```
171
172 return employees;
```

Login.java, line 205 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: login_BACKUP()
File: Login.java:205

```
202 catch (SQLException sqle)
203 {
204 s.setMessage("Error logging in");
205 sqle.printStackTrace();
206 }
207 }
208 catch (Exception e)
```

ListStaff.java, line 121 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getAllEmployees()
File: ListStaff.java:121

```
118 catch (Exception e)
119 {
120 s.setMessage("Error getting employees");
121 e.printStackTrace();
122 }
123
124 return employees;
```

System Information Leak — Low

Package: org.owasp.webgoat.lessons.SQLInjection

ViewProfile.java, line 206 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getEmployeeProfile_BACKUP()
File: ViewProfile.java:206

```
203 catch (SQLException sqle)
204 {
205 s.setMessage("Error getting employee profile");
206 sqle.printStackTrace();
207 }
208 }
209 catch (Exception e)
```

Package: org.owasp.webgoat.lessons.admin

SummaryReportCardScreen.java, line 101 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: SummaryReportCardScreen.java:101

```
98 }
99 catch (Exception e)
100 {
101 e.printStackTrace();
102 }
103
104 ec.addElement(new Center().addElement(makeSummary(s)));
```

RefreshDBScreen.java, line 169 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation

System Information Leak — Low

Package: org.owasp.webgoat.lessons.admin

RefreshDBScreen.java, line 169 (System Information Leak) — Low

Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: refreshDB()
File: RefreshDBScreen.java:169

```
166 {
167 s.setMessage("Error refreshing database "
168 + this.getClass().getName());
169 e.printStackTrace();
170 }
171 }
172 }
```

ViewDatabase.java, line 105 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: ViewDatabase.java:105

```
102 catch (Exception e)
103 {
104 s.setMessage("Error generating " + this.getClass().getName());
105 e.printStackTrace();
106 }
107
108 return (ec);
```

RefreshDBScreen.java, line 99 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

System Information Leak — Low

Package: org.owasp.webgoat.lessons.admin

RefreshDBScreen.java, line 99 (System Information Leak) — Low

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: RefreshDBScreen.java:99

```
96  catch (Exception e)
97  {
98  s.setMessage("Error generating " + this.getClass().getName());
99  e.printStackTrace();
100 }
101
102 return (ec);
```

UserAdminScreen.java, line 88 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: UserAdminScreen.java:88

```
85  catch (Exception e)
86  {
87  s.setMessage("Error generating " + this.getClass().getName());
88  e.printStackTrace();
89  }
90
91  return (ec);
```

ProductsAdminScreen.java, line 88 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createContent()
File: ProductsAdminScreen.java:88

System Information Leak — Low

Package: org.owasp.webgoat.lessons.admin

ProductsAdminScreen.java, line 88 (System Information Leak) — Low

```
85 catch (Exception e)
86 {
87 s.setMessage("Error generating " + this.getClass().getName());
88 e.printStackTrace();
89 }
90
91 return (ec);
```

Package: org.owasp.webgoat.session

CreateDB.java, line 165 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createMessageTable()
File: CreateDB.java:165

```
162 catch (SQLException e)
163 {
164 System.out.println("Error creating message database");
165 e.printStackTrace();
166 }
167 }
168
```

CreateDB.java, line 406 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createWeatherDataTable()
File: CreateDB.java:406

```
403 catch (SQLException e)
404 {
```

System Information Leak — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 406 (System Information Leak) — Low

```
405 System.out.println("Error creating weather database");
406 e.printStackTrace();
407 }
408
409 // Populate it
```

CreateDB.java, line 203 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createProductTable()
File: CreateDB.java:203

```
200 catch (SQLException e)
201 {
202 System.out.println("Error creating product database");
203 e.printStackTrace();
204 }
205
206 // Populate
```

WebSession.java, line 448 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:448

```
445 catch ( Exception e )
446 {
447 System.out.println( "Couldn't open database: check web.xml database parameters" );
448 e.printStackTrace();
449 }
450
```

System Information Leak — Low

Package: org.owasp.webgoat.session

WebSession.java, line 448 (System Information Leak) — Low

```
451 return null;
```

WebgoatProperties.java, line 122 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: main()
File: WebgoatProperties.java:122

```
119 catch (IOException e)
120 {
121 System.out.println("Error loading properties");
122 e.printStackTrace();
123 }
124 System.out.println(properties.getProperty("CommandInjection.category"));
125 }
```

ErrorScreen.java, line 258 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: getStackTrace()
File: ErrorScreen.java:258

```
255 {
256 ByteArrayOutputStream bytes = new ByteArrayOutputStream();
257 PrintWriter writer = new PrintWriter( bytes, true );
258 t.printStackTrace( writer );
259
260 return ( bytes.toString() );
261 }
```

System Information Leak — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 307 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createUserDataTable()
File: CreateDB.java:307

```
304 catch (SQLException e)
305 {
306 System.out.println("Error creating user database");
307 e.printStackTrace();
308 }
309
310 // Populate it
```

DatabaseUtilities.java, line 110 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:110

```
107 }
108 catch (Exception e)
109 {
110 e.printStackTrace();
111 return null;
112 }
113 }
```

CreateDB.java, line 126 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

System Information Leak — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 126 (System Information Leak) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: main()
File: CreateDB.java:126

```
123 }
124 catch (SQLException sqle)
125 {
126 sqle.printStackTrace();
127 }
128 }
129
```

CreateDB.java, line 101 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: main()
File: CreateDB.java:101

```
98 }
99 catch (SQLException sqle)
100 {
101 sqle.printStackTrace();
102 }
103
104 /**
```

Course.java, line 72 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

System Information Leak — Low

Package: org.owasp.webgoat.session

Course.java, line 72 (System Information Leak) — Low

Sink: printStackTrace()
Enclosing Method: Course()
File: Course.java:72

```
69 catch (IOException e)
70 {
71 System.out.println("Error loading WebGoat properties");
72 e.printStackTrace();
73 }
74 }
75
```

CreateDB.java, line 255 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createUserAdminTable()
File: CreateDB.java:255

```
252 catch (SQLException e)
253 {
254 System.out.println("Error creating user admin database");
255 e.printStackTrace();
256 }
257
258 // Populate
```

CreateDB.java, line 78 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: main()
File: CreateDB.java:78

```
75 catch (Exception e)
```

System Information Leak — Low

Package: org.owasp.webgoat.session

CreateDB.java, line 78 (System Information Leak) — Low

```
76 {
77 System.out.println("Driver Manager failed!");
78 e.printStackTrace();
79 }
80
81 /**
```

CreateDB.java, line 63 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: main()
File: CreateDB.java:63

```
60 catch (Exception e)
61 {
62 System.out.println("Failed to load DB driver");
63 e.printStackTrace();
64 }
65
66 try
```

LessonTracker.java, line 254 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: load()
File: LessonTracker.java:254

```
251 catch (Exception e)
252 {
253 System.out.println("Failed to load lesson state for " + screen);
254 e.printStackTrace();
255 }
```

System Information Leak — Low

Package: org.owasp.webgoat.session

LessonTracker.java, line 254 (System Information Leak) — Low

```
256 finally
257 {
```

CreateDB.java, line 365 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: createLoginTable()
File: CreateDB.java:365

```
362 catch (SQLException e)
363 {
364 System.out.println("Error creating user database");
365 e.printStackTrace();
366 }
367
368 }
```

Package: org.owasp.webgoat.util

Interceptor.java, line 115 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doFilter()
File: Interceptor.java:115

```
112 }
113 catch (IOException e)
114 {
115 e.printStackTrace();
116 }
117 finally
118 {
```

System Information Leak — Low

Package: org.owasp.webgoat.util

Interceptor.java, line 110 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: printStackTrace()
Enclosing Method: doFilter()
File: Interceptor.java:110

```
107 }
108 catch (UnknownHostException e)
109 {
110 e.printStackTrace();
111
112 }
113 catch (IOException e)
```

System Information Leak: External (3 issues)

Abstract

Revealing system data or debugging information helps an adversary learn about the system and form a plan of attack.

Explanation

An external information leak occurs when system data or debugging information leaves the program to a remote machine via a socket or network connection. External leaks can help an attacker by revealing specific data about operating systems, full pathnames, the existence of usernames, or locations of configuration files, and are more serious than internal information leaks which are more difficult for an attacker to access.

Example 1: The following code leaks Exception information in the HTTP response:

```
protected void doPost (HttpServletRequest req, HttpServletResponse res) throws
IOException {
    ...
    PrintWriter out = res.getWriter();
    try {
        ...
    } catch (Exception e) {
      out.println(e.getMessage());
    }
}
```

This information can be exposed to a remote user. In some cases the error message tells the attacker precisely what sort of an attack the system is vulnerable to. For example, a database error message can reveal that the application is vulnerable to a SQL injection attack. Other error messages can reveal more oblique clues about the system. In the example above, the leaked information could imply information about the type of operating system, the applications installed on the system, and the amount of care that the administrators have put into configuring the program.

In the mobile world, information leaks are also a concern. The essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which is why application authors need to be careful about what information they include in messages addressed to other applications running on the device.

Example 2: The code below broadcasts the stack trace of a caught exception to all the registered Android receivers.

```
...
try {
  ...
} catch (Exception e) {
    String exception = Log.getStackTraceString(e);
    Intent i = new Intent();
    i.setAction("SEND_EXCEPTION");
    i.putExtra("exception", exception);
    view.getContext().sendBroadcast(i);
}
...
```

Here is another scenario specific to the mobile world. Most mobile devices now implement a Near-Field Communication (NFC) protocol for quickly sharing information between devices using radio communication. It works by bringing devices to close proximity or simply having them touch each other. Even though the

communication range of NFC is limited to just a few centimeters, eavesdropping, data modification and various other types of attacks are possible, since NFC alone does not ensure secure communication.

Example 3: The Android platform provides support for NFC. The following code creates a message that gets pushed to the other device within the range.

```
...
public static final String TAG = "NfcActivity";
private static final String DATA_SPLITTER = "__:DATA:__";
private static final String MIME_TYPE = "application/my.applications.mimetype";
...
TelephonyManager tm =
(TelephonyManager)Context.getSystemService(Context.TELEPHONY_SERVICE);
String VERSION = tm.getDeviceSoftwareVersion();
...
NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(this);
if (nfcAdapter == null)
  return;

String text = TAG + DATA_SPLITTER + VERSION;
NdefRecord record = new NdefRecord(NdefRecord.TNF_MIME_MEDIA,
            MIME_TYPE.getBytes(), new byte[0], text.getBytes());
NdefRecord[] records = { record };
NdefMessage msg = new NdefMessage(records);
nfcAdapter.setNdefPushMessage(msg, this);
...
```

NFC Data Exchange Format (NDEF) message contains typed data, a URI, or a custom application payload. If the message contains information about the application, such as its name, MIME type, or device software version, this information could be leaked to an eavesdropper.

Recommendation

Write error messages with security in mind. In production environments, turn off detailed error information in favor of brief messages. Restrict the generation and storage of detailed output that can help administrators and programmers diagnose problems. Be careful, debugging traces can sometimes appear in non-obvious places (embedded in comments in the HTML for an error page, for example).

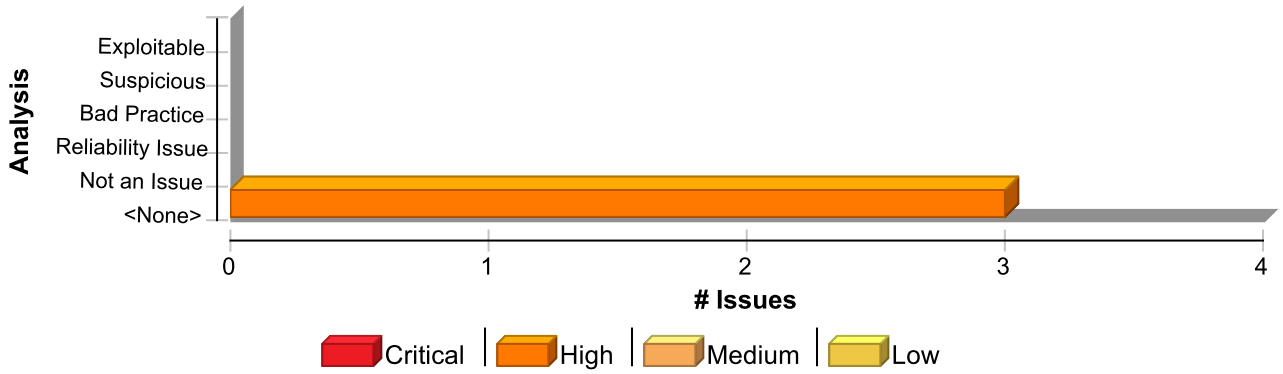
Even brief error messages that do not reveal stack traces or database dumps can potentially aid an attacker. For example, an "Access Denied" message can reveal that a file or user exists on the system. Due to this, it's advised to always keep information instead of sending it to a resource directly outside the program.

Example 4: The code below broadcasts the stack trace of a caught exception within your app only, so that it cannot be leaked to other apps on the system. There is also the added bonus that this is more efficient than globally broadcasting through the system.

```
...
try {
  ...
} catch (Exception e) {
    String exception = Log.getStackTraceString(e);
    Intent i = new Intent();
    i.setAction("SEND_EXCEPTION");
    i.putExtra("exception", exception);
    LocalBroadcastManager.getInstance(view.getContext()).sendBroadcast(i);
}
...
```

If you are concerned about leaking system data via NFC on an Android device, you could do one of the following three things. Either do not include system data in the messages pushed to other devices in range, or encrypt the payload of the message, or establish secure communication channel at a higher layer.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
System Information Leak: External	3	3	0	6
Total	**3**	**3**	**0**	**6**

System Information Leak: External — **High**

Package: /

main.jsp, line 194 (System Information Leak: External) — **High**

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.AbstractLesson.getFileText
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:471

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

System Information Leak: External — High

Package: /

main.jsp, line 194 (System Information Leak: External) — High

Sink Details

Sink: javax.servlet.jsp.JspWriter.print()
Enclosing Method: _jspService()
File: main.jsp:194
Taint Flags: PROPERTY, SYSTEMINFO

```
191 out.println(printCookies);
192 }
193 }%>
194 <div id="lessonPlans" style="visibility:hidden; height:1px; position:absolute; left:260px; top:130px; width:425px; z-index:105;"><
%=currentLesson.getLessonPlan(webSession) %>
195 <br/>
196 <br/>
197 <a href="javascript:toggle('lessonPlans')" target="_top" onclick="MM_nbGroup('down','group1','plans','',1)">Close this Window</a>
```

Package: org.owasp.webgoat.lessons

XMLInjection.java, line 119 (System Information Leak: External) — High

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.XMLInjection.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/XMLInjection.java:108

```
105 if (s.getParser().getRawParameter(ACCOUNTID, "").equals(
106 "836239"))
107 {
108 String lineSep = System.getProperty("line.separator");
109 String xmlStr = "<root>" + lineSep
110 + "<reward>WebGoat t-shirt 20 Pts</reward>"
111 + lineSep
```

Sink Details

Sink: java.io.PrintWriter.print()
Enclosing Method: handleRequest()
File: XMLInjection.java:119
Taint Flags: PROPERTY, SYSTEMINFO

```
116 s.getResponse().setHeader("Cache-Control", "no-cache");
```

System Information Leak: External — High

Package: org.owasp.webgoat.lessons

XMLInjection.java, line 119 (System Information Leak: External) — High

```
117 PrintWriter out = new PrintWriter(s.getResponse()
118 .getOutputStream());
119 out.print(xmlStr);
120 out.flush();
121 out.close();
122 return;
```

JSONInjection.java, line 90 (System Information Leak: External) — High

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.JSONInjection.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/JSONInjection.java:73

```
70 {
71 if (s.getParser().getRawParameter("from", "").equals("ajax"))
72 {
73 String lineSep = System.getProperty("line.separator");
74 String jsonStr = "{"
75 + lineSep
76 + "\"From\": \"Boston\","
```

Sink Details

Sink: java.io.PrintWriter.print()
Enclosing Method: handleRequest()
File: JSONInjection.java:90
Taint Flags: PROPERTY, SYSTEMINFO

```
87 s.getResponse().setHeader("Cache-Control", "no-cache");
88 PrintWriter out = new PrintWriter(s.getResponse()
89 .getOutputStream());
90 out.print(jsonStr);
91 out.flush();
92 out.close();
93 return;
```

System Information Leak: HTML Comment in JSP (3 issues)

Abstract

Any information revealed in an HTML comment might help an adversary learn about the system and form a plan of attack.

Explanation

HTML comments provide an attacker with an easy source of information about a dynamically generated web page.

Example 1:

```
<!-- TBD: this needs a security audit -->
<form method="POST" action="recalcOrbit">
...
```

Even comments that seem innocuous may be useful to someone trying to understand the way the system is built.

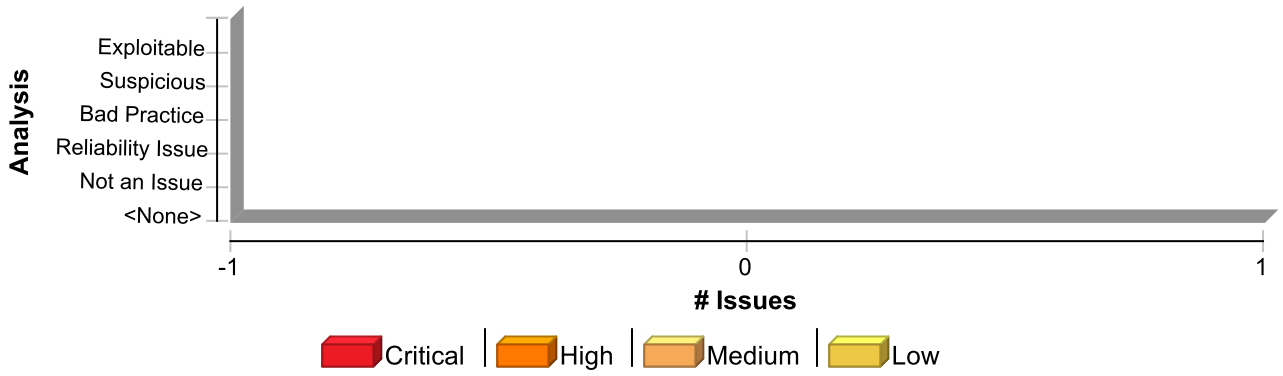
Recommendation

Replace HTML comments with JSP comments (which will not be transmitted to the user).

Example 2: The previous example is rewritten to use JSP comments, which will not be displayed to the user.

```
<%-- TBD: this needs a security audit --%>
<form method="POST" action="recalcOrbit">
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
System Information Leak: HTML Comment in JSP	3	3	0	6
Total	**3**	**3**	**0**	**6**

System Information Leak: HTML Comment in JSP — Low

Package: WebContent.lessons.CrossSiteScripting

ViewProfile.jsp, line 84 (System Information Leak: HTML Comment in JSP) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.jsp:84

```
81 Comments:
82 </TD>
83 <TD>
84 <!-- Encode data that might contain HTML content to protect against XSS -->
85
86 <%=webSession.htmlEncode(employee.getPersonalDescription())%>
87 </TD>
```

ViewProfile.jsp, line 31 (System Information Leak: HTML Comment in JSP) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Comment
File: ViewProfile.jsp:31

```
28 Street:
29 </TD>
30 <TD>
31 <!-- STAGE 4 - FIX Note that the description value below gets encoded and address1 here is not -->
32
33 <%=employee.getAddress1()%>
34 </TD>
```

ViewProfile.jsp, line 1 (System Information Leak: HTML Comment in JSP) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

System Information Leak: HTML Comment in JSP	Low

Package: WebContent.lessons.CrossSiteScripting

ViewProfile.jsp, line 1 (System Information Leak: HTML Comment in JSP)	Low

Sink Details

Sink: Comment
File: ViewProfile.jsp:1

```
1 <!--
2 STAGE 4 FIXES Look for the <-- STAGE 4 - FIX
3 -->
4 <%@ page contentType="text/html; charset=ISO-8859-1" language="java"
5  import="org.owasp.webgoat.session.*, org.owasp.webgoat.lessons.CrossSiteScripting.CrossSiteScripting" errorPage="" %>
6 <%
7 WebSession webSession = ((WebSession)session.getAttribute("websession"));
```

System Information Leak: Incomplete Servlet Error Handling (3 issues)

Abstract

If a Servlet fails to catch all exceptions, it might reveal debugging information that will help an adversary form a plan of attack.

Explanation

When a Servlet throws an exception, the default error response the Servlet container sends back to the user typically includes debugging information. This information is of great value to an attacker. For example, a stack trace might show the attacker a malformed SQL query string, the type of database being used, and the version of the application container. This information enables the attacker to target known vulnerabilities in these components.

Example 1: In the following method a DNS lookup failure will cause the Servlet to throw an exception.

```
protected void doPost (HttpServletRequest req,
                       HttpServletResponse res)
                throws IOException {
    String ip = req.getRemoteAddr();
    InetAddress addr = InetAddress.getByName(ip);
    ...
    out.println("hello " + addr.getHostName());
}
```

Example 2: The following method will throw a `NullPointerException` if the parameter "name" is not part of the request.

```
protected void doPost (HttpServletRequest req,
                       HttpServletResponse res)
                throws IOException {
    String name = getParameter("name");
    ...
    out.println("hello " + name.trim());
}
```

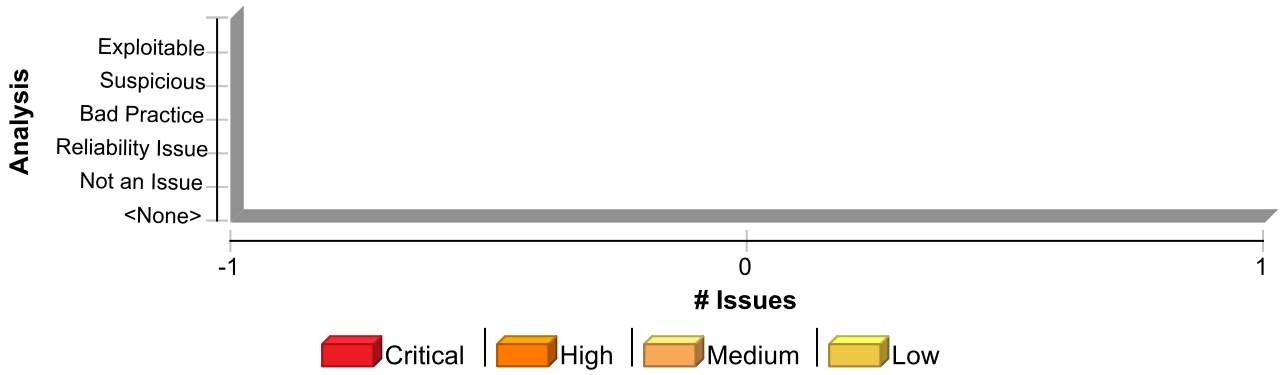
Recommendation

All top-level Servlet methods should catch `Throwable`, thereby minimizing the chance that the Servlet's error response mechanism is invoked.

Example 3: The method from Example 1 should be rewritten as follows:

```
protected void doPost (HttpServletRequest req,
                       HttpServletResponse res) {
      try {
          String ip = req.getRemoteAddr();
          InetAddress addr = InetAddress.getByName(ip);
      ...
          out.println("hello " + addr.getHostName());
      }catch (Throwable t) {
          logger.error("caught throwable at top level", t);
      }
  }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
System Information Leak: Incomplete Servlet Error Handling	3	3	0	6
Total	**3**	**3**	**0**	**6**

System Information Leak: Incomplete Servlet Error Handling — Low

Package: org.owasp.webgoat

HammerHead.java, line 119 (System Information Leak: Incomplete Servlet Error Handling) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: doPost
Enclosing Method: doPost()
File: HammerHead.java:119

```
116 * @exception ServletException
117 * Description of the Exception
118 */
119 public void doPost(HttpServletRequest request, HttpServletResponse response)
120 throws IOException, ServletException
121 {
122 Screen screen = null;
```

LessonSource.java, line 67 (System Information Leak: Incomplete Servlet Error Handling) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

System Information Leak: Incomplete Servlet Error Handling — Low

Package: org.owasp.webgoat

LessonSource.java, line 67 (System Information Leak: Incomplete Servlet Error Handling) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: doPost
Enclosing Method: doPost()
File: LessonSource.java:67

```
64 * @exception IOException Description of the Exception
65 * @exception ServletException Description of the Exception
66 */
67 public void doPost(HttpServletRequest request, HttpServletResponse response)
68 throws IOException, ServletException
69 {
70 String source = null;
```

Package: org.owasp.webgoat.servlets

Controller.java, line 60 (System Information Leak: Incomplete Servlet Error Handling) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Structural)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: Function: doPost
Enclosing Method: doPost()
File: Controller.java:60

```
57 undefined
58 undefined
59 undefined
60 undefined
61 undefined
62 undefined
63 undefined
```

System Information Leak: Internal (62 issues)

Abstract

Revealing system data or debugging information helps an adversary learn about the system and form a plan of attack.

Explanation

An internal information leak occurs when system data or debugging information is sent to a local file, console, or screen via printing or logging.

Example 1: The following code prints an exception to the standard error stream:

```
try {
    ...
} catch (Exception e) {
    e.printStackTrace();
}
```

Depending upon the system configuration, this information can be dumped to a console, written to a log file, or exposed to a user. In some cases the error message tells the attacker precisely what sort of an attack the system is vulnerable to. For example, a database error message can reveal that the application is vulnerable to a SQL injection attack. Other error messages can reveal more oblique clues about the system. In the example above, the leaked information could imply information about the type of operating system, the applications installed on the system, and the amount of care that the administrators have put into configuring the program.

In the mobile world, information leaks are also a concern.

Example 2: The code below logs the stack trace of a caught exception on the Android platform.

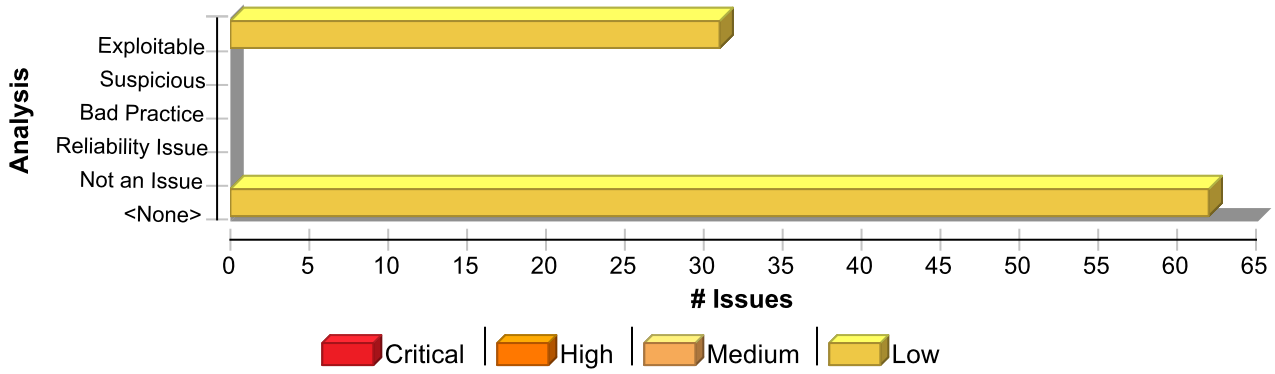
```
...
try {
  ...
} catch (Exception e) {
    Log.e(TAG, Log.getStackTraceString(e));
}
...
```

Recommendation

Write error messages with security in mind. In production environments, turn off detailed error information in favor of brief messages. Restrict the generation and storage of detailed output that can help administrators and programmers diagnose problems. Be careful, debugging traces can sometimes appear in non-obvious places (embedded in comments in the HTML for an error page, for example).

Even brief error messages that do not reveal stack traces or database dumps can potentially aid an attacker. For example, an "Access Denied" message can reveal that a file or user exists on the system.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
System Information Leak: Internal	62	62	0	124
Total	**62**	**62**	**0**	**124**

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 478 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.AbstractLesson.getFileText
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:478

```
475  }
476  catch (Exception e)
477  {
478  System.out.println(e);
479  e.printStackTrace();
480  }
481
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: getFileText()
File: AbstractLesson.java:478
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
475  }
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 478 (System Information Leak: Internal) — Low

```
476 catch (Exception e)
477 {
478 System.out.println(e);
479 e.printStackTrace();
480 }
481
```

Challenge2Screen.java, line 434 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.lessons.AbstractLesson.getFileText
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:471

```
468 {
469 sb.append(pad(++count) + " ");
470 }
471 sb.append(line + System.getProperty("line.separator"));
472 }
473
474 reader.close();
```

Sink Details

Sink: java.io.Writer.write()
Enclosing Method: resetWebPage()
File: Challenge2Screen.java:434
Taint Flags: PROPERTY, SYSTEMINFO

```
431 // replace the defaced text with the original
432 File usersFile = new File(defacedpath);
433 FileWriter fw = new FileWriter(usersFile);
434 fw.write(getFileText(new BufferedReader(new FileReader(
435 masterFilePath)), false));
436 fw.close();
437 // System.out.println("webgoat_guest replaced: " + getFileText( new BufferedReader( new FileReader( defacedpath ) ), false ) );
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 285 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletContext.getRealPath()
From: org.owasp.webgoat.lessons.CommandInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/CommandInjection.java:152

```
149  illegalCommand = false;
150  }
151  }
152  File safeDir = new File(s.getContext().getRealPath("/lesson_plans"));
153
154  ec
155  .addElement(new StringElement(
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: exec()
File: CommandInjection.java:285
Taint Flags: SYSTEMINFO, TAINTED_PATH

```
282  */
283  private String exec(WebSession s, String[] command)
284  {
285  System.out.println("Executing OS command: " + Arrays.asList(command));
286  ExecResults er = Exec.execSimple(command);
287  if (!er.getError())
288  {
```

AbstractLesson.java, line 1044 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 1044 (System Information Leak: Internal) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.AbstractLesson.readFromURL
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:1044

```
1041  }
1042  catch (Exception e)
1043  {
1044  System.out.println(e);
1045  e.printStackTrace();
1046  }
1047
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: readFromURL()
File: AbstractLesson.java:1044
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
1041  }
1042  catch (Exception e)
1043  {
1044  System.out.println(e);
1045  e.printStackTrace();
1046  }
1047
```

SqlStringInjection.java, line 316 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.SqlStringInjection.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/SqlStringInjection.java:316

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

SqlStringInjection.java, line 316 (System Information Leak: Internal) — Low

```
313  }
314  catch (Exception e)
315  {
316  System.out.println("Exception caught: " + e);
317  e.printStackTrace(System.out);
318  }
319  }
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: handleRequest()
File: SqlStringInjection.java:316
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
313  }
314  catch (Exception e)
315  {
316  System.out.println("Exception caught: " + e);
317  e.printStackTrace(System.out);
318  }
319  }
```

ThreadSafetyProblem.java, line 216 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.ThreadSafetyProblem.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/ThreadSafetyProblem.java:216

```
213  }
214  catch (Exception e)
215  {
216  System.out.println("Exception caught: " + e);
217  e.printStackTrace(System.out);
218  }
219  }
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

ThreadSafetyProblem.java, line 216 (System Information Leak: Internal) — Low

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: handleRequest()
File: ThreadSafetyProblem.java:216
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
213 }
214 catch (Exception e)
215 {
216 System.out.println("Exception caught: " + e);
217 e.printStackTrace(System.out);
218 }
219 }
```

BlindSqlInjection.java, line 343 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.BlindSqlInjection.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/BlindSqlInjection.java:343

```
340 }
341 catch (Exception e)
342 {
343 System.out.println("Exception caught: " + e);
344 e.printStackTrace(System.out);
345 }
346 }
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: handleRequest()
File: BlindSqlInjection.java:343
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
340 }
341 catch (Exception e)
342 {
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

BlindSqlInjection.java, line 343 (System Information Leak: Internal) — Low

```
343 System.out.println("Exception caught: " + e);
344 e.printStackTrace(System.out);
345 }
346 }
```

AbstractLesson.java, line 422 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.AbstractLesson.getFileMethod
File: JavaSource/org/owasp/webgoat/lessons/AbstractLesson.java:422

```
419 }
420 catch (Exception e)
421 {
422 System.out.println(e);
423 e.printStackTrace();
424 }
425
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: getFileMethod()
File: AbstractLesson.java:422
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
419 }
420 catch (Exception e)
421 {
422 System.out.println(e);
423 e.printStackTrace();
424 }
425
```

LessonAdapter.java, line 136 (System Information Leak: Internal) — Low

Issue Details

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 136 (System Information Leak: Internal) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.LessonAdapter.createStagedContent
File: JavaSource/org/owasp/webgoat/lessons/LessonAdapter.java:136

```
133  catch (Exception e)
134  {
135  s.setMessage("Error generating " + this.getClass().getName());
136  System.out.println(e);
137  e.printStackTrace();
138  }
139
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: createStagedContent()
File: LessonAdapter.java:136
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
133  catch (Exception e)
134  {
135  s.setMessage("Error generating " + this.getClass().getName());
136  System.out.println(e);
137  e.printStackTrace();
138  }
139
```

CommandInjection.java, line 264 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletContext.getRealPath()

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 264 (System Information Leak: Internal) — Low

From: org.owasp.webgoat.lessons.CommandInjection.createContent
File: JavaSource/org/owasp/webgoat/lessons/CommandInjection.java:152

```
149  illegalCommand = false;
150  }
151  }
152  File safeDir = new File(s.getContext().getRealPath("/lesson_plans"));
153
154  ec
155  .addElement(new StringElement(
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: exec()
File: CommandInjection.java:264
Taint Flags: SYSTEMINFO, TAINTED_PATH

```
261  */
262  private String exec(WebSession s, String command)
263  {
264  System.out.println("Executing OS command: " + command);
265  ExecResults er = Exec.execSimple(command);
266  if ((command.indexOf("&") != -1 || command.indexOf(";") != -1)
267  && !er.getError())
```

AbstractLesson.java, line 960 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletContext.getRealPath()
From: org.owasp.webgoat.session.WebSession.getWebResource
File: JavaSource/org/owasp/webgoat/session/WebSession.java:719

```
716  public String getWebResource( String fileName )
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 960 (System Information Leak: Internal) — Low

```
717  {
718  // Note: doesn't work for admin path! Maybe with a ../ attack
719  return ( context.getRealPath( fileName ));
720  }
721
722  /**
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: makeSourceDump_DELETEME()
File: AbstractLesson.java:960
Taint Flags: SYSTEMINFO

```
957  }
958  catch (IOException e)
959  {
960  System.out.println("reading file EXCEPTION: " + filename);
961  s.setMessage("Could not find source file");
962  }
963
```

SqlNumericInjection.java, line 400 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read e
From: org.owasp.webgoat.lessons.SqlNumericInjection.handleRequest
File: JavaSource/org/owasp/webgoat/lessons/SqlNumericInjection.java:400

```
397  }
398  catch (Exception e)
399  {
400  System.out.println("Exception caught: " + e);
401  e.printStackTrace(System.out);
402  }
403  }
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.lessons

SqlNumericInjection.java, line 400 (System Information Leak: Internal) — Low

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: handleRequest()
File: SqlNumericInjection.java:400
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
397 }
398 catch (Exception e)
399 {
400 System.out.println("Exception caught: " + e);
401 e.printStackTrace(System.out);
402 }
403 }
```

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 95 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletContext.getRealPath()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:282

```
279 course.loadCourses( enterprise, context, "/" );
280
281 // FIXME: hack to save context for web service calls
282 DatabaseUtilities.servletContextRealPath = context.getRealPath("/");
283 System.out.println("Context Path: " + DatabaseUtilities.servletContextRealPath);
284 // FIXME: need to solve concurrency problem here -- make tables for this user
285 if ( !databaseBuilt )
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:95
Taint Flags: SYSTEMINFO

```
92 if (os.toLowerCase().indexOf("window") != -1)
93 {
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.session

DatabaseUtilities.java, line 95 (System Information Leak: Internal) — Low

```
94  dbName = dbName.concat("webgoat.mdb");
95  System.out.println("DBName: " + dbName);
96  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
97  return DriverManager
98  .getConnection("jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ="
```

WebSession.java, line 439 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletContext.getRealPath()
From: org.owasp.webgoat.session.WebSession.getDatabaseConnectionString
File: JavaSource/org/owasp/webgoat/session/WebSession.java:438

```
435  {
436  try
437  {
438  String path = context.getRealPath( "/database" ).replace( '\\', '/' );
439  System.out.println( "PATH: " + path );
440  String realConnectionString = databaseConnectionString.replaceAll( "PATH", path );
441  System.out.println( "Database Connection String: " + realConnectionString );
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:439
Taint Flags: SYSTEMINFO, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR

```
436  try
437  {
438  String path = context.getRealPath( "/database" ).replace( '\\', '/' );
439  System.out.println( "PATH: " + path );
440  String realConnectionString = databaseConnectionString.replaceAll( "PATH", path );
441  System.out.println( "Database Connection String: " + realConnectionString );
442
```

WebSession.java, line 283 (System Information Leak: Internal) — Low

Issue Details

System Information Leak: Internal — Low

Package: org.owasp.webgoat.session

WebSession.java, line 283 (System Information Leak: Internal) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletContext.getRealPath()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:282

```
279  course.loadCourses( enterprise, context, "/" );
280
281  // FIXME: hack to save context for web service calls
282  DatabaseUtilities.servletContextRealPath = context.getRealPath("/");
283  System.out.println("Context Path: " + DatabaseUtilities.servletContextRealPath);
284  // FIXME: need to solve concurrency problem here -- make tables for this user
285  if ( !databaseBuilt )
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: WebSession()
File: WebSession.java:283
Taint Flags: SYSTEMINFO

```
280
281  // FIXME: hack to save context for web service calls
282  DatabaseUtilities.servletContextRealPath = context.getRealPath("/");
283  System.out.println("Context Path: " + DatabaseUtilities.servletContextRealPath);
284  // FIXME: need to solve concurrency problem here -- make tables for this user
285  if ( !databaseBuilt )
286  {
```

WebSession.java, line 441 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

System Information Leak: Internal — Low

Package: org.owasp.webgoat.session

WebSession.java, line 441 (System Information Leak: Internal) — Low

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: javax.servlet.ServletContext.getRealPath()
From: org.owasp.webgoat.session.WebSession.getDatabaseConnectionString
File: JavaSource/org/owasp/webgoat/session/WebSession.java:438

```
435  {
436  try
437  {
438  String path = context.getRealPath( "/database" ).replace( '\\', '/' );
439  System.out.println( "PATH: " + path );
440  String realConnectionString = databaseConnectionString.replaceAll( "PATH", path );
441  System.out.println( "Database Connection String: " + realConnectionString );
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: getDatabaseConnectionString()
File: WebSession.java:441
Taint Flags: SYSTEMINFO, VALIDATED_PORTABILITY_FLAW_FILE_SEPARATOR

```
438  String path = context.getRealPath( "/database" ).replace( '\\', '/' );
439  System.out.println( "PATH: " + path );
440  String realConnectionString = databaseConnectionString.replaceAll( "PATH", path );
441  System.out.println( "Database Connection String: " + realConnectionString );
442
443  return realConnectionString;
444  }
```

Package: org.owasp.webgoat.util

Exec.java, line 509 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 509 (System Information Leak: Internal) — Low

Audit Comments

Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314  }
315  catch (IOException e1)
316  {
317  results.setThrowable(e1);
318  }
319  }
320
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:509
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
506  System.out.println("-------------------------------------------" + sep
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509  System.out.println(results);
510  System.out.println("-------------------------------------------" + sep
511  + "TEST 2: execSimple (with search)");
512  results = Exec.execSimple("netstat -r");
```

Exec.java, line 513 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 513 (System Information Leak: Internal) — Low

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314  }
315  catch (IOException e1)
316  {
317  results.setThrowable(e1);
318  }
319  }
320
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:513
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
510  System.out.println("-----------------------------------------" + sep
511  + "TEST 2: execSimple (with search)");
512  results = Exec.execSimple("netstat -r");
513  System.out.println(results);
514
515  if (results.outputContains("localhost:1031"))
516  {
```

Exec.java, line 524 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 524 (System Information Leak: Internal) — Low

File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314  }
315  catch (IOException e1)
316  {
317  results.setThrowable(e1);
318  }
319  }
320
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:524
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
521  + "TEST 3: execInput");
522  results = Exec.execInput("find \"cde\"",
523  "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524  System.out.println(results);
525  System.out.println("----------------------------------------" + sep
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
```

Exec.java, line 528 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314  }
315  catch (IOException e1)
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 528 (System Information Leak: Internal) — Low

```
316 {
317 results.setThrowable(e1);
318 }
319 }
320
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:528
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
525 System.out.println("----------------------------------------" + sep
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528 System.out.println(results);
529 System.out.println("----------------------------------------" + sep
530 + "TEST 5:execLazy");
531 results = Exec.execLazy("ping -t 127.0.0.1");
```

Exec.java, line 532 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314 }
315 catch (IOException e1)
316 {
317 results.setThrowable(e1);
318 }
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 532 (System Information Leak: Internal) — Low

```
319  }
320
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:532
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
529  System.out.println("----------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
533  System.out.println("----------------------------------------" + sep
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
```

Exec.java, line 536 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314  }
315  catch (IOException e1)
316  {
317  results.setThrowable(e1);
318  }
319  }
320
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 536 (System Information Leak: Internal) — Low

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:536
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
533 System.out.println("----------------------------------------" + sep
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536 System.out.println(results);
537 System.out.println("----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
```

Exec.java, line 540 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e1
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:317

```
314 }
315 catch (IOException e1)
316 {
317 results.setThrowable(e1);
318 }
319 }
320
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 540 (System Information Leak: Internal) — Low

File: Exec.java:540
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
537 System.out.println("----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540 System.out.println(results);
541 }
542 }
543
```

Exec.java, line 509 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430 }
431 catch (Throwable t)
432 {
433 results.setThrowable(t);
434 }
435 finally
436 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:509
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
506 System.out.println("----------------------------------------" + sep
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 509 (System Information Leak: Internal) — Low

```
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509  System.out.println(results);
510  System.out.println("----------------------------------------" + sep
511  + "TEST 2: execSimple (with search)");
512  results = Exec.execSimple("netstat -r");
```

Exec.java, line 513 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430  }
431  catch (Throwable t)
432  {
433  results.setThrowable(t);
434  }
435  finally
436  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:513
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
510  System.out.println("----------------------------------------" + sep
511  + "TEST 2: execSimple (with search)");
512  results = Exec.execSimple("netstat -r");
513  System.out.println(results);
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 513 (System Information Leak: Internal) — Low

```
514
515 if (results.outputContains("localhost:1031"))
516 {
```

Exec.java, line 524 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430 }
431 catch (Throwable t)
432 {
433 results.setThrowable(t);
434 }
435 finally
436 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:524
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
521 + "TEST 3: execInput");
522 results = Exec.execInput("find \"cde\"",
523 "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524 System.out.println(results);
525 System.out.println("----------------------------------------" + sep
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 528 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430 }
431 catch (Throwable t)
432 {
433 results.setThrowable(t);
434 }
435 finally
436 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:528
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
525 System.out.println("----------------------------------------" + sep
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528 System.out.println(results);
529 System.out.println("----------------------------------------" + sep
530 + "TEST 5:execLazy");
531 results = Exec.execLazy("ping -t 127.0.0.1");
```

Exec.java, line 532 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 532 (System Information Leak: Internal) — Low

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430  }
431  catch (Throwable t)
432  {
433  results.setThrowable(t);
434  }
435  finally
436  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:532
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
529  System.out.println("-------------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
533  System.out.println("-------------------------------------------" + sep
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
```

Exec.java, line 536 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 536 (System Information Leak: Internal) — Low

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430  }
431  catch (Throwable t)
432  {
433  results.setThrowable(t);
434  }
435  finally
436  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:536
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
533  System.out.println("----------------------------------------" + sep
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536  System.out.println(results);
537  System.out.println("----------------------------------------" + sep
538  + "TEST 7:ExecTimeout process waits for input");
539  results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
```

Exec.java, line 540 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 540 (System Information Leak: Internal) — Low

Source Details

Source: Read t
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:433

```
430 }
431 catch (Throwable t)
432 {
433 results.setThrowable(t);
434 }
435 finally
436 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:540
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
537 System.out.println("----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540 System.out.println(results);
541 }
542 }
543
```

Exec.java, line 509 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 509 (System Information Leak: Internal) — Low

File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400  }
401  catch (IOException e3)
402  {
403  results.setThrowable(e3);
404  }
405  finally
406  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:509
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
506  System.out.println("----------------------------------------" + sep
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509  System.out.println(results);
510  System.out.println("----------------------------------------" + sep
511  + "TEST 2: execSimple (with search)");
512  results = Exec.execSimple("netstat -r");
```

Exec.java, line 513 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400  }
401  catch (IOException e3)
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 513 (System Information Leak: Internal) — Low

```
402 {
403 results.setThrowable(e3);
404 }
405 finally
406 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:513
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
512 results = Exec.execSimple("netstat -r");
513 System.out.println(results);
514
515 if (results.outputContains("localhost:1031"))
516 {
```

Exec.java, line 524 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400 }
401 catch (IOException e3)
402 {
403 results.setThrowable(e3);
404 }
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 524 (System Information Leak: Internal) — Low

```
405 finally
406 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:524
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
521 + "TEST 3: execInput");
522 results = Exec.execInput("find \"cde\"",
523 "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524 System.out.println(results);
525 System.out.println("----------------------------------------" + sep
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
```

Exec.java, line 528 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400 }
401 catch (IOException e3)
402 {
403 results.setThrowable(e3);
404 }
405 finally
406 {
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 528 (System Information Leak: Internal) — Low

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:528
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
525 System.out.println("----------------------------------------" + sep
526 + "TEST 4:execTimeout");
527 results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528 System.out.println(results);
529 System.out.println("----------------------------------------" + sep
530 + "TEST 5:execLazy");
531 results = Exec.execLazy("ping -t 127.0.0.1");
```

Exec.java, line 532 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400 }
401 catch (IOException e3)
402 {
403 results.setThrowable(e3);
404 }
405 finally
406 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 532 (System Information Leak: Internal) — Low

File: Exec.java:532
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
529 System.out.println("----------------------------------------" + sep
530 + "TEST 5:execLazy");
531 results = Exec.execLazy("ping -t 127.0.0.1");
532 System.out.println(results);
533 System.out.println("----------------------------------------" + sep
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
```

Exec.java, line 536 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400 }
401 catch (IOException e3)
402 {
403 results.setThrowable(e3);
404 }
405 finally
406 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:536
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
533 System.out.println("----------------------------------------" + sep
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 536 (System Information Leak: Internal) — Low

```
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536  System.out.println(results);
537  System.out.println("----------------------------------------" + sep
538  + "TEST 7:ExecTimeout process waits for input");
539  results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
```

Exec.java, line 540 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e3
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:403

```
400  }
401  catch (IOException e3)
402  {
403  results.setThrowable(e3);
404  }
405  finally
406  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:540
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
537  System.out.println("----------------------------------------" + sep
538  + "TEST 7:ExecTimeout process waits for input");
539  results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540  System.out.println(results);
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 540 (System Information Leak: Internal) — Low

```
541 }
542 }
543
```

Exec.java, line 510 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

```
502 public static void main(String[] args)
503 {
504 ExecResults results;
505 String sep = System.getProperty("line.separator");
506 System.out.println("----------------------------------------" + sep
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:510
Taint Flags: PROPERTY, SYSTEMINFO

```
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509 System.out.println(results);
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
512 results = Exec.execSimple("netstat -r");
513 System.out.println(results);
```

Exec.java, line 506 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 506 (System Information Leak: Internal) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

```
502  public static void main(String[] args)
503  {
504  ExecResults results;
505  String sep = System.getProperty("line.separator");
506  System.out.println("----------------------------------------" + sep
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:506
Taint Flags: PROPERTY, SYSTEMINFO

```
503  {
504  ExecResults results;
505  String sep = System.getProperty("line.separator");
506  System.out.println("----------------------------------------" + sep
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509  System.out.println(results);
```

Exec.java, line 520 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 520 (System Information Leak: Internal) — Low

```
502 public static void main(String[] args)
503 {
504 ExecResults results;
505 String sep = System.getProperty("line.separator");
506 System.out.println("----------------------------------------" + sep
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:520
Taint Flags: PROPERTY, SYSTEMINFO

```
517 System.out.println("ERROR: listening on 1031");
518 }
519
520 System.out.println("----------------------------------------" + sep
521 + "TEST 3: execInput");
522 results = Exec.execInput("find \"cde\"",
523 "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
```

Exec.java, line 537 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

```
502 public static void main(String[] args)
503 {
504 ExecResults results;
505 String sep = System.getProperty("line.separator");
506 System.out.println("----------------------------------------" + sep
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 537 (System Information Leak: Internal) — Low

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:537
Taint Flags: PROPERTY, SYSTEMINFO

```
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536  System.out.println(results);
537  System.out.println("----------------------------------------" + sep
538  + "TEST 7:ExecTimeout process waits for input");
539  results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540  System.out.println(results);
```

Exec.java, line 533 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

```
502  public static void main(String[] args)
503  {
504  ExecResults results;
505  String sep = System.getProperty("line.separator");
506  System.out.println("----------------------------------------" + sep
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:533
Taint Flags: PROPERTY, SYSTEMINFO

```
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 533 (System Information Leak: Internal) — Low

```
533 System.out.println("----------------------------------------" + sep
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536 System.out.println(results);
```

Exec.java, line 509 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354 }
355 catch (IOException e2)
356 {
357 results.setThrowable(e2);
358 }
359 finally
360 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:509
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
506 System.out.println("----------------------------------------" + sep
507 + "TEST 1: execSimple");
508 results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
509 System.out.println(results);
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 509 (System Information Leak: Internal) — Low

```
512 results = Exec.execSimple("netstat -r");
```

Exec.java, line 513 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354 }
355 catch (IOException e2)
356 {
357 results.setThrowable(e2);
358 }
359 finally
360 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:513
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
510 System.out.println("----------------------------------------" + sep
511 + "TEST 2: execSimple (with search)");
512 results = Exec.execSimple("netstat -r");
513 System.out.println(results);
514
515 if (results.outputContains("localhost:1031"))
516 {
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 524 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354  }
355  catch (IOException e2)
356  {
357  results.setThrowable(e2);
358  }
359  finally
360  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:524
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
521  + "TEST 3: execInput");
522  results = Exec.execInput("find \"cde\"",
523  "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524  System.out.println(results);
525  System.out.println("----------------------------------------" + sep
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
```

Exec.java, line 528 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 528 (System Information Leak: Internal) — Low

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354  }
355  catch (IOException e2)
356  {
357  results.setThrowable(e2);
358  }
359  finally
360  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:528
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
525  System.out.println("-------------------------------------------" + sep
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528  System.out.println(results);
529  System.out.println("-------------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
```

Exec.java, line 532 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

Analysis	Exploitable
AA_Prediction	Exploitable

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 532 (System Information Leak: Internal) — Low

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354  }
355  catch (IOException e2)
356  {
357  results.setThrowable(e2);
358  }
359  finally
360  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:532
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
529  System.out.println("-----------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
533  System.out.println("-----------------------------------------" + sep
534  + "TEST 6:ExecTimeout process never outputs");
535  results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
```

Exec.java, line 536 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 536 (System Information Leak: Internal) — Low

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions
File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354 }
355 catch (IOException e2)
356 {
357 results.setThrowable(e2);
358 }
359 finally
360 {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:536
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
533 System.out.println("-----------------------------------------" + sep
534 + "TEST 6:ExecTimeout process never outputs");
535 results = Exec.execTimeout("c:/swarm-2.1.1/bin/sleep.exe 20", 5 * 1000);
536 System.out.println(results);
537 System.out.println("-----------------------------------------" + sep
538 + "TEST 7:ExecTimeout process waits for input");
539 results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
```

Exec.java, line 540 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Source Details

Source: Read e2
From: org.owasp.webgoat.util.Exec.execOptions

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 540 (System Information Leak: Internal) — Low

File: JavaSource/org/owasp/webgoat/util/Exec.java:357

```
354  }
355  catch (IOException e2)
356  {
357  results.setThrowable(e2);
358  }
359  finally
360  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:540
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
537  System.out.println("----------------------------------------" + sep
538  + "TEST 7:ExecTimeout process waits for input");
539  results = Exec.execTimeout("c:/swarm-2.1.1/bin/cat", 5 * 1000);
540  System.out.println(results);
541  }
542 }
543
```

Exec.java, line 529 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

```
502  public static void main(String[] args)
503  {
504  ExecResults results;
505  String sep = System.getProperty("line.separator");
506  System.out.println("----------------------------------------" + sep
507  + "TEST 1: execSimple");
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 529 (System Information Leak: Internal) — Low

```
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:529
Taint Flags: PROPERTY, SYSTEMINFO

```
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528  System.out.println(results);
529  System.out.println("----------------------------------------" + sep
530  + "TEST 5:execLazy");
531  results = Exec.execLazy("ping -t 127.0.0.1");
532  System.out.println(results);
```

Exec.java, line 525 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.System.getProperty()
From: org.owasp.webgoat.util.Exec.main
File: JavaSource/org/owasp/webgoat/util/Exec.java:505

```
502  public static void main(String[] args)
503  {
504  ExecResults results;
505  String sep = System.getProperty("line.separator");
506  System.out.println("----------------------------------------" + sep
507  + "TEST 1: execSimple");
508  results = Exec.execSimple("c:/swarm-2.1.1/bin/whoami.exe");
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: main()
File: Exec.java:525
Taint Flags: PROPERTY, SYSTEMINFO

```
522  results = Exec.execInput("find \"cde\"",
```

System Information Leak: Internal — Low

Package: org.owasp.webgoat.util

Exec.java, line 525 (System Information Leak: Internal) — Low

```
523  "abcdefg1\nhijklmnop\nqrstuv\nabcdefg2");
524  System.out.println(results);
525  System.out.println("----------------------------------------" + sep
526  + "TEST 4:execTimeout");
527  results = Exec.execTimeout("ping -t 127.0.0.1", 5 * 1000);
528  System.out.println(results);
```

URL: /attack

HammerHead.java, line 306 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:206
URL: /attack

```
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
209  }
```

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: log()
File: HammerHead.java:306
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
303  {
304  String output = new Date() + " | " + request.getRemoteHost() + ":"
305  + request.getRemoteAddr() + " | " + message;
306  log(output);
307  System.out.println(output);
308  }
309
```

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 306 (System Information Leak: Internal) — Low

HammerHead.java, line 306 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:206
URL: /attack

```
203 {
204 thr.printStackTrace();
205 log(request, "Could not write error screen: "
206 + thr.getMessage());
207 }
208 // System.out.println( "HH Leaving doPost: " );
209 }
```

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: log()
File: HammerHead.java:306
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
303 {
304 String output = new Date() + " | " + request.getRemoteHost() + ":"
305 + request.getRemoteAddr() + " | " + message;
306 log(output);
307 System.out.println(output);
308 }
309
```

HammerHead.java, line 306 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 306 (System Information Leak: Internal) — Low

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:206
URL: /attack

```
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
209  }
```

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: log()
File: HammerHead.java:306
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
303  {
304  String output = new Date() + " | " + request.getRemoteHost() + ":"
305  + request.getRemoteAddr() + " | " + message;
306  log(output);
307  System.out.println(output);
308  }
309
```

HammerHead.java, line 307 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:206
URL: /attack

```
203  {
```

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 307 (System Information Leak: Internal) — Low

```
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
209  }
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: log()
File: HammerHead.java:307
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
304  String output = new Date() + " | " + request.getRemoteHost() + ":"
305  + request.getRemoteAddr() + " | " + message;
306  log(output);
307  System.out.println(output);
308  }
309
310
```

HammerHead.java, line 307 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:206
URL: /attack

```
203  {
204  thr.printStackTrace();
205  log(request, "Could not write error screen: "
206  + thr.getMessage());
207  }
208  // System.out.println( "HH Leaving doPost: " );
209  }
```

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 307 (System Information Leak: Internal) — Low

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: log()
File: HammerHead.java:307
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
304 String output = new Date() + " | " + request.getRemoteHost() + ":"
305 + request.getRemoteAddr() + " | " + message;
306 log(output);
307 System.out.println(output);
308 }
309
310
```

HammerHead.java, line 307 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:206
URL: /attack

```
203 {
204 thr.printStackTrace();
205 log(request, "Could not write error screen: "
206 + thr.getMessage());
207 }
208 // System.out.println( "HH Leaving doPost: " );
209 }
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: log()
File: HammerHead.java:307
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
304 String output = new Date() + " | " + request.getRemoteHost() + ":"
305 + request.getRemoteAddr() + " | " + message;
```

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 307 (System Information Leak: Internal) — Low

```
306  log(output);
307  System.out.println(output);
308  }
309
310
```

HammerHead.java, line 306 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.LessonSource.doPost
File: JavaSource/org/owasp/webgoat/LessonSource.java:106
URL: /attack

```
103  {
104  thr.printStackTrace();
105  log(request, "Could not write error screen: "
106  + thr.getMessage());
107  }
108  //System.out.println( "Leaving doPost: " );
109
```

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: log()
File: HammerHead.java:306
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
303  {
304  String output = new Date() + " | " + request.getRemoteHost() + ":"
305  + request.getRemoteAddr() + " | " + message;
306  log(output);
307  System.out.println(output);
308  }
309
```

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 307 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.lang.Throwable.getMessage()
From: org.owasp.webgoat.LessonSource.doPost
File: JavaSource/org/owasp/webgoat/LessonSource.java:106
URL: /attack

```
103  {
104  thr.printStackTrace();
105  log(request, "Could not write error screen: "
106  + thr.getMessage());
107  }
108  //System.out.println( "Leaving doPost: " );
109
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: log()
File: HammerHead.java:307
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
304  String output = new Date() + " | " + request.getRemoteHost() + ":"
305  + request.getRemoteAddr() + " | " + message;
306  log(output);
307  System.out.println(output);
308  }
309
310
```

HammerHead.java, line 193 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

System Information Leak: Internal — Low

URL: /attack

HammerHead.java, line 193 (System Information Leak: Internal) — Low

Source Details

Source: Read t
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:193
URL: /attack

```
190  catch (Throwable t)
191  {
192  t.printStackTrace();
193  log("ERROR: " + t);
194  screen = new ErrorScreen(mySession, t);
195  }
196  finally
```

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: doPost()
File: HammerHead.java:193
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
190  catch (Throwable t)
191  {
192  t.printStackTrace();
193  log("ERROR: " + t);
194  screen = new ErrorScreen(mySession, t);
195  }
196  finally
```

ErrorScreen.java, line 159 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read t
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:194
URL: /attack

```
191  {
192  t.printStackTrace();
```

System Information Leak: Internal — Low

URL: /attack

ErrorScreen.java, line 159 (System Information Leak: Internal) — Low

```
193  log("ERROR: " + t);
194  screen = new ErrorScreen(mySession, t);
195  }
196  finally
197  {
```

Sink Details

Sink: java.io.PrintStream.println()
Enclosing Method: createContent()
File: ErrorScreen.java:159
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
156  */
157  protected Element createContent( WebSession s )
158  {
159  System.out.println( "errorscreen createContent Error:" + this.error + " message:" + this.message );
160
161  Element content;
162
```

URL: /source

LessonSource.java, line 94 (System Information Leak: Internal) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: Read t
From: org.owasp.webgoat.LessonSource.doPost
File: JavaSource/org/owasp/webgoat/LessonSource.java:94
URL: /source

```
91  catch (Throwable t)
92  {
93  t.printStackTrace();
94  log("ERROR: " + t);
95  }
96  finally
97  {
```

System Information Leak: Internal — Low

URL: /source

LessonSource.java, line 94 (System Information Leak: Internal) — Low

Sink Details

Sink: javax.servlet.GenericServlet.log()
Enclosing Method: doPost()
File: LessonSource.java:94
Taint Flags: EXCEPTIONINFO, SYSTEMINFO

```
91 catch (Throwable t)
92 {
93 t.printStackTrace();
94 log("ERROR: " + t);
95 }
96 finally
97 {
```

Trust Boundary Violation (25 issues)

Abstract

Commingling trusted and untrusted data in the same data structure encourages programmers to mistakenly trust unvalidated data.

Explanation

A trust boundary can be thought of as line drawn through a program. On one side of the line, data is untrusted. On the other side of the line, data is assumed to be trustworthy. The purpose of validation logic is to allow data to safely cross the trust boundary--to move from untrusted to trusted.

A trust boundary violation occurs when a program blurs the line between what is trusted and what is untrusted. The most common way to make this mistake is to allow trusted and untrusted data to commingle in the same data structure.

Example: The following Java code accepts an HTTP request and stores the `usrname` parameter in the HTTP session object before checking to ensure that the user has been authenticated.

```
usrname = request.getParameter("usrname");
if (session.getAttribute(ATTR_USR) != null) {
    session.setAttribute(ATTR_USR, usrname);
}
```

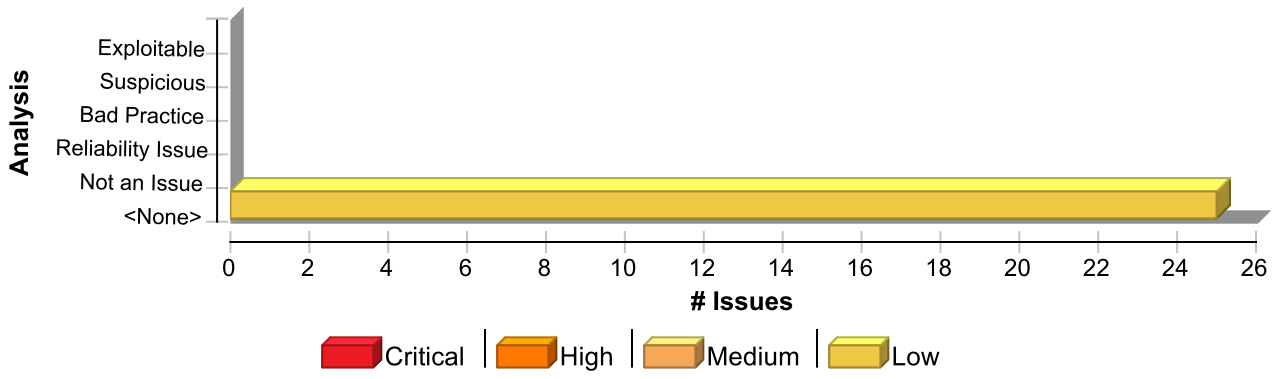
Without well-established and maintained trust boundaries, programmers will inevitably lose track of which pieces of data have been validated and which have not. This confusion will eventually allow some data to be used without first being validated.

Recommendation

Define clear trust boundaries in the application. Do not use the same data structure to hold trusted data in some contexts and untrusted data in other contexts. Minimize the number of ways that data can move across a trust boundary.

Trust boundary violations sometimes occur when input needs to be built up over a series of user interactions before being processed. It may not be possible to do complete input validation until all of the data has arrived. In these situations, it is still important to maintain a trust boundary. The untrusted data should be built up in a single untrusted data structure, validated, and then moved into a trusted location.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Trust Boundary Violation	25	25	0	50
Total	**25**	**25**	**0**	**50**

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.ListStaff.getAllEmployees
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/ListStaff.java:98

```
95  Statement answer_statement = WebSession.getConnection(s)
96  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
97  ResultSet.CONCUR_READ_ONLY);
98  ResultSet answer_results = answer_statement.executeQuery(query);
99  answer_results.beforeFirst();
100  while (answer_results.next())
101  {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SQLInjection.ListStaff.getAllEmployees
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/ListStaff.java:98

```
95  Statement answer_statement = WebSession.getConnection(s)
96  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
97  ResultSet.CONCUR_READ_ONLY);
98  ResultSet answer_results = answer_statement.executeQuery(query);
99  answer_results.beforeFirst();
100  while (answer_results.next())
101  {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SQLInjection.Login.getAllEmployees
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/Login.java:236

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

```
233  Statement answer_statement = WebSession.getConnection(s)
234  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
235  ResultSet.CONCUR_READ_ONLY);
236  ResultSet answer_results = answer_statement.executeQuery(query);
237  answer_results.beforeFirst();
238  while (answer_results.next())
239  {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.CrossSiteScripting.EditProfile.getEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/EditProfile.java:97

```
94  ResultSet.TYPE_SCROLL_INSENSITIVE,
95  ResultSet.CONCUR_READ_ONLY);
96  answer_statement.setInt(1, subjectUserId);
97  ResultSet answer_results = answer_statement.executeQuery();
98  if (answer_results.next())
99  {
100  // Note: Do NOT get the password field.
```

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.ViewProfile.getEmployeeProfi
le
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/ViewProfile.java:
132

```
129  Statement answer_statement = WebSession.getConnection(s)
130  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
131  ResultSet.CONCUR_READ_ONLY);
132  ResultSet answer_results = answer_statement.executeQuery(query);
133  if (answer_results.next())
134  {
135  // Note: Do NOT get the password field.
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
```

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

```
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.CrossSiteScripting.ViewProfile.getEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/CrossSiteScripting/ViewProfile.java:112

```
109  Statement answer_statement = WebSession.getConnection(s)
110  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
111  ResultSet.CONCUR_READ_ONLY);
112  ResultSet answer_results = answer_statement.executeQuery(query);
113  if (answer_results.next())
114  {
115  // Note: Do NOT get the password field.
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: NUMBER, PRIMARY_KEY, WEB, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.sql.Statement.executeQuery()

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

From: org.owasp.webgoat.lessons.RoleBasedAccessControl.Login.getAllEmployees
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/Login.java:194

```
191  Statement answer_statement = WebSession.getConnection(s)
192  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
193  ResultSet.CONCUR_READ_ONLY);
194  ResultSet answer_results = answer_statement.executeQuery(query);
195  answer_results.beforeFirst();
196  while (answer_results.next())
197  {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: org.owasp.webgoat.lessons.RoleBasedAccessControl.EditProfile.getEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/RoleBasedAccessControl/EditProfile.java:97

```
94  ResultSet.TYPE_SCROLL_INSENSITIVE,
95  ResultSet.CONCUR_READ_ONLY);
96  answer_statement.setInt(1, subjectUserId);
97  ResultSet answer_results = answer_statement.executeQuery();
```

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

```
98  if (answer_results.next())
99  {
100  // Note: Do NOT get the password field.
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Taint Flags: WEB, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

DefaultLessonAction.java, line 94 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getStringParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:690

```
687  public String getStringParameter(String name)
688  throws ParameterNotFoundException
689  {
690  String[] values = request.getParameterValues(name);
691  String value;
692
693  if (values == null)
```

Sink Details

Sink: javax.servlet.ServletRequest.setAttribute()
Enclosing Method: setRequestAttribute()
File: DefaultLessonAction.java:94
Taint Flags: NUMBER, PRIMARY_KEY, WEB

```
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
93  {
94  s.getRequest().setAttribute(name, value);
95  }
96
97  public void removeSessionAttribute(WebSession s, String name)
```

Trust Boundary Violation — Low

Package: org.owasp.webgoat.lessons

DefaultLessonAction.java, line 94 (Trust Boundary Violation) — Low

DefaultLessonAction.java, line 89 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.sql.Statement.executeQuery()
From: org.owasp.webgoat.lessons.SQLInjection.ViewProfile.getEmployeeProfile
File: JavaSource/org/owasp/webgoat/lessons/SQLInjection/ViewProfile.java:118

```
115  Statement answer_statement = WebSession.getConnection(s)
116  .createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
117  ResultSet.CONCUR_READ_ONLY);
118  ResultSet answer_results = answer_statement.executeQuery(query);
119  if (answer_results.next())
120  {
121  // Note: Do NOT get the password field.
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: setSessionAttribute()
File: DefaultLessonAction.java:89
Taint Flags: DATABASE, XSS

```
86
87  public void setSessionAttribute(WebSession s, String name, Object value)
88  {
89  s.getRequest().getSession().setAttribute(name, value);
90  }
91
92  public void setRequestAttribute(WebSession s, String name, Object value)
```

Package: org.owasp.webgoat.servlets

Controller.java, line 77 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Trust Boundary Violation — Low

Package: org.owasp.webgoat.servlets

Controller.java, line 77 (Trust Boundary Violation) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65

```
62 undefined
63 undefined
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Sink: javax.servlet.ServletRequest.setAttribute()
Enclosing Method: doPost()
File: Controller.java:77
Taint Flags: WEB, XSS

```
74 undefined
75 undefined
76 undefined
77 undefined
78 undefined
79 undefined
80 undefined
```

URL: /attack

HammerHead.java, line 495 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: java.util.Properties.load()
From: org.owasp.webgoat.session.WebgoatProperties.WebgoatProperties

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 495 (Trust Boundary Violation) — Low

File: JavaSource/org/owasp/webgoat/session/WebgoatProperties.java:44
URL: /attack

```
41 try
42 {
43 FileInputStream in = new FileInputStream(propertiesFileName);
44 load(in);
45 }
46 catch (IOException e)
47 {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: updateSession()
File: HammerHead.java:495
Taint Flags: PROPERTY

```
492 // Create new custom session and save it in the HTTP session
493 // System.out.println( "HH Creating new WebSession: " );
494 session = new WebSession(this, context);
495 hs.setAttribute(WebSession.SESSION, session);
496 // reset timeout
497 hs.setMaxInactiveInterval(sessionTimeoutSeconds);
498
```

HammerHead.java, line 185 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: java.util.Properties.load()
From: org.owasp.webgoat.session.WebgoatProperties.WebgoatProperties
File: JavaSource/org/owasp/webgoat/session/WebgoatProperties.java:44
URL: /attack

```
41 try
42 {
43 FileInputStream in = new FileInputStream(propertiesFileName);
44 load(in);
45 }
```

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 185 (Trust Boundary Violation) — Low

```
46 catch (IOException e)
47 {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:185
Taint Flags: PROPERTY

```
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
188 request, response);
```

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:274
URL: /attack

```
271 .getInitParameter( FEEDBACK_ADDRESS ) : feedbackAddress;
272 showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273 isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274 databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
275 databaseDriver = servlet.getInitParameter( DATABASE_DRIVER );
276 servletName = servlet.getServletName();
277 this.context = context;
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:184

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Taint Flags: PROPERTY

```
181 clientBrowser = userAgent;
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
```

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:271
URL: /attack

```
268 defuseOSCommands = "true".equals( servlet.getInitParameter( DEFUSEOSCOMMANDS ) );
269 enterprise = "true".equals( servlet.getInitParameter( ENTERPRISE ) );
270 feedbackAddress = servlet.getInitParameter( FEEDBACK_ADDRESS ) != null ? servlet
271 .getInitParameter( FEEDBACK_ADDRESS ) : feedbackAddress;
272 showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273 isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274 databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:184
Taint Flags: PROPERTY

```
181 clientBrowser = userAgent;
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
```

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 184 (Trust Boundary Violation) — Low

```
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
```

HammerHead.java, line 495 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:275
URL: /attack

```
272 showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273 isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274 databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
275 databaseDriver = servlet.getInitParameter( DATABASE_DRIVER );
276 servletName = servlet.getServletName();
277 this.context = context;
278 course = new Course();
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: updateSession()
File: HammerHead.java:495
Taint Flags: PROPERTY

```
492 // Create new custom session and save it in the HTTP session
493 // System.out.println( "HH Creating new WebSession: " );
494 session = new WebSession(this, context);
495 hs.setAttribute(WebSession.SESSION, session);
496 // reset timeout
497 hs.setMaxInactiveInterval(sessionTimeoutSeconds);
498
```

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:275
URL: /attack

```
272  showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273  isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274  databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
275  databaseDriver = servlet.getInitParameter( DATABASE_DRIVER );
276  servletName = servlet.getServletName();
277  this.context = context;
278  course = new Course();
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:184
Taint Flags: PROPERTY

```
181  clientBrowser = userAgent;
182  }
183  request.setAttribute("client.browser", clientBrowser);
184  request.getSession().setAttribute("websession", mySession);
185  request.getSession().setAttribute("course", mySession.getCourse());
186
187  request.getRequestDispatcher(getViewPage(mySession)).forward(
```

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 184 (Trust Boundary Violation) — Low

URL: /attack

```
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
177  String userAgent = request.getHeader("user-agent");
178  String clientBrowser = "Not known!";
179  if (userAgent != null)
180  {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:184
Taint Flags: WEB, XSS

```
181  clientBrowser = userAgent;
182  }
183  request.setAttribute("client.browser", clientBrowser);
184  request.getSession().setAttribute("websession", mySession);
185  request.getSession().setAttribute("course", mySession.getCourse());
186
187  request.getRequestDispatcher(getViewPage(mySession)).forward(
```

HammerHead.java, line 183 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.HammerHead.doPost
File: JavaSource/org/owasp/webgoat/HammerHead.java:177
URL: /attack

```
174  + mySession.getParser().toString());
175
176  // Redirect the request to our View servlet
177  String userAgent = request.getHeader("user-agent");
178  String clientBrowser = "Not known!";
```

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 183 (Trust Boundary Violation) — Low

```
179 if (userAgent != null)
180 {
```

Sink Details

Sink: javax.servlet.ServletRequest.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:183
Taint Flags: WEB, XSS

```
180 {
181 clientBrowser = userAgent;
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
```

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.http.HttpServletRequest.getHeader()
From: org.owasp.webgoat.servlets.Controller.doPost
File: JavaSource/org/owasp/webgoat/servlets/Controller.java:65
URL: /attack

```
62 undefined
63 undefined
64 undefined
65 undefined
66 undefined
67 undefined
68 undefined
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:184

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Taint Flags: WEB, XSS

```
181 clientBrowser = userAgent;
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
```

HammerHead.java, line 184 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Source Details

Source: java.util.Properties.load()
From: org.owasp.webgoat.session.WebgoatProperties.WebgoatProperties
File: JavaSource/org/owasp/webgoat/session/WebgoatProperties.java:44
URL: /attack

```
41 try
42 {
43 FileInputStream in = new FileInputStream(propertiesFileName);
44 load(in);
45 }
46 catch (IOException e)
47 {
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: doPost()
File: HammerHead.java:184
Taint Flags: PROPERTY

```
181 clientBrowser = userAgent;
182 }
183 request.setAttribute("client.browser", clientBrowser);
184 request.getSession().setAttribute("websession", mySession);
185 request.getSession().setAttribute("course", mySession.getCourse());
186
```

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 184 (Trust Boundary Violation) — Low

```
187 request.getRequestDispatcher(getViewPage(mySession)).forward(
```

HammerHead.java, line 495 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:271
URL: /attack

```
268 defuseOSCommands = "true".equals( servlet.getInitParameter( DEFUSEOSCOMMANDS ) );
269 enterprise = "true".equals( servlet.getInitParameter( ENTERPRISE ) );
270 feedbackAddress = servlet.getInitParameter( FEEDBACK_ADDRESS ) != null ? servlet
271 .getInitParameter( FEEDBACK_ADDRESS ) : feedbackAddress;
272 showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273 isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274 databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: updateSession()
File: HammerHead.java:495
Taint Flags: PROPERTY

```
492 // Create new custom session and save it in the HTTP session
493 // System.out.println( "HH Creating new WebSession: " );
494 session = new WebSession(this, context);
495 hs.setAttribute(WebSession.SESSION, session);
496 // reset timeout
497 hs.setMaxInactiveInterval(sessionTimeoutSeconds);
498
```

HammerHead.java, line 495 (Trust Boundary Violation) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Data Flow)

Trust Boundary Violation — Low

URL: /attack

HammerHead.java, line 495 (Trust Boundary Violation) — Low

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:274
URL: /attack

```
271  .getInitParameter( FEEDBACK_ADDRESS ) : feedbackAddress;
272  showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273  isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274  databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
275  databaseDriver = servlet.getInitParameter( DATABASE_DRIVER );
276  servletName = servlet.getServletName();
277  this.context = context;
```

Sink Details

Sink: javax.servlet.http.HttpSession.setAttribute()
Enclosing Method: updateSession()
File: HammerHead.java:495
Taint Flags: PROPERTY

```
492  // Create new custom session and save it in the HTTP session
493  // System.out.println( "HH Creating new WebSession: " );
494  session = new WebSession(this, context);
495  hs.setAttribute(WebSession.SESSION, session);
496  // reset timeout
497  hs.setMaxInactiveInterval(sessionTimeoutSeconds);
498
```

Unchecked Return Value (2 issues)

Abstract

Ignoring a method's return value can cause the program to overlook unexpected states and conditions.

Explanation

It is not uncommon for Java programmers to misunderstand `read()` and related methods that are part of many `java.io` classes. Most errors and unusual events in Java result in an exception being thrown. (This is one of the advantages that Java has over languages like C: Exceptions make it easier for programmers to think about what can go wrong.) But the stream and reader classes do not consider it unusual or exceptional if only a small amount of data becomes available. These classes simply add the small amount of data to the return buffer, and set the return value to the number of bytes or characters read. There is no guarantee that the amount of data returned is equal to the amount of data requested.

This behavior makes it important for programmers to examine the return value from `read()` and other IO methods to ensure that they receive the amount of data they expect.

Example: The following code loops through a set of users, reading a private data file for each user. The programmer assumes that the files are always exactly 1 kilobyte in size and therefore ignores the return value from `read()`. If an attacker can create a smaller file, the program will recycle the remainder of the data from the previous user and handle it as though it belongs to the attacker.

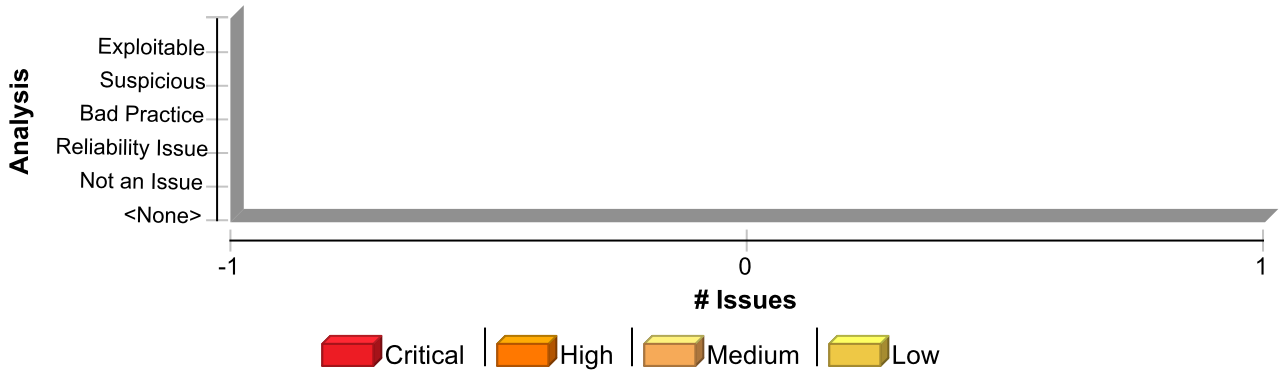
```
FileInputStream fis;
byte[] byteArray = new byte[1024];
for (Iterator i=users.iterator(); i.hasNext();) {
    String userName = (String) i.next();
    String pFileName = PFILE_ROOT + "/" + userName;
    FileInputStream fis = new FileInputStream(pFileName);
    fis.read(byteArray); // the file is always 1k bytes
    fis.close();
    processPFile(userName, byteArray);
}
```

Recommendation

```
  FileInputStream fis;
  byte[] byteArray = new byte[1024];
  for (Iterator i=users.iterator(); i.hasNext();) {
    String userName = (String) i.next();
    String pFileName = PFILE_ROOT + "/" + userName;
    fis = new FileInputStream(pFileName);
    int bRead = 0;
    while (bRead < 1024) {
        int rd = fis.read(byteArray, bRead, 1024 - bRead);
        if (rd == -1) {
          throw new IOException("file is unusually small");
        }
        bRead += rd;
    }
    // could add check to see if file is too large here
    fis.close();
    processPFile(userName, byteArray);
  }
```

Note: Because the fix for this problem is relatively complicated, you might be tempted to use a simpler approach, such as checking the size of the file before you begin reading. Such an approach would render the application vulnerable to a file system race condition, whereby an attacker could replace a well-formed file with a malicious file between the file size check and the call to read data from the file.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unchecked Return Value	2	2	0	4
Total	**2**	**2**	**0**	**4**

Unchecked Return Value — Low

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 140 (Unchecked Return Value) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: replaceAll()
Enclosing Method: getHints()
File: SoapRequest.java:140

```
137  + "  &lt;/SOAP-ENV:Body&gt; <br>"
138  + "&lt;/SOAP-ENV:Envelope&gt; <br><br>"
139  + "Intercept the HTTP request and try to create a SOAP request.";
140  soapEnv.replaceAll("(?s) ", " ");
141  hints.add(soapEnv);
142
143  return hints;
```

Unchecked Return Value — Low

Package: org.owasp.webgoat.lessons

CommandInjection.java, line 220 (Unchecked Return Value) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Sink Details

Sink: replaceAll()
Enclosing Method: parseResults()
File: CommandInjection.java:220

```
217
218 private String parseResults(String results)
219 {
220 results.replaceAll("(?s).*Output...\\s", "").replaceAll("(?s)Returncode.*", "");
221 StringTokenizer st = new StringTokenizer(results, "\n");
222 StringBuffer modified = new StringBuffer();
223
```

Unreleased Resource: Database (13 issues)

Abstract

The program can potentially fail to release a database resource.

Explanation

Resource leaks have at least two common causes:

- Error conditions and other exceptional circumstances.

- Confusion over which part of the program is responsible for releasing the resource.

Most unreleased resource issues result in general software reliability problems, but if an attacker can intentionally trigger a resource leak, the attacker may be able to launch a denial of service attack by depleting the resource pool.

Example: Under normal conditions, the following code executes a database query, processes the results returned by the database, and closes the allocated statement object. But if an exception occurs while executing the SQL or processing the results, the statement object will not be closed. If this happens often enough, the database will run out of available cursors and not be able to execute any more SQL queries.

```
  Statement stmt = conn.createStatement();
  ResultSet rs = stmt.executeQuery(CXN_SQL);
  harvestResults(rs);
  stmt.close();
```

In this case, there are program paths on which a Statement is not released.

Recommendation

1. Never rely on `finalize()` to reclaim resources. In order for an object's `finalize()` method to be invoked, the garbage collector must determine that the object is eligible for garbage collection. Because the garbage collector is not required to run unless the JVM is low on memory, there is no guarantee that an object's `finalize()` method will be invoked in an expedient fashion. When the garbage collector finally does run, it may cause a large number of resources to be reclaimed in a short period of time, which can lead to "bursty" performance and lower overall system throughput. This effect becomes more pronounced as the load on the system increases.

Finally, if it is possible for a resource reclamation operation to hang (if it requires communicating over a network to a database, for example), then the thread that is executing the `finalize()` method will hang.

2. Release resources in a `finally` block. The code for the Example should be rewritten as follows:

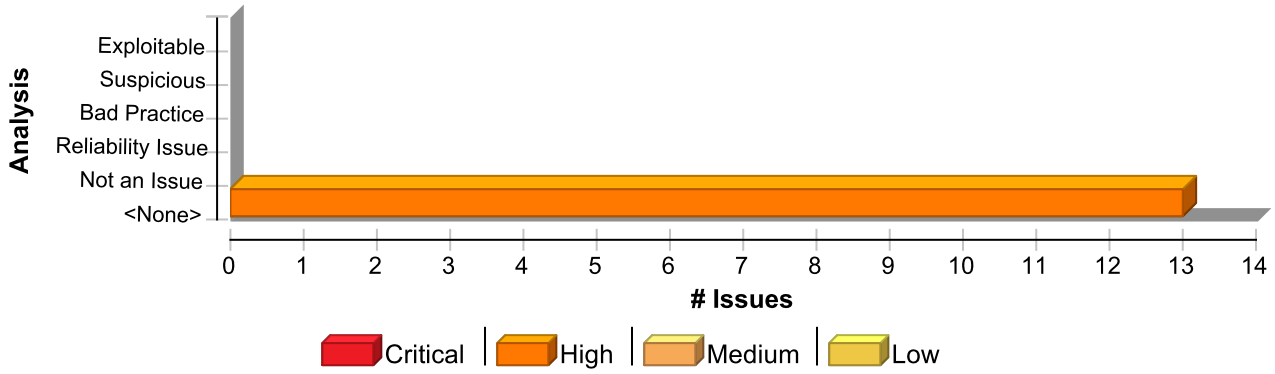
```
  public void execCxnSql(Connection conn) {
    Statement stmt;
    try {
      stmt = conn.createStatement();
      ResultSet rs = stmt.executeQuery(CXN_SQL);
      ...
    }
    finally {
      if (stmt != null) {
        safeClose(stmt);
      }
    }
}

public static void safeClose(Statement stmt) {
```

```
  if (stmt != null) {
    try {
      stmt.close();
    } catch (SQLException e) {
      log(e);
    }
  }
}
```

This solution uses a helper function to log the exceptions that might occur when trying to close the statement. Presumably this helper function will be reused whenever a statement needs to be closed.

Also, the `execCxnSql` method does not initialize the `stmt` object to null. Instead, it checks to ensure that `stmt` is not `null` before calling `safeClose()`. Without the `null` check, the Java compiler reports that `stmt` might not be initialized. This choice takes advantage of Java's ability to detect uninitialized variables. If `stmt` is initialized to `null` in a more complex method, cases in which `stmt` is used without being initialized will not be detected by the compiler.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unreleased Resource: Database	13	13	0	26
Total	**13**	**13**	**0**	**26**

Unreleased Resource: Database	**High**
Package: org.owasp.webgoat.lessons	
SoapRequest.java, line 418 (Unreleased Resource: Database)	**High**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Sink Details

Unreleased Resource: Database — High

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 418 (Unreleased Resource: Database) — High

Sink: ps = connection.prepareStatement(...)
Enclosing Method: getResults()
File: SoapRequest.java:418

```
415 return null;
416 }
417 PreparedStatement ps = connection
418 .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
419 ps.setInt(1, id);
420 try
421 {
```

WSDLScanning.java, line 274 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: connection = makeConnection()
Enclosing Method: getResults()
File: WSDLScanning.java:274

```
271 {
272 try
273 {
274 Connection connection = DatabaseUtilities.makeConnection();
275 if (connection == null)
276 {
277 return null;
```

WSDLScanning.java, line 284 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: results = ps.executeQuery()
Enclosing Method: getResults()

Unreleased Resource: Database — High

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 284 (Unreleased Resource: Database) — High

File: WSDLScanning.java:284

```
281 ps.setInt(1, id);
282 try
283 {
284 ResultSet results = ps.executeQuery();
285 if ((results != null) && (results.next() == true))
286 {
287 return results.getString(field);
```

SoapRequest.java, line 422 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: results = ps.executeQuery()
Enclosing Method: getResults()
File: SoapRequest.java:422

```
419 ps.setInt(1, id);
420 try
421 {
422 ResultSet results = ps.executeQuery();
423 if ((results != null) && (results.next() == true))
424 {
425 return results.getString(field);
```

WsSqlInjection.java, line 237 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: statement = connection.createStatement(...)
Enclosing Method: getResults()
File: WsSqlInjection.java:237

Unreleased Resource: Database — High

Package: org.owasp.webgoat.lessons

WsSqlInjection.java, line 237 (Unreleased Resource: Database) — High

```
234 String query = "SELECT * FROM user_data WHERE userid = " + id;
235 try
236 {
237 Statement statement = connection.createStatement(
238 ResultSet.TYPE_SCROLL_INSENSITIVE,
239 ResultSet.CONCUR_READ_ONLY);
240 ResultSet results = statement.executeQuery(query);
```

WsSqlInjection.java, line 229 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: connection = makeConnection()
Enclosing Method: getResults()
File: WsSqlInjection.java:229

```
226 {
227 try
228 {
229 Connection connection = DatabaseUtilities.makeConnection();
230 if (connection == null)
231 {
232 return null;
```

SoapRequest.java, line 412 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: connection = makeConnection()
Enclosing Method: getResults()
File: SoapRequest.java:412

```
409 {
410 try
```

Unreleased Resource: Database — High

Package: org.owasp.webgoat.lessons

SoapRequest.java, line 412 (Unreleased Resource: Database) — High

```
411 {
412 Connection connection = DatabaseUtilities.makeConnection();
413 if (connection == null)
414 {
415 return null;
```

WsSqlInjection.java, line 240 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: results = statement.executeQuery(...)
Enclosing Method: getResults()
File: WsSqlInjection.java:240

```
237 Statement statement = connection.createStatement(
238 ResultSet.TYPE_SCROLL_INSENSITIVE,
239 ResultSet.CONCUR_READ_ONLY);
240 ResultSet results = statement.executeQuery(query);
241 return results;
242 }
243 catch (SQLException sqle)
```

WSDLScanning.java, line 280 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: ps = connection.prepareStatement(...)
Enclosing Method: getResults()
File: WSDLScanning.java:280

```
277 return null;
278 }
279 PreparedStatement ps = connection
280 .prepareStatement("SELECT * FROM user_data WHERE userid = ?");
```

Unreleased Resource: Database — High

Package: org.owasp.webgoat.lessons

WSDLScanning.java, line 280 (Unreleased Resource: Database) — High

```
281 ps.setInt(1, id);
282 try
283 {
```

Package: org.owasp.webgoat.session

CreateDB.java, line 91 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: answer_results = answer_statement.executeQuery(...)
Enclosing Method: main()
File: CreateDB.java:91

```
88 Statement answer_statement = connection.createStatement(
89 ResultSet.TYPE_SCROLL_INSENSITIVE,
90 ResultSet.CONCUR_READ_ONLY);
91 ResultSet answer_results = answer_statement.executeQuery(query);
92 answer_results.first();
93 int employeeId = answer_results.getInt("userid");
94 String firstName = answer_results.getString("first_name");
```

CreateDB.java, line 88 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: answer_statement = connection.createStatement(...)
Enclosing Method: main()
File: CreateDB.java:88

```
85
86 try
87 {
88 Statement answer_statement = connection.createStatement(
```

Unreleased Resource: Database — High

Package: org.owasp.webgoat.session

CreateDB.java, line 88 (Unreleased Resource: Database) — High

```
89 ResultSet.TYPE_SCROLL_INSENSITIVE,
90 ResultSet.CONCUR_READ_ONLY);
91 ResultSet answer_results = answer_statement.executeQuery(query);
```

CreateDB.java, line 70 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: connection = getConnection(...)
Enclosing Method: main()
File: CreateDB.java:70

```
67 {
68
69 connection = DriverManager
70 .getConnection(
71 "jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=c:/webgoat.mdb;PWD=webgoat",
72 "webgoat", "webgoat");
73 db.makeDB(connection);
```

CreateDB.java, line 73 (Unreleased Resource: Database) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: ?.makeDB(connection)
Enclosing Method: main()
File: CreateDB.java:73

```
70 .getConnection(
71 "jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=c:/webgoat.mdb;PWD=webgoat",
72 "webgoat", "webgoat");
73 db.makeDB(connection);
74 }
75 catch (Exception e)
```

Unreleased Resource: Database	High
Package: org.owasp.webgoat.session	
CreateDB.java, line 73 (Unreleased Resource: Database)	**High**

```
76 {
```

Unreleased Resource: Sockets (3 issues)

Abstract

The program can potentially fail to release a socket.

Explanation

The program can potentially fail to release a socket.

Resource leaks have at least two common causes:

- Error conditions and other exceptional circumstances.

- Confusion over which part of the program is responsible for releasing the resource.

Most unreleased resource issues result in general software reliability problems, but if an attacker can intentionally trigger a resource leak, the attacker may be able to launch a denial of service attack by depleting the resource pool.

Example 1: The following method never closes the socket it opens. In a busy environment, this can result in the JVM using up all of its sockets.

```
private void echoSocket(String host, int port) throws UnknownHostException,
SocketException, IOException
{
  Socket sock = new Socket(host, port);
  BufferedReader reader = new BufferedReader(new
InputStreamReader(sock.getInputStream()));

  while ((String socketData = reader.readLine()) != null) {
    System.out.println(socketData);
  }
}
```

Example 2: Under normal conditions, the following fix properly closes the socket and any associated streams. But if an exception occurs while reading the input or writing the data to screen, the socket object will not be closed. If this happens often enough, the system will run out of sockets and not be able to handle any further connections.

```
private void echoSocket(String host, int port) throws UnknownHostException,
SocketException, IOException
{
  Socket sock = new Socket(host, port);
  BufferedReader reader = new BufferedReader(new
InputStreamReader(sock.getInputStream()));

  while ((String socketData = reader.readLine()) != null) {
    System.out.println(socketData);
  }
  sock.close();
}
```

Recommendation

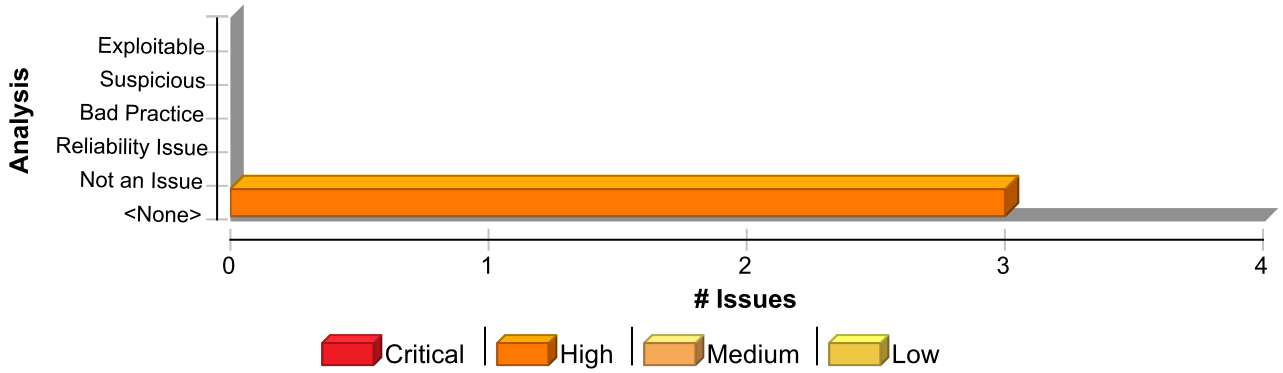
Release socket resources in a `finally` block. The code for Example 2 should be rewritten as follows:

```
private void echoSocket(String host, int port) throws UnknownHostException,
SocketException, IOException
{
  Socket sock;
  BufferedReader reader;

  try {
    sock = new Socket(host, port);
    reader = new BufferedReader(new InputStreamReader(sock.getInputStream()));

    while ((String socketData = reader.readLine()) != null) {
        System.out.println(socketData);
    }
  }
  finally {
    safeClose(sock);
  }
}

public static void safeClose(Socket s) {
  if (s != null && !s.isClosed()) {
    try {
      s.close();
    } catch (IOException e) {
      log(e);
    }
  }
}
```

This solution uses a helper function to log the exceptions that might occur when trying to close the socket. Presumably this helper function will be reused whenever a socket needs to be closed.

Also, the `echoSocket()` method does not initialize the `sock` socket object to null. Instead, it checks to ensure that `sock` is not `null` before calling `safeClose()`. Without the `null` check, the Java compiler reports that `sock` might not be initialized. This choice takes advantage of Java's ability to detect uninitialized variables. If `sock` is initialized to `null` in a more complex method, cases in which `sock` is used without being initialized will not be detected by the compiler.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unreleased Resource: Sockets	3	3	0	6
Total	**3**	**3**	**0**	**6**

Unreleased Resource: Sockets — High

Package: org.owasp.webgoat.util

Interceptor.java, line 97 (Unreleased Resource: Sockets) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: osgSocket.getOutputStream()
Enclosing Method: doFilter()
File: Interceptor.java:97

```
94  .parseInt(osgServerPort));
95  if (osgSocket != null)
96  {
97  out = new PrintWriter(osgSocket.getOutputStream(), true);
98  in = new BufferedReader(new InputStreamReader(osgSocket
99  .getInputStream()));
100 //String message = "HTTPRECEIVEHTTPREQUEST,-,DataValidation_SqlInjection_Basic.aspx";
```

Interceptor.java, line 99 (Unreleased Resource: Sockets) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: osgSocket.getInputStream()
Enclosing Method: doFilter()
File: Interceptor.java:99

```
96  {
97  out = new PrintWriter(osgSocket.getOutputStream(), true);
98  in = new BufferedReader(new InputStreamReader(osgSocket
99  .getInputStream()));
100 //String message = "HTTPRECEIVEHTTPREQUEST,-,DataValidation_SqlInjection_Basic.aspx";
101 //out.println(message);
```

Unreleased Resource: Sockets — High

Package: org.owasp.webgoat.util

Interceptor.java, line 99 (Unreleased Resource: Sockets) — High

```
102
```

Interceptor.java, line 93 (Unreleased Resource: Sockets) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: osgSocket = new Socket(...)
Enclosing Method: doFilter()
File: Interceptor.java:93

```
90  if (osgServerName != null && osgServerName.length() != 0
91  && osgServerPort != null && osgServerPort.length() != 0)
92  {
93  osgSocket = new Socket(osgServerName, Integer
94  .parseInt(osgServerPort));
95  if (osgSocket != null)
96  {
```

Unreleased Resource: Streams (15 issues)

Abstract

The program can potentially fail to release a system resource.

Explanation

The program can potentially fail to release a system resource.

Resource leaks have at least two common causes:

- Error conditions and other exceptional circumstances.

- Confusion over which part of the program is responsible for releasing the resource.

Most unreleased resource issues result in general software reliability problems, but if an attacker can intentionally trigger a resource leak, the attacker may be able to launch a denial of service attack by depleting the resource pool.

Example: The following method never closes the file handle it opens. The `finalize()` method for `FileInputStream` eventually calls `close()`, but there is no guarantee as to how long it will take before the `finalize()` method will be invoked. In a busy environment, this can result in the JVM using up all of its file handles.

```
private void processFile(String fName) throws FileNotFoundException,
IOException {
  FileInputStream fis = new FileInputStream(fName);
  int sz;
  byte[] byteArray = new byte[BLOCK_SIZE];
  while ((sz = fis.read(byteArray)) != -1) {
    processBytes(byteArray, sz);
  }
}
```

Recommendation

1. Never rely on `finalize()` to reclaim resources. In order for an object's `finalize()` method to be invoked, the garbage collector must determine that the object is eligible for garbage collection. Because the garbage collector is not required to run unless the JVM is low on memory, there is no guarantee that an object's `finalize()` method will be invoked in an expedient fashion. When the garbage collector finally does run, it may cause a large number of resources to be reclaimed in a short period of time, which can lead to "bursty" performance and lower overall system throughput. This effect becomes more pronounced as the load on the system increases.

Finally, if it is possible for a resource reclamation operation to hang (if it requires communicating over a network to a database, for example), then the thread that is executing the `finalize()` method will hang.

2. Release resources in a `finally` block. The code for the Example should be rewritten as follows:

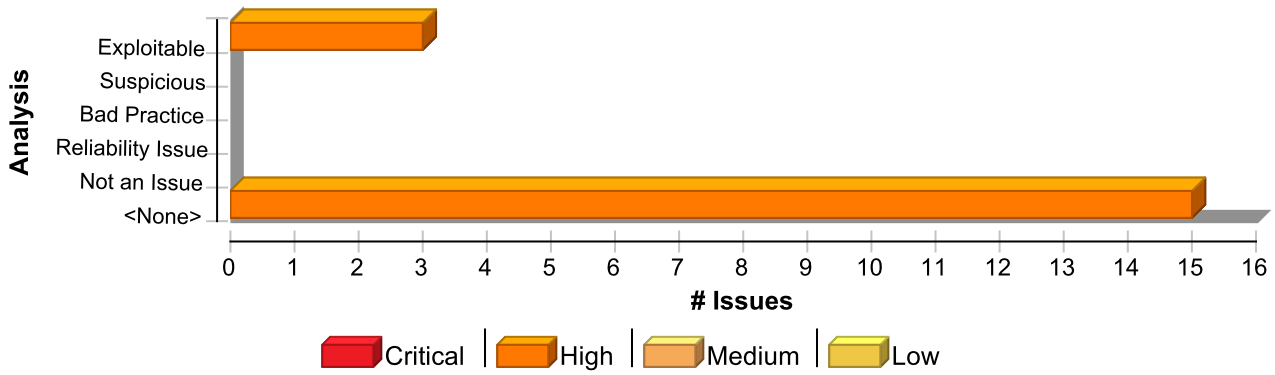
```
public void processFile(String fName) throws FileNotFoundException,
IOException {
  FileInputStream fis;
  try {
    fis = new FileInputStream(fName);
    int sz;
    byte[] byteArray = new byte[BLOCK_SIZE];
    while ((sz = fis.read(byteArray)) != -1) {
```

```
      processBytes(byteArray, sz);
    }
  }
  finally {
    if (fis != null) {
      safeClose(fis);
    }
  }
}

public static void safeClose(FileInputStream fis) {
  if (fis != null) {
    try {
      fis.close();
    } catch (IOException e) {
      log(e);
    }
  }
}
```

This solution uses a helper function to log the exceptions that might occur when trying to close the stream. Presumably this helper function will be reused whenever a stream needs to be closed.

Also, the `processFile` method does not initialize the `fis` object to null. Instead, it checks to ensure that `fis` is not `null` before calling `safeClose()`. Without the `null` check, the Java compiler reports that `fis` might not be initialized. This choice takes advantage of Java's ability to detect uninitialized variables. If `fis` is initialized to `null` in a more complex method, cases in which `fis` is used without being initialized will not be detected by the compiler.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unreleased Resource: Streams	15	15	0	30
Total	**15**	**15**	**0**	**30**

Unreleased Resource: Streams	**High**
Package: org.owasp.webgoat.lessons	
Challenge2Screen.java, line 381 (Unreleased Resource: Streams)	**High**
Issue Details	

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 381 (Unreleased Resource: Streams) — High

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: getFileText(new java.io.BufferedReader(), ?)
Enclosing Method: isDefaced()
File: Challenge2Screen.java:381

```
378 WEBGOAT_CHALLENGE + "_" + s.getUserName() + JSP);
379 String masterFilePath = s.getContext().getRealPath(
380 WEBGOAT_CHALLENGE_JSP);
381 String defacedText = getFileText(new BufferedReader(new FileReader(
382 origpath)), false);
383 String origText = getFileText(new BufferedReader(new FileReader(
384 masterFilePath)), false);
```

Challenge2Screen.java, line 383 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: getFileText(new java.io.BufferedReader(), ?)
Enclosing Method: isDefaced()
File: Challenge2Screen.java:383

```
380 WEBGOAT_CHALLENGE_JSP);
381 String defacedText = getFileText(new BufferedReader(new FileReader(
382 origpath)), false);
383 String origText = getFileText(new BufferedReader(new FileReader(
384 masterFilePath)), false);
385
386 defaced = (!origText.equals(defacedText));
```

AbstractLesson.java, line 1031 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 1031 (Unreleased Resource: Streams) — High

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: reader = new BufferedReader(new java.io.InputStreamReader())
Enclosing Method: readFromURL()
File: AbstractLesson.java:1031

```
1028 {
1029 URL u = new URL(url);
1030 HttpURLConnection huc = (HttpURLConnection) u.openConnection();
1031 BufferedReader reader = new BufferedReader(new InputStreamReader(
1032 huc.getInputStream()));
1033 String line;
1034
```

Challenge2Screen.java, line 434 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: getFileText(new java.io.BufferedReader(), ?)
Enclosing Method: resetWebPage()
File: Challenge2Screen.java:434

```
431 // replace the defaced text with the original
432 File usersFile = new File(defacedpath);
433 FileWriter fw = new FileWriter(usersFile);
434 fw.write(getFileText(new BufferedReader(new FileReader(
435 masterFilePath)), false));
436 fw.close();
437 // System.out.println("webgoat_guest replaced: " + getFileText( new BufferedReader( new FileReader( defacedpath ) ), false ) );
```

XPATHInjection.java, line 155 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

XPATHInjection.java, line 155 (Unreleased Resource: Streams) — High

Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:02 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: new FileInputStream(...)
Enclosing Method: createContent()
File: XPATHInjection.java:155

```
152 File d = new File(dir);
153 XPathFactory factory = XPathFactory.newInstance();
154 XPath xPath = factory.newXPath();
155 InputSource inputSource = new InputSource(new FileInputStream(d));
156 String expression = "/employees/employee[loginID/text()='"
157 + username + "' and passwd/text()='" + password + "']";
158 nodes = (NodeList) xPath.evaluate(expression, inputSource,
```

Challenge2Screen.java, line 404 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Exploitable threshold)

Sink Details

Sink: getFileText(new java.io.BufferedReader(), ?)
Enclosing Method: showDefaceAttempt()
File: Challenge2Screen.java:404

```
401 // get current text and compare to the new text
402 String origpath = s.getContext().getRealPath(
403 WEBGOAT_CHALLENGE + "_" + s.getUserName() + JSP);
404 String defaced = getFileText(new BufferedReader(
405 new FileReader(origpath)), false);
406 String origText = getFileText(new BufferedReader(new FileReader(s
407 .getContext().getRealPath(WEBGOAT_CHALLENGE_JSP))), false);
```

Challenge2Screen.java, line 406 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

Challenge2Screen.java, line 406 (Unreleased Resource: Streams) — High

Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: getFileText(new java.io.BufferedReader(), ?)
Enclosing Method: showDefaceAttempt()
File: Challenge2Screen.java:406

```
403 WEBGOAT_CHALLENGE + "_" + s.getUserName() + JSP);
404 String defaced = getFileText(new BufferedReader(
405 new FileReader(origpath)), false);
406 String origText = getFileText(new BufferedReader(new FileReader(s
407 .getContext().getRealPath(WEBGOAT_CHALLENGE_JSP))), false);
408
409 // show webgoat.jsp text
```

Challenge2Screen.java, line 433 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Not An Issue threshold)

Sink Details

Sink: fw = new FileWriter(...)
Enclosing Method: resetWebPage()
File: Challenge2Screen.java:433

```
430
431 // replace the defaced text with the original
432 File usersFile = new File(defacedpath);
433 FileWriter fw = new FileWriter(usersFile);
434 fw.write(getFileText(new BufferedReader(new FileReader(
435 masterFilePath)), false));
436 fw.close();
```

AbstractLesson.java, line 669 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 669 (Unreleased Resource: Streams) — High

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: readFromFile(new java.io.BufferedReader(), ?)
Enclosing Method: getSource()
File: AbstractLesson.java:669

```
666 {
667 // System.out.println("Loading source file: " +
668 // getSourceFileName());
669 src = convertMetacharsJavaCode(readFromFile(new BufferedReader(
670 new FileReader(s.getWebResource(getSourceFileName()))),
671 true));
672
```

AbstractLesson.java, line 567 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence	
Analysis	Exploitable
AA_Prediction	Exploitable

Audit Comments

Auto applied: Thu Oct 11 2018 10:09:03 GMT-0500 (CDT)
Value of attribute [Analysis] was set to [Exploitable] because of enabled auto apply configuration and master attribute [AA_Prediction] value was [Exploitable]

Sink Details

Sink: readFromFile(new java.io.BufferedReader(), ?)
Enclosing Method: getLessonPlan()
File: AbstractLesson.java:567

```
564 {
565 // System.out.println("Loading lesson plan file: " +
566 // getLessonPlanFileName());
567 src = readFromFile(new BufferedReader(new FileReader(s
568 .getWebResource(getLessonPlanFileName()))), false);
```

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

AbstractLesson.java, line 567 (Unreleased Resource: Streams) — High

```
569
570 }
```

PathBasedAccessControl.java, line 192 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: getFileText(new java.io.BufferedReader(), ?)
Enclosing Method: createContent()
File: PathBasedAccessControl.java:192

```
189 {
190 throw new Exception("File is too large");
191 }
192 String fileData = getFileText(new BufferedReader(
193 new FileReader(f)), false);
194 if (fileData.indexOf(0x00) != -1)
195 {
```

LessonAdapter.java, line 93 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: in = new BufferedReader(new java.io.FileReader())
Enclosing Method: createContent()
File: LessonAdapter.java:93

```
90 try
91 {
92 PRE pre = new PRE();
93 BufferedReader in = new BufferedReader(new FileReader(fileName));
94 String line = null;
95 while ((line = in.readLine()) != null)
96 {
```

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.lessons

LessonAdapter.java, line 93 (Unreleased Resource: Streams) — High

LessonAdapter.java, line 285 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: in = new BufferedReader(new java.io.FileReader())
Enclosing Method: getInstructions()
File: LessonAdapter.java:285

```
282 String fileName = s.getWebResource(getLessonPlanFileName());
283 if (fileName != null)
284 {
285 BufferedReader in = new BufferedReader(new FileReader(fileName));
286 String line = null;
287 boolean startAppending = false;
288 while ((line = in.readLine()) != null)
```

AbstractLesson.java, line 954 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: readFromFile(new java.io.BufferedReader(), ?)
Enclosing Method: makeSourceDump_DELETEME()
File: AbstractLesson.java:954

```
951 t
952 .addElement(new TR()
953 .addElement(new TD()
954 .addElement(convertMetachars(readFromFile(
955 new BufferedReader(new FileReader(
956 filename)), true)))));
957 }
```

Unreleased Resource: Streams — High

Package: org.owasp.webgoat.session

WebgoatProperties.java, line 43 (Unreleased Resource: Streams) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Audit Details

AA_Confidence

AA_Prediction Indeterminate (Below Exploitable threshold)

Sink Details

Sink: in = new FileInputStream(...)
Enclosing Method: WebgoatProperties()
File: WebgoatProperties.java:43

```
40 {
41 try
42 {
43 FileInputStream in = new FileInputStream(propertiesFileName);
44 load(in);
45 }
46 catch (IOException e)
```

Unsafe Reflection (1 issue)

Abstract

An attacker may be able to create unexpected control flow paths through the application, potentially bypassing security checks.

Explanation

If an attacker can supply values that the application then uses to determine which class to instantiate or which method to invoke, the potential exists for the attacker to create control flow paths through the application that were not intended by the application developers. This attack vector may allow the attacker to bypass authentication or access control checks or otherwise cause the application to behave in an unexpected manner. Even the ability to control the arguments passed to a given method or constructor may give a wily attacker the edge necessary to mount a successful attack.

This situation becomes a doomsday scenario if the attacker may upload files into a location that appears on the application's classpath or add new entries to the application's classpath. Under either of these conditions, the attacker may use reflection to introduce new, presumably malicious, behavior into the application.

Example: A common reason that programmers use the reflection API is to implement their own command dispatcher. The following example shows a command dispatcher that does not use reflection:

```
String ctl = request.getParameter("ctl");
Worker ao = null;
if (ctl.equals("Add")) {
  ao = new AddCommand();
} else if (ctl.equals("Modify")) {
  ao = new ModifyCommand();
} else {
  throw new UnknownActionError();
}
ao.doAction(request);
```

A programmer might refactor this code to use reflection as follows:

```
String ctl = request.getParameter("ctl");
Class cmdClass = Class.forName(ctl + "Command");
Worker ao = (Worker) cmdClass.newInstance();
ao.doAction(request);
```

The refactoring initially appears to offer a number of advantages. There are fewer lines of code, the `if/else` blocks have been entirely eliminated, and it is now possible to add new command types without modifying the command dispatcher.

However, the refactoring allows an attacker to instantiate any object that implements the `Worker` interface. If the command dispatcher is still responsible for access control, then whenever programmers create a new class that implements the `Worker` interface, they must remember to modify the dispatcher's access control code. If they fail to modify the access control code, then some `Worker` classes will not have any access control.

One way to address this access control problem is to make the `Worker` object responsible for performing the access control check. An example of the re-refactored code is as follows:

```
String ctl = request.getParameter("ctl");
Class cmdClass = Class.forName(ctl + "Command");
Worker ao = (Worker) cmdClass.newInstance();
ao.checkAccessControl(request);
ao.doAction(request);
```

Although this is an improvement, it encourages a decentralized approach to access control, which makes it easier for programmers to make access control mistakes.

This code also highlights another security problem with using reflection to build a command dispatcher. An attacker may invoke the default constructor for any kind of object. In fact, the attacker is not even constrained to objects that implement the `Worker` interface; the default constructor for any object in the system can be invoked. If the object does not implement the `Worker` interface, a `ClassCastException` will be thrown before the assignment to `ao`, but if the constructor performs operations that work in the attacker's favor, the damage will have already been done. Although this scenario is relatively benign in simple applications, in larger applications where complexity grows exponentially it is not unreasonable to assume that an attacker could find a constructor to leverage as part of an attack.

Access checks may also be compromised further down the code execution chain, if certain Java APIs that perform tasks using the immediate caller's class loader check, are invoked on untrusted objects returned by reflection calls. These Java APIs bypass the SecurityManager check that ensures all callers in the execution chain have the requisite security permissions. Care should be taken to ensure these APIs are not invoked on the untrusted objects returned by reflection as they can bypass security access checks and leave the system vulnerable to remote attacks. For more information on these Java APIs please refer to Guideline 9 of The Secure Coding Guidelines for the Java Programming Language.

Recommendation

The best way to prevent unsafe reflection is with a level of indirection: create a list of legitimate names that users are allowed to specify, and only allow users to select from the list. With this approach, input provided by users is never used directly to specify a name that is passed to the reflection API.

Reflection can also be used to create a custom data-driven architecture, whereby a configuration file determines the types and combinations of objects that are used by the application. This style of programming introduces the following security concerns:

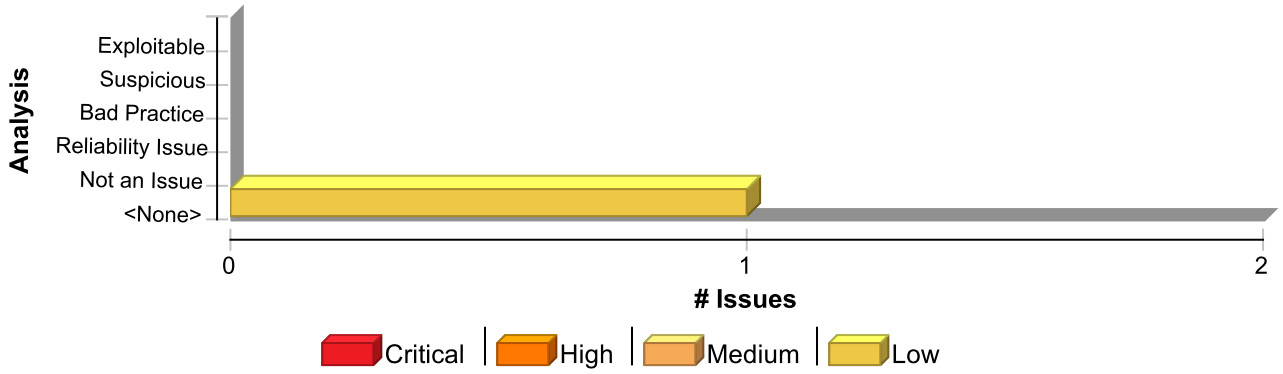
- The configuration file that controls the program is an essential part of the program's source code and must be protected and reviewed accordingly.

- Because the configuration file is unique to the application, unique work must be performed to evaluate the security of the design.

- Because the semantics of the application are now governed by a configuration file with a custom format, custom rules are required for obtaining optimal static analysis results.

For these reasons, avoid using this style of design unless your team can devote a large amount of effort to security evaluation.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unsafe Reflection	1	1	0	2
Total	**1**	**1**	**0**	**2**

Unsafe Reflection — Low

URL: /attack

DatabaseUtilities.java, line 66 (Unsafe Reflection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence	
AA_Prediction	Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.GenericServlet.getInitParameter()
From: org.owasp.webgoat.session.WebSession.WebSession
File: JavaSource/org/owasp/webgoat/session/WebSession.java:275
URL: /attack

```
272  showRequest = "true".equals( servlet.getInitParameter( SHOWREQUEST ) );
273  isDebug = "true".equals( servlet.getInitParameter( DEBUG ) );
274  databaseConnectionString = servlet.getInitParameter( DATABASE_CONNECTION_STRING );
275  databaseDriver = servlet.getInitParameter( DATABASE_DRIVER );
276  servletName = servlet.getServletName();
277  this.context = context;
278  course = new Course();
```

Sink Details

Sink: java.lang.Class.forName()
Enclosing Method: makeConnection()
File: DatabaseUtilities.java:66
Taint Flags: PROPERTY

```
63  public static Connection makeConnection(WebSession s)
```

Unsafe Reflection	Low

URL: /attack

DatabaseUtilities.java, line 66 (Unsafe Reflection)	Low

```
64 throws ClassNotFoundException, SQLException
65 {
66 Class.forName(s.getDatabaseDriver());
67
68 return (DriverManager.getConnection(s.getDatabaseConnectionString()));
69 }
```

Weak Cryptographic Hash (2 issues)

Abstract

Weak cryptographic hashes cannot guarantee data integrity and should not be used in security-critical contexts.

Explanation

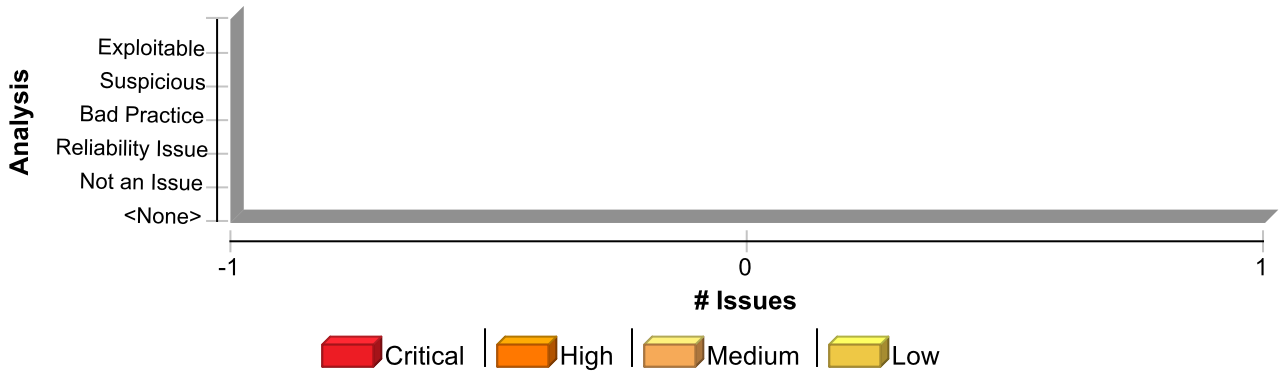
MD2, MD4, MD5, RIPEMD-160, and SHA-1 are popular cryptographic hash algorithms often used to verify the integrity of messages and other data. However, as recent cryptanalysis research has revealed fundamental weaknesses in these algorithms, they should no longer be used within security-critical contexts.

Effective techniques for breaking MD and RIPEMD hashes are widely available, so those algorithms should not be relied upon for security. In the case of SHA-1, current techniques still require a significant amount of computational power and are more difficult to implement. However, attackers have found the Achilles' heel for the algorithm, and techniques for breaking it will likely lead to the discovery of even faster attacks.

Recommendation

Discontinue the use of MD2, MD4, MD5, RIPEMD-160, and SHA-1 for data-verification in security-critical contexts. Currently, SHA-224, SHA-256, SHA-384, SHA-512, and SHA-3 are good alternatives. However, these variants of the Secure Hash Algorithm have not been scrutinized as closely as SHA-1, so be mindful of future research that might impact the security of these algorithms.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Weak Cryptographic Hash	2	2	0	4
Total	**2**	**2**	**0**	**4**

Weak Cryptographic Hash **Low**

Package: org.owasp.webgoat.lessons

Encoding.java, line 640 (Weak Cryptographic Hash) **Low**

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Cryptographic Hash — Low

Package: org.owasp.webgoat.lessons

Encoding.java, line 640 (Weak Cryptographic Hash) — Low

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getInstance()
Enclosing Method: hashMD5()
File: Encoding.java:640

```
637
638 try
639 {
640 md = MessageDigest.getInstance( "MD5" );
641 md.update( b );
642 }
643 catch ( NoSuchAlgorithmException e )
```

HttpOnly.java, line 175 (Weak Cryptographic Hash) — Low

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getInstance()
Enclosing Method: createCustomCookieValue()
File: HttpOnly.java:175

```
172 BASE64Encoder encoder = new BASE64Encoder();
173
174 try {
175 md = MessageDigest.getInstance("SHA");
176 buffer = new Date().toString().getBytes();
177
178 md.update(buffer);
```

Weak Cryptographic Hash: Insecure PBE Iteration Count (4 issues)

Abstract

The iteration count used by a password-based key derivation function is too low.

Explanation

A key derivation function is used to derive a key from a base key and other parameters. In a password-based key derivation function, the base key is a password and the other parameters are a salt value and an iteration count. An iteration count has traditionally served the purpose of increasing the cost of generating keys from a password. If the iteration count is too low, the feasibility of an attack increases as an attacker may compute "rainbow tables" for the application and more easily determine the hashed password values.

Example 1: The following code uses an iteration count of 50:

```
...
final int iterationCount=50;
PBEParameterSpec pbeps=new PBEParameterSpec(salt,iterationCount);
...
```

Applications that use a low iteration count for password-based encryption are vulnerable to trivial dictionary-based attacks -- exactly the type of attack that password-based encryption schemes were designed to protect against.

PBE (Password-Based Encryption) schemes

Best practices for use of `PBEParameterSpec` include: - use a non-constant salt - use at least 1000 iterations

Best practices for use of `PBEKeySpec` include: - use a non-constant salt - use at least 10000 iterations

See "An Empirical Study of Cryptographic Misuse in Android Applications" for a more thorough explanation.

This finding is from research found in "An Empirical Study of Cryptographic Misuse in Android Applications". http://www.cs.ucsb.edu/~chris/research/doc/ccs13_cryptolint.pdf and recommendations in "NIST Special Publication 800-132" http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf

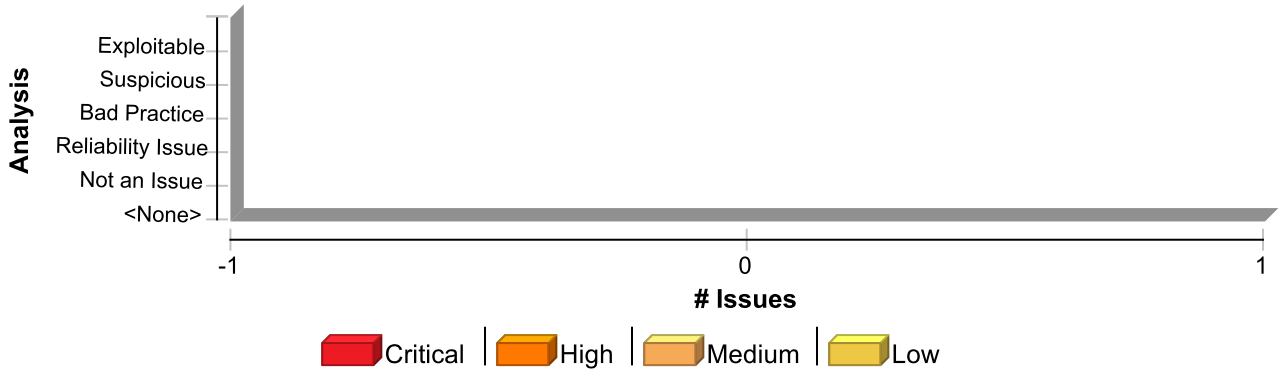
Recommendation

When using password-based key derivation functions, the iteration count should be at least 1,000. This will significantly increase the cost of an exhaustive search for passwords without a noticeable impact on the cost of deriving individual keys. NIST SP 800-132 recommends using an iteration count of as high as 10,000,000 for critical keys or very powerful systems.

Example 2: The following code uses an iteration count of 100,000:

```
...
final int iterationCount=100000;
PBEParameterSpec pbeps=new PBEParameterSpec(salt,iterationCount);
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Weak Cryptographic Hash: Insecure PBE Iteration Count	4	4	0	8
Total	**4**	**4**	**0**	**8**

Weak Cryptographic Hash: Insecure PBE Iteration Count — Medium

Package: org.owasp.webgoat.lessons

Encoding.java, line 539 (Weak Cryptographic Hash: Insecure PBE Iteration Count) — Medium

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: PBEKeySpec()
Enclosing Method: encryptString()
File: Encoding.java:539

```
536
537 char[] pass = pw.toCharArray();
538
539 SecretKey k = kf.generateSecret( new javax.crypto.spec.PBEKeySpec( pass ) );
540
541 passwordEncryptCipher.init( Cipher.ENCRYPT_MODE, k, ps );
542
```

Encoding.java, line 531 (Weak Cryptographic Hash: Insecure PBE Iteration Count) — Medium

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Weak Cryptographic Hash: Insecure PBE Iteration Count — Medium

Package: org.owasp.webgoat.lessons

Encoding.java, line 531 (Weak Cryptographic Hash: Insecure PBE Iteration Count) — Medium

Sink: PBEParameterSpec()
Enclosing Method: encryptString()
File: Encoding.java:531

```
528 try
529 {
530
531 PBEParameterSpec ps = new javax.crypto.spec.PBEParameterSpec( salt, 20 );
532
533 SecretKeyFactory kf = SecretKeyFactory.getInstance( "PBEWithMD5AndDES" );
534
```

Encoding.java, line 487 (Weak Cryptographic Hash: Insecure PBE Iteration Count) — Medium

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: PBEParameterSpec()
Enclosing Method: decryptString()
File: Encoding.java:487

```
484 try
485 {
486
487 PBEParameterSpec ps = new javax.crypto.spec.PBEParameterSpec( salt, 20 );
488
489 SecretKeyFactory kf = SecretKeyFactory.getInstance( "PBEWithMD5AndDES" );
490
```

Encoding.java, line 495 (Weak Cryptographic Hash: Insecure PBE Iteration Count) — Medium

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: PBEKeySpec()
Enclosing Method: decryptString()
File: Encoding.java:495

```
492
```

Weak Cryptographic Hash: Insecure PBE Iteration Count	Medium

Package: org.owasp.webgoat.lessons

Encoding.java, line 495 (Weak Cryptographic Hash: Insecure PBE Iteration Count)	Medium

```
493  char[] pass = pw.toCharArray();
494
495  SecretKey k = kf.generateSecret( new javax.crypto.spec.PBEKeySpec( pass ) );
496
497  passwordDecryptCipher.init( Cipher.DECRYPT_MODE, k, ps );
498
```

Weak Encryption (4 issues)

Abstract

The identified call uses a weak encryption algorithm that cannot guarantee the confidentiality of sensitive data.

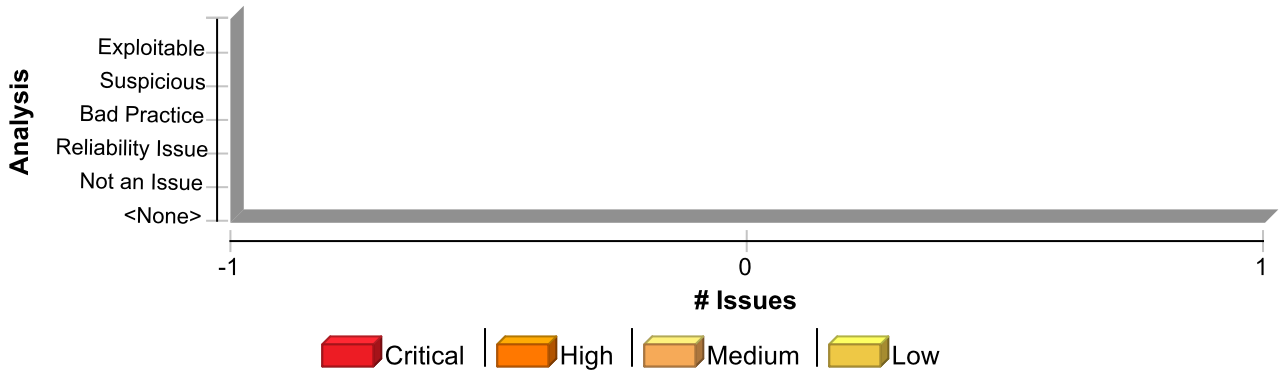
Explanation

Antiquated encryption algorithms such as DES no longer provide sufficient protection for use with sensitive data. Encryption algorithms rely on key size as one of the primary mechanisms to ensure cryptographic strength. Cryptographic strength is often measured by the time and computational power needed to generate a valid key. Advances in computing power have made it possible to obtain small encryption keys in a reasonable amount of time. For example, the 56-bit key used in DES posed a significant computational hurdle in the 1970's when the algorithm was first developed, but today DES can be cracked in less than a day using commonly available equipment.

Recommendation

Use strong encryption algorithms with large key sizes to protect sensitive data. Examples of strong alternatives to DES are Rijndael (Advanced Encryption Standard or AES) and Triple DES (3DES). Before selecting an algorithm, first determine if your organization has standardized on a specific algorithm and implementation.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Weak Encryption	4	4	0	8
Total	**4**	**4**	**0**	**8**

Weak Encryption — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 489 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction	Not Predicted

Weak Encryption — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 489 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: decryptString()
File: Encoding.java:489

```
486
487  PBEParameterSpec ps = new javax.crypto.spec.PBEParameterSpec( salt, 20 );
488
489  SecretKeyFactory kf = SecretKeyFactory.getInstance( "PBEWithMD5AndDES" );
490
491  Cipher passwordDecryptCipher = Cipher.getInstance( "PBEWithMD5AndDES/CBC/PKCS5Padding" );
492
```

Encoding.java, line 491 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getInstance()
Enclosing Method: decryptString()
File: Encoding.java:491

```
488
489  SecretKeyFactory kf = SecretKeyFactory.getInstance( "PBEWithMD5AndDES" );
490
491  Cipher passwordDecryptCipher = Cipher.getInstance( "PBEWithMD5AndDES/CBC/PKCS5Padding" );
492
493  char[] pass = pw.toCharArray();
494
```

Encoding.java, line 533 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getInstance()
Enclosing Method: encryptString()
File: Encoding.java:533

Weak Encryption — Critical

Package: org.owasp.webgoat.lessons

Encoding.java, line 533 (Weak Encryption) — Critical

```
530
531  PBEParameterSpec ps = new javax.crypto.spec.PBEParameterSpec( salt, 20 );
532
533  SecretKeyFactory kf = SecretKeyFactory.getInstance( "PBEWithMD5AndDES" );
534
535  Cipher passwordEncryptCipher = Cipher.getInstance( "PBEWithMD5AndDES/CBC/PKCS5Padding" );
536
```

Encoding.java, line 535 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Audit Details

AA_Prediction Not Predicted

Sink Details

Sink: getInstance()
Enclosing Method: encryptString()
File: Encoding.java:535

```
532
533  SecretKeyFactory kf = SecretKeyFactory.getInstance( "PBEWithMD5AndDES" );
534
535  Cipher passwordEncryptCipher = Cipher.getInstance( "PBEWithMD5AndDES/CBC/PKCS5Padding" );
536
537  char[] pass = pw.toCharArray();
538
```

XML Entity Expansion Injection (1 issue)

Abstract

Using XML parsers configured to not prevent nor limit Document Type Definition (DTD) entity resolution can expose the parser to an XML Entity Expansion injection

Explanation

XML Entity Expansion injection also known as XML Bombs are Denial of Service (DoS) attacks that benefit from valid and well-formed XML blocks that expand exponentially until they exhaust the server allocated resources. XML allows to define custom entities which act as string substitution macros. By nesting recurrent entity resolutions, an attacker may easily crash the server resources.

The following XML document shows an example of an XML Bomb.

```
<?xml version="1.0"?>
<!DOCTYPE lolz [
  <!ENTITY lol "lol">
  <!ENTITY lol2 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
  <!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">
  <!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;">
  <!ENTITY lol5 "&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;">
  <!ENTITY lol6 "&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;">
  <!ENTITY lol7 "&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;">
  <!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;">
  <!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;">
]>
<lolz>&lol9;</lolz>
```

This test could crash the server by expanding the small XML document into more than 3GB in memory.

Recommendation

An XML parser should be configured securely so that it does not allow document type definition (DTD) custom entities as part of an incoming XML document.

To avoid XML Entity Expansion injection the "secure-processing" property should be set for an XML factory, parser or reader:

```
factory.setFeature("http://javax.xml.XMLConstants/feature/secure-processing",
true);
```

In JAXP 1.3 and earlier versions, when the secure processing feature is on, default limitations are set for DOM and SAX parsers. These limits are:

```
entityExpansionLimit = 64,000
elementAttributeLimit = 10,000
```

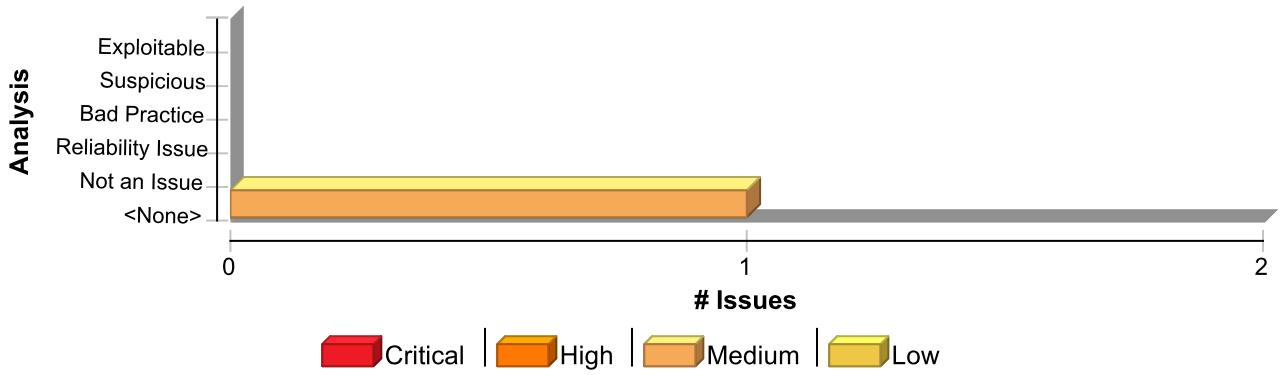
Since JAXP 1.4, the secure processing feature is turned on by default. In addition to the above limits, a new maxOccur limit is added to the validating parser. The limit is:

```
maxOccur = 5,000
```

If inline DOCTYPE declaration is not needed, it can be completely disabled with the following property:

```
factory.setFeature("http://apache.org/xml/features/disallow-doctype-decl",
true);
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
XML Entity Expansion Injection	1	1	0	2
Total	**1**	**1**	**0**	**2**

XML Entity Expansion Injection — Medium

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 179 (XML Entity Expansion Injection) — Medium

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624 public String getRawParameter(String name)
625 throws ParameterNotFoundException
626 {
627 String[] values = request.getParameterValues(name);
628
629 if (values == null)
630 {
```

Sink Details

Sink: org.xml.sax.XMLReader.parse()
Enclosing Method: checkXML()
File: WsSAXInjection.java:179

XML Entity Expansion Injection	Medium
Package: org.owasp.webgoat.lessons	
WsSAXInjection.java, line 179 (XML Entity Expansion Injection)	**Medium**

Taint Flags: WEB, XSS

```
176 XMLReader reader = XMLReaderFactory.createXMLReader();
177 PasswordChanger changer = new PasswordChanger();
178 reader.setContentHandler(changer);
179 reader.parse(new InputSource(new StringReader(xml)));
180 if (!"101".equals(changer.getId()))
181 {
182 makeSuccess(s);
```

XML External Entity Injection (1 issue)

Abstract

Using XML parsers configured to not prevent nor limit external entities resolution can expose the parser to an XML External Entities attack

Explanation

XML External Entities attacks benefit from an XML feature to build documents dynamically at the time of processing. An XML entity allows inclusion of data dynamically from a given resource. External entities allow an XML document to include data from an external URI. Unless configured to do otherwise, external entities force the XML parser to access the resource specified by the URI, e.g., a file on the local machine or on a remote system. This behavior exposes the application to XML External Entity (XXE) attacks, which can be used to perform denial of service of the local system, gain unauthorized access to files on the local machine, scan remote machines, and perform denial of service of remote systems.

The following XML document shows an example of an XXE attack.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///dev/random" >]><foo>&xxe;</foo>
```

This example could crash the server (on a UNIX system), if the XML parser attempts to substitute the entity with the contents of the /dev/random file.

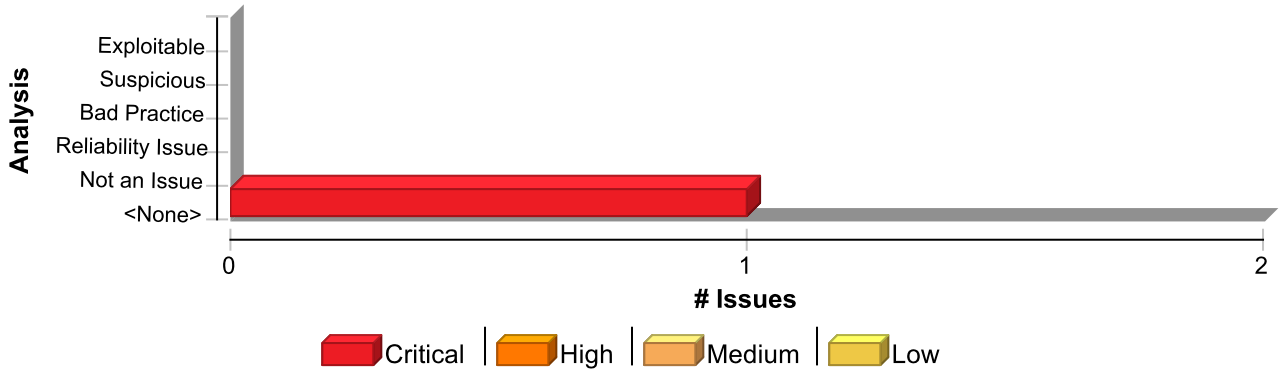
Recommendation

The XML unmarshaller should be configured securely so that it does not allow external entities as part of an incoming XML document.

To avoid XXE injection do not use unmarshal methods that process an XML source directly as java.io.File, java.io.Reader or java.io.InputStream. Parse the document with a securely configured parser and use an unmarshal method that takes the secure parser as the XML source as shown in the following example:

```
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
dbf.setFeature("http://apache.org/xml/features/disallow-doctype-decl", true);
DocumentBuilder db = dbf.newDocumentBuilder();
Document document = db.parse(<XML Source>);
Model model = (Model) u.unmarshal(document);
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
XML External Entity Injection	1	1	0	2
Total	**1**	**1**	**0**	**2**

XML External Entity Injection — Critical

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 179 (XML External Entity Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Not An Issue threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: org.xml.sax.XMLReader.parse()
Enclosing Method: checkXML()
File: WsSAXInjection.java:179
Taint Flags: WEB, XSS

```
176  XMLReader reader = XMLReaderFactory.createXMLReader();
```

XML External Entity Injection — Critical

Package: org.owasp.webgoat.lessons

WsSAXInjection.java, line 179 (XML External Entity Injection) — Critical

```
177 PasswordChanger changer = new PasswordChanger();
178 reader.setContentHandler(changer);
179 reader.parse(new InputSource(new StringReader(xml)));
180 if (!"101".equals(changer.getId()))
181 {
182 makeSuccess(s);
```

XPath Injection (1 issue)

Abstract

Constructing a dynamic XPath query with user input could allow an attacker to modify the statement's meaning.

Explanation

XPath injection occurs when:

1. Data enters a program from an untrusted source.

2. The data used to dynamically construct an XPath query.

Example 1: The following code dynamically constructs and executes an XPath query that retrieves an e-mail address for a given account ID. The account ID is read from an HTTP request, and is therefore untrusted.

```
...
String acctID = request.getParameter("acctID");
String query = null;
if(acctID != null) {
       StringBuffer sb = new StringBuffer("/accounts/account[acctID='");
       sb.append(acctID);
       sb.append("']/email/text()");
       query = sb.toString();
}

DocumentBuilderFactory domFactory = DocumentBuilderFactory.newInstance();
domFactory.setNamespaceAware(true);
DocumentBuilder builder = domFactory.newDocumentBuilder();
Document doc = builder.parse("accounts.xml");
XPathFactory factory = XPathFactory.newInstance();
XPath xpath = factory.newXPath();
XPathExpression expr = xpath.compile(query);
Object result = expr.evaluate(doc, XPathConstants.NODESET);
...
```

Under normal conditions, such as searching for an e-mail address that belongs to the account number 1, the query that this code executes will look like the following:

```
/accounts/account[acctID='1']/email/text()
```

However, because the query is constructed dynamically by concatenating a constant base query string and a user input string, the query only behaves correctly if `acctID` does not contain a single-quote character. If an attacker enters the string `1' or '1' = '1` for `acctID`, then the query becomes the following:

```
/accounts/account[acctID='1' or '1' = '1']/email/text()
```

The addition of the `1' or '1' = '1` condition causes the where clause to always evaluate to true, so the query becomes logically equivalent to the much simpler query:

```
//email/text()
```

This simplification of the query allows the attacker to bypass the requirement that the query only return items owned

by the authenticated user; the query now returns all e-mail addresses stored in the document, regardless of their specified owner.

Recommendation

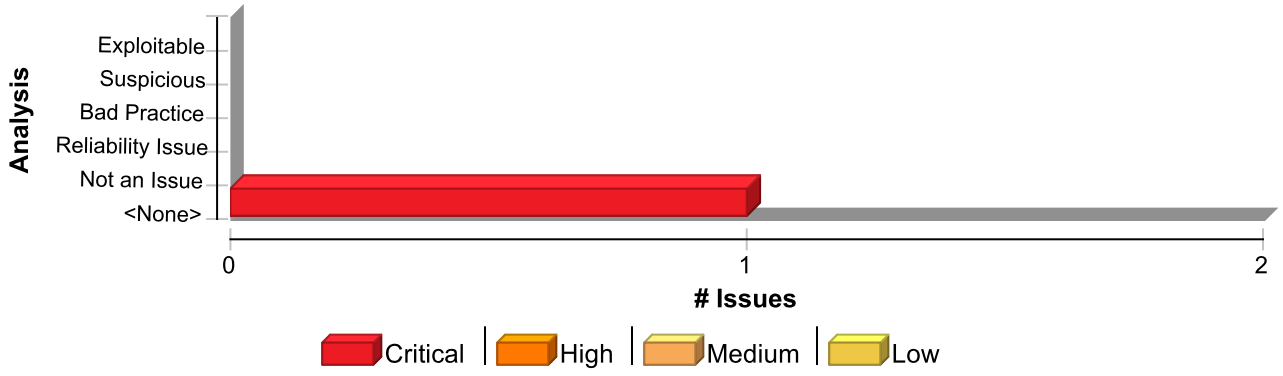
The root cause of XPath injection vulnerability is the ability of an attacker to change context in the XPath query, causing a value that the programmer intended to be interpreted as data to be interpreted as a command instead. When an XPath query is constructed, the programmer knows what should be interpreted as part of the command and what should be interpreted as data.

To prevent an attacker from violating the programmer's expectations, use a whitelist to ensure that user-controlled values used in an XPath query are composed from only the expected set of characters and do not contain any XPath metacharacters given the context in which they are used. If a user-controlled value requires that it contain XPath metacharacters, use an appropriate encoding mechanism to remove their significance within the XPath query.

Example 2

```
...
String acctID = request.getParameter("acctID");
String query = null;
if(acctID != null) {
       Integer iAcctID = -1;
       try {
              iAcctID = Integer.parseInt(acctID);
       }
       catch (NumberFormatException e) {
              throw new InvalidParameterException();
       }
       StringBuffer sb = new StringBuffer("/accounts/account[acctID='");
       sb.append(iAcctID.toString());
       sb.append("']/email/text()");
       query = sb.toString();
}

DocumentBuilderFactory domFactory = DocumentBuilderFactory.newInstance();
domFactory.setNamespaceAware(true);
DocumentBuilder builder = domFactory.newDocumentBuilder();
Document doc = builder.parse("accounts.xml");
XPathFactory factory = XPathFactory.newInstance();
XPath xpath = factory.newXPath();
XPathExpression expr = xpath.compile(query);
Object result = expr.evaluate(doc, XPathConstants.NODESET);
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
XPath Injection	1	1	0	2
Total	**1**	**1**	**0**	**2**

XPath Injection — Critical

Package: org.owasp.webgoat.lessons

XPATHInjection.java, line 158 (XPath Injection) — Critical

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Data Flow)

Audit Details

AA_Confidence
AA_Prediction Indeterminate (Below Exploitable threshold)

Source Details

Source: javax.servlet.ServletRequest.getParameterValues()
From: org.owasp.webgoat.session.ParameterParser.getRawParameter
File: JavaSource/org/owasp/webgoat/session/ParameterParser.java:627

```
624  public String getRawParameter(String name)
625  throws ParameterNotFoundException
626  {
627  String[] values = request.getParameterValues(name);
628
629  if (values == null)
630  {
```

Sink Details

Sink: javax.xml.xpath.XPath.evaluate()
Enclosing Method: createContent()
File: XPATHInjection.java:158
Taint Flags: WEB, XSS

```
155  InputSource inputSource = new InputSource(new FileInputStream(d));
156  String expression = "/employees/employee[loginID/text()='"
157  + username + "' and passwd/text()='" + password + "']";
158  nodes = (NodeList) xPath.evaluate(expression, inputSource,
159  XPathConstants.NODESET);
160  int nodesLength = nodes.getLength();
161
```

Description of Key Terminology

Likelihood and Impact

Likelihood
Likelihood is the probability that a vulnerability will be accurately identified and successfully exploited.

Impact
Impact is the potential damage an attacker could do to assets by successfully exploiting a vulnerability. This damage can be in the form of, but not limited to, financial loss, compliance violation, loss of brand reputation, and negative publicity.

Fortify Priority Order

Critical
Critical-priority issues have high impact and high likelihood. Critical-priority issues are easy to detect and exploit and result in large asset damage. These issues represent the highest security risk to the application. As such, they should be remediated immediately.

SQL Injection is an example of a critical issue.

High
High-priority issues have high impact and low likelihood. High-priority issues are often difficult to detect and exploit, but can result in large asset damage. These issues represent a high security risk to the application. High-priority issues should be remediated in the next scheduled patch release.

Password Management: Hardcoded Password is an example of a high issue.

Medium
Medium-priority issues have low impact and high likelihood. Medium-priority issues are easy to detect and exploit, but typically result in small asset damage. These issues represent a moderate security risk to the application. Medium-priority issues should be remediated in the next scheduled product update.

Path Manipulation is an example of a medium issue.

Low
Low-priority issues have low impact and low likelihood. Low-priority issues can be difficult to detect and exploit and typically result in small asset damage. These issues represent a minor security risk to the application. Low-priority issues should be remediated as time allows.

Dead Code is an example of a low issue.

About Fortify Solutions

Fortify is the leader in end-to-end application security solutions with the flexibility of testing on-premise and on-demand to cover the entire software development lifecycle. Learn more at software.microfocus.com/en-us/solutions/application-security.

Fortify Professional Services

Fortify Professional Services takes a holistic approach to building and operating cyber security and response solutions and capabilities that support the cyber threat management and regulatory compliance needs of the world's largest enterprises. We use a combination of operational expertise (yours and ours) and proven methodologies to deliver fast, effective results and demonstrate ROI. Our proven, use-case driven solutions combine market-leading technology together with sustainable business and technical process executed by trained and organized people. Learn more about Fortify Professional Services at software.microfocus.com/en-us/services/enterprise-security-consulting-services.