opentext™

kisa49

5/31/24

ikkb7

# Executive Summary

## Issues Overview

On May 31, 2024, a source code review was performed over the LoginProject code base. 15 files, 673 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 28 reviewed findings were uncovered during the analysis.

| Issues by 49 | |
|---|---|
| <none> | 16 |
| 04.01. | 7 |
| 02.09. | 1 |
| 01.11. | 3 |
| 01.04. | 1 |

## Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level.  The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

## Project Summary

### Code Base Summary

Code location: C:/Users/ikkb7/Fortify_SCA_Samples_24.2.0/advanced/javaWebApp/LoginProject
Number of Files: 15
Lines of Code: 673
Build Label: <No Build Label>

### Scan Information

Scan time: 01:02
SCA Engine version: 24.2.0.0150
Machine Name: kbkim-surface4
Username running scan: ikkb7

### Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

### Attack Surface

Attack Surface:
Java Properties:
javax.servlet.ServletContext.getInitParameter

System Information:
null.null.null
javax.servlet.ServletContext.getInitParameter

Web:
null.null.null

### Filter Set Summary

Current Enabled Filter Set:
Security Auditor View

Filter Set Details:

Folder Filters:
If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High

If [fortify priority order] contains medium Then set folder to Medium

If [fortify priority order] contains low Then set folder to Low

## Audit Guide Summary

Audit guide not enabled

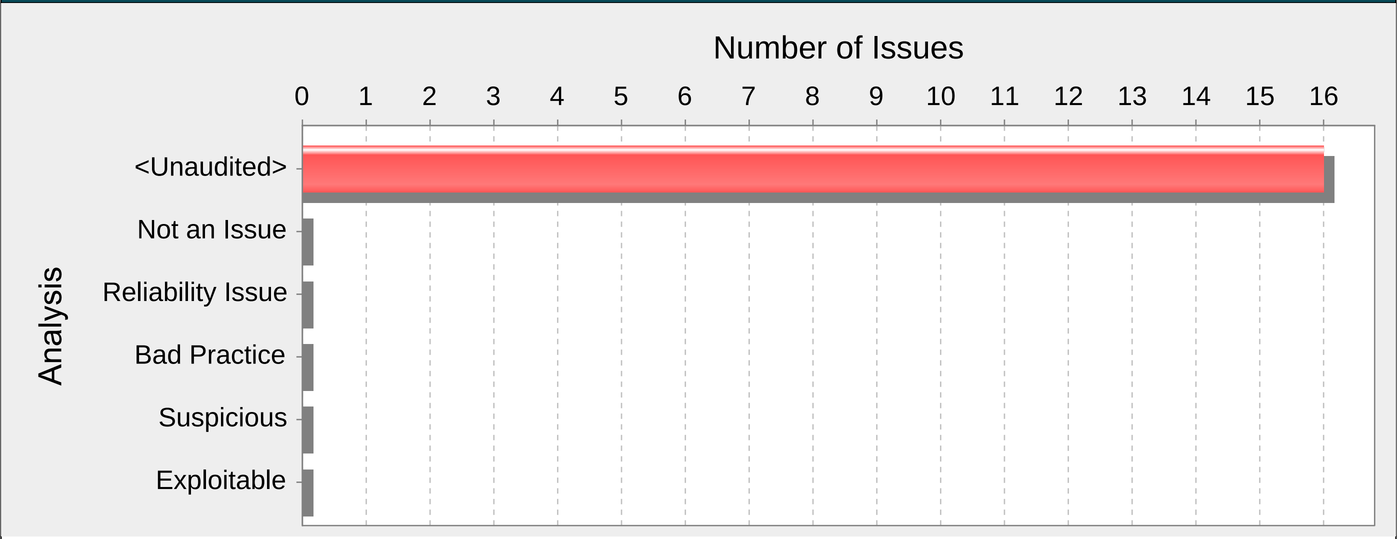
## Results Outline

### Overall number of results

The scan found 28 issues.

### Vulnerability Examples by Category

### 49: <none> (16 Issues)



**Abstract:**

login.html 19 .

**Explanation:**

, .

**Recommendations:**

. . " " .

1: HTML form autocomplete off .

<form method="post" autocomplete="off">
Address: <input name="address" />
Password: <input name="password" type="password" />
</form>

2: autocomplete off .

<form method="post">
Address: <input name="address" />
Password: <input name="password" type="password" autocomplete="off"/>
</form>

autocomplete on . , .

### registerServlet.java, line 64 (SQL Injection)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Critical | **Folder** | Critical |
| **Kingdom:** | Input Validation and Representation | | |

**Abstract:** registerServlet.java 64 doPost() SQL . SQL .

**Source:** registerServlet.java:44 javax.servlet.ServletRequest.getParameter()

```
42      //doGet(request, response);
43      String sno = request.getParameter("Sno");
44      String user = request.getParameter("LoginUser");
45      String pass = request.getParameter("LoginPass");
46      Connection conn = (Connection)(getServletContext().getAttribute("conn"));
```

**Sink:** registerServlet.java:64 java.sql.Statement.executeUpdate()

```
62        System.out.println("f = true");
```

```
63              sql = "insert into user values('" + sno + "', '" + user + "', '" + pass + "')";
64              state.executeUpdate(sql);
65
66             }
```

## web.xml, line 2 (J2EE Misconfiguration: Missing Error Handling)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Environment | | |

**Abstract:** .

**Sink:** web.xml:2 /web-app()

```
0    <?xml version="1.0" encoding="UTF-8"?>
1    <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns="http://java.sun.com/xml/ns/javaee"
     xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
     http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" id="WebApp_ID" version="3.0">
2      <display-name>LoginProject</display-name>
3      <welcome-file-list>
```

## registerServlet.java, line 62 (Poor Logging Practice: Use of a System Output Stream)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** println() .

**Sink:** registerServlet.java:62 FunctionCall: println()

```
60             if(f){
61              res = "true";
62              System.out.println("f = true");
63              sql = "insert into user values('" + sno + "', '" + user + "', '" + pass + "')";
64              state.executeUpdate(sql);
```

## register.html, line 40 (Privacy Violation: Autocomplete)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | High | **Folder** | High |
| **Kingdom:** | Security Features | | |

**Abstract:** register.html 40 .

**Sink:** register.html:40

```
38      <span>: <input onmouseout = "UserCheck()" id="User" type="text" name="LoginUser"
     ></input><p id="2"></p><br/><br/></span>
39      <span>: <input onmouseout = "PassCheck()" id="Pass" type="password"
     name="LoginPass" ></input><br/><br/><p id="3"></p></span>
40      <span style="margin-left: 218px;">: <input onmouseout = "PassTwoCheck()" id =
     "PassTwo" type="password" name="PassTwo" ></input><br/><br/><p id="4"></p></span>
41     <br/><br/><br/>
42      <span><input id = "submit" disabled style="width: 452px;color: #fff;background:
     #999;" type="submit" value=""/></span>
```

## LoginModel.java, line 9 (SQL Injection)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Input Validation and Representation | | |

**Abstract:** LoginModel.java 9 check() SQL . SQL .

**Sink:** LoginModel.java:9 executeQuery()

```
7     public String check(Connection conn, String user, String pass) throws SQLException{
8       state = conn.createStatement();
9       rs = state.executeQuery(sql);
10      while(rs.next()){
11       if(rs.getString(2).equals(user)){
```

## MyContextListener.java, line 30 (Poor Logging Practice: Use of a System Output Stream)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** println() .

**Sink:** MyContextListener.java:30 FunctionCall: println()

```
28                      Connection conn = DriverManager.getConnection(url, user, pass);
29                      sc.setAttribute("conn", conn);
30                      System.out.println("Connect Succeed!");
31                     }
32                     catch(Exception e){
```

## login.jsp, line 28 (Privacy Violation: Autocomplete)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | High | **Folder** | High |
| **Kingdom:** | Security Features | | |
| **Abstract:** | login.jsp  28          . | | |

**Sink:** login.jsp:28

```
26                           <center>ç¨æ·ç»å½</center><br/><br/>
27                           USER: <br/> <input type="text" name="user"></input><br/><br/>
28                           PASS: <br/><input type="password" name="pass"></input>
29                           <br/>
30                           <br/>
```

## registerServlet.java, line 64 (SQL Injection)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Critical | **Folder** | Critical |
| **Kingdom:** | Input Validation and Representation | | |
| **Abstract:** | registerServlet.java 64  doPost()        SQL  .        SQL   . | | |

**Source:** registerServlet.java:45 javax.servlet.ServletRequest.getParameter()

```
43                     String sno = request.getParameter("Sno");
44                     String user = request.getParameter("LoginUser");
45                     String pass = request.getParameter("LoginPass");
46                     Connection conn = (Connection)(getServletContext().getAttribute("conn"));
47                     try{
```

**Sink:** registerServlet.java:64 java.sql.Statement.executeUpdate()

```
62                       System.out.println("f = true");
63                       sql = "insert into user values('" + sno + "', '" + user + "', '" + pass + "')";
64                       state.executeUpdate(sql);
65
66                      }
```

## registerServlet.java, line 64 (SQL Injection)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Critical | **Folder** | Critical |
| **Kingdom:** | Input Validation and Representation | | |
| **Abstract:** | registerServlet.java 64  doPost()        SQL  .        SQL   . | | |

**Source:** registerServlet.java:43 javax.servlet.ServletRequest.getParameter()

```
41                     // TODO Auto-generated method stub
42                     //doGet(request, response);
43                     String sno = request.getParameter("Sno");
44                     String user = request.getParameter("LoginUser");
45                     String pass = request.getParameter("LoginPass");
```

**Sink:** registerServlet.java:64 java.sql.Statement.executeUpdate()

```
62                       System.out.println("f = true");
63                       sql = "insert into user values('" + sno + "', '" + user + "', '" + pass + "')";
64                       state.executeUpdate(sql);
65
66                      }
```

## MyContextListener.java, line 27 (Unsafe Reflection)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Input Validation and Representation | | |
| **Abstract:** | MyContextListener.java 27  forName()     .             . | | |

**Source:** MyContextListener.java:22 javax.servlet.ServletContext.getInitParameter()

```
20                     // TODO Auto-generated method stub
21                     ServletContext sc = event.getServletContext();
```

```
22          String driver = sc.getInitParameter("driver");
23          String url = sc.getInitParameter("url");
24          String user = sc.getInitParameter("user");
```

**Sink:** MyContextListener.java:27 java.lang.Class.forName()

```
25          String pass = sc.getInitParameter("pass");
26          try{
27           Class.forName(driver);//, jdbc,
28           Connection conn = DriverManager.getConnection(url, user, pass);
29           sc.setAttribute("conn", conn);
```

## register.html, line 39 (Privacy Violation: Autocomplete)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | High | **Folder** | High |
| **Kingdom:** | Security Features | | |

**Abstract:** register.html 39 .

**Sink:** register.html:39

```
37            <span>: <input onmouseout = "SnoCheck()"  id="Sno"  type="text" name="Sno" /><p
id="1"></p><br/><br/></span>
38            <span>: <input onmouseout = "UserCheck()" id="User" type="text" name="LoginUser"
></input><p id="2"></p><br/><br/></span>
39            <span>: <input onmouseout = "PassCheck()" id="Pass" type="password"
name="LoginPass" ></input><br/><br/><p id="3"></p></span>
40            <span style="margin-left: 218px;">: <input onmouseout = "PassTwoCheck()" id =
"PassTwo" type="password" name="PassTwo" ></input><br/><br/><p id="4"></p></span>
41           <br/><br/><br/>
```

## LoginServlet.java, line 48 (Trust Boundary Violation)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** LoginServlet.java doPost() . .

**Source:** LoginModel.java:9 java.sql.Statement.executeQuery()

```
7          public String check(Connection conn, String user, String pass) throws SQLException{
8           state = conn.createStatement();
9           rs = state.executeQuery(sql);
10          while(rs.next()){
11           if(rs.getString(2).equals(user)){
```

**Sink:** LoginServlet.java:48 javax.servlet.ServletRequest.setAttribute()

```
46          try{
47           String res = obj.check(conn, user, pass);
48           request.setAttribute("res", res);
49          // request.setAttribute("Sno", sno);
50           RequestDispatcher view = request.getRequestDispatcher("login.jsp");
```

## MyContextListener.java, line 28 (J2EE Bad Practices: getConnection())

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | API Abuse | | |

**Abstract:** J2EE .

**Sink:** MyContextListener.java:28 getConnection()

```
26          try{
27           Class.forName(driver);//, jdbc,
28           Connection conn = DriverManager.getConnection(url, user, pass);
29           sc.setAttribute("conn", conn);
30           System.out.println("Connect Succeed!");
```

## login.html, line 19 (Privacy Violation: Autocomplete)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | High | **Folder** | High |
| **Kingdom:** | Security Features | | |

**Abstract:** login.html 19 .

**Sink:** login.html:19

```
17                <center></center><br/><br/>
```

```
                              USER: <br/> <input type="text" name="user"></input><br/><br/>
                              PASS: <br/><input type="password" name="pass"></input>
                              <br/>
                              <br/>
```

## web.xml, line 2 (J2EE Misconfiguration: Excessive Session Timeout)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Environment | | |
| **Abstract:** | . | | |
| **Sink:** | web.xml:2 | | |

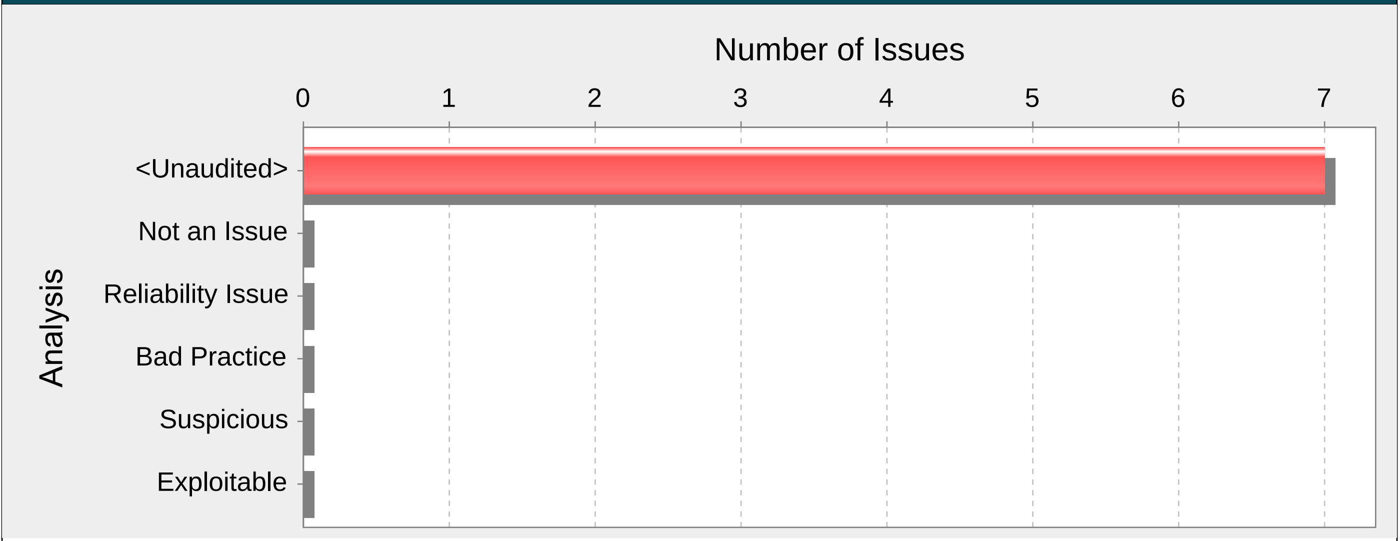
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" id="WebApp_ID" version="3.0">
  <display-name>LoginProject</display-name>
  <welcome-file-list>
```

## LoginServlet.java, line 40 (Poor Logging Practice: Use of a System Output Stream)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |
| **Abstract:** | println() . | | |
| **Sink:** | LoginServlet.java:40 FunctionCall: println() | | |

```
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
  // TODO Auto-generated method stub
  System.out.println("Login Test");
  String user = request.getParameter("user");
  String pass = request.getParameter("pass");
```

## 49: 04.01. (7 Issues)



**Abstract:**

LoginServlet Servlet doGet() . Servlet .

**Explanation:**

Servlet Servlet . . , SQL , . .

1: DNS Servlet .

```
protected void doPost (HttpServletRequest req,
HttpServletResponse res)
throws IOException {
String ip = req.getRemoteAddr();
InetAddress addr = InetAddress.getByName(ip);
...
out.println("hello " + addr.getHostName());
}
```

2: "name" NullPointerException .

```
protected void doPost (HttpServletRequest req,
HttpServletResponse res)
throws IOException {
String name = getParameter("name");
...
out.println("hello " + name.trim());
}
```

**Recommendations:**

Servlet Throwable (catch) Servlet .

3: Example 1 .

```
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);
}
}
}
```

## registerServlet.java, line 40 (System Information Leak: Incomplete Servlet Error Handling)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** registerServlet Servlet doPost()  . Servlet  .

**Sink:** registerServlet.java:40 Function: doPost()

```
38       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
39       */
40      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
     ServletException, IOException {
41       // TODO Auto-generated method stub
42       //doGet(request, response);
```

## registerServlet.java, line 72 (System Information Leak)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** registerServlet.java doPost() 72 printStackTrace()()  . printStackTrace()  .

**Sink:** registerServlet.java:72 printStackTrace()

```
70       }
71       catch(Exception e){
72        e.printStackTrace();
73       }
74      }
```

## LoginServlet.java, line 54 (System Information Leak)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** LoginServlet.java doPost() 54 printStackTrace()()  . printStackTrace()  .

**Sink:** LoginServlet.java:54 printStackTrace()

```
52       }
53       catch(Exception e){
54        e.printStackTrace();
55       }
56      }
```

## LoginServlet.java, line 30 (System Information Leak: Incomplete Servlet Error Handling)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** LoginServlet Servlet doGet()  . Servlet  .

**Sink:** LoginServlet.java:30 Function: doGet()

```
28       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
29       */
30      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
     ServletException, IOException {
31       // TODO Auto-generated method stub
32       response.getWriter().append("Served at: ").append(request.getContextPath());
```

## registerServlet.java, line 32 (System Information Leak: Incomplete Servlet Error Handling)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** registerServlet Servlet doGet()  . Servlet  .

**Sink:** registerServlet.java:32 Function: doGet()

```
30       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
31       */
32      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
     ServletException, IOException {
33       // TODO Auto-generated method stub
34       response.getWriter().append("Served at: ").append(request.getContextPath());
```

## MyContextListener.java, line 33 (System Information Leak)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** MyContextListener.java contextInitialized()  33  printStackTrace()()       .
printStackTrace()            .

**Sink:** MyContextListener.java:33 printStackTrace()

```
31        }
32        catch(Exception e){
33         e.printStackTrace();
34        }
35       }
```

## LoginServlet.java, line 38 (System Information Leak: Incomplete Servlet Error Handling)

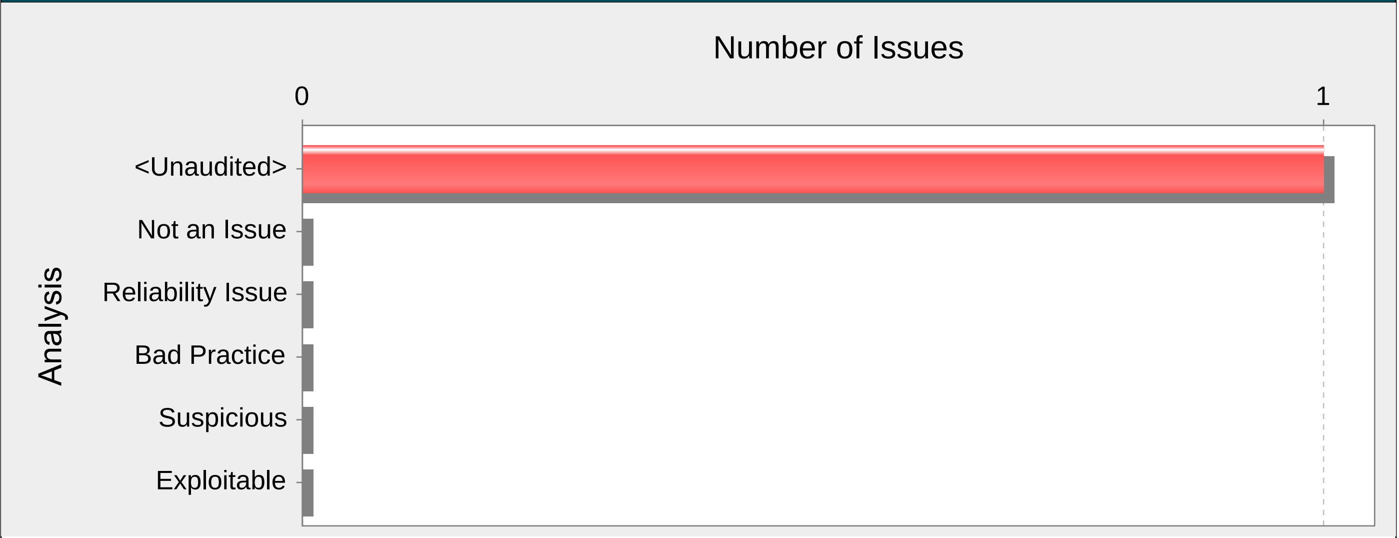
| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |

**Abstract:** LoginServlet Servlet doPost()      . Servlet                .

**Sink:** LoginServlet.java:38 Function: doPost()

```
36        * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
37        */
38       protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
       ServletException, IOException {
39        // TODO Auto-generated method stub
40        System.out.println("Login Test");
```

## 49: 02.09. (1 Issues)



**Abstract:**

MyContextListener.java contextInitialized() 28 . .

**Explanation:**

Password management .

1: .

```
...
Properties prop = new Properties();
prop.load(new FileInputStream("config.properties"));
String password = prop.getProperty("password");
DriverManager.getConnection(url, usr, password);
...
```

config.properties password . .

.

2: Android WebView .

```
...
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];
handler.proceed(username, password);
}
});
...
```

WebView . .

**Recommendations:**

. . . .

. , WebSphere Application Server 4.x XOR . WebSphere . .

Android SQLite SQLCipher . SQLCipher SQLite , 256 AES . .

3: SQLCipher Android .

```
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});
...
```

android.database.sqlite.SQLiteDatabase net.sqlcipher.database.SQLiteDatabase .

WebView sqlcipher.so WebKit .

**Tips:**

1. Fortify Secure Coding Rulepacks . Fortify Static Code Analyzer .

Password Management . . . .

(pass-through rule) . Password Management .

2. (Struts Struts 2 ). , Fortify Fortify Static Code Analyzer . Context-Sensitive Ranking( ) . Fortify , Fortify Software Security Research Group .

## MyContextListener.java, line 28 (Password Management)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Security Features | | |

**Abstract:** MyContextListener.java contextInitialized() 28 . .

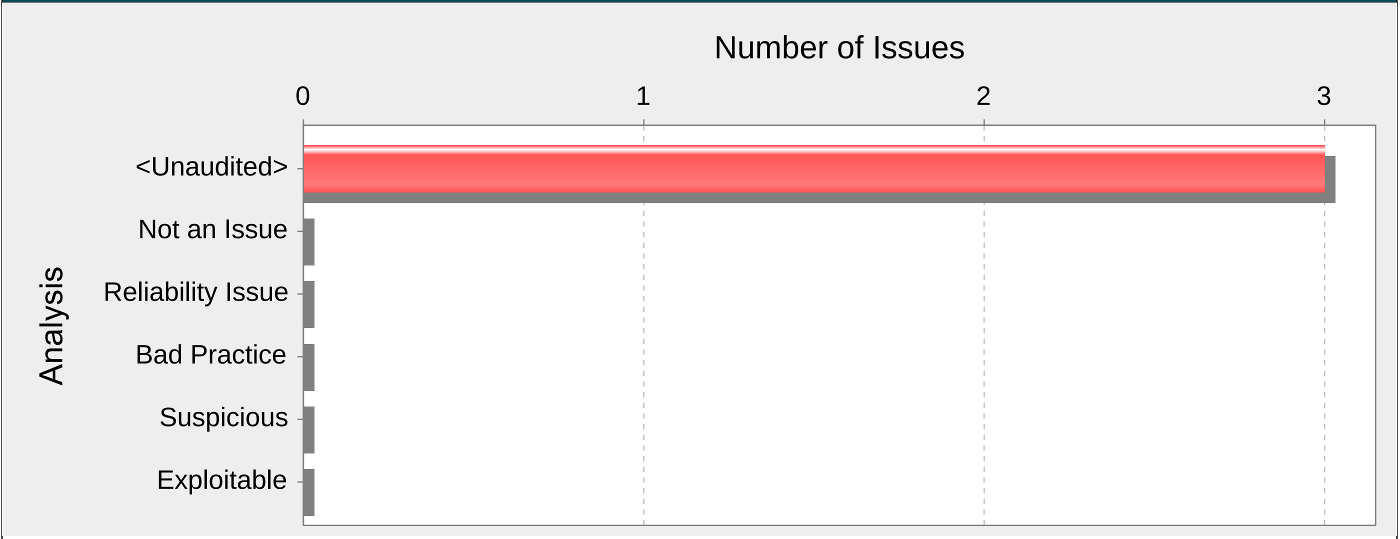
**Source:** MyContextListener.java:25 javax.servlet.ServletContext.getInitParameter()

```
23      String url = sc.getInitParameter("url");
24      String user = sc.getInitParameter("user");
25      String pass = sc.getInitParameter("pass");
26      try{
27       Class.forName(driver);//, jdbc,
```

**Sink:** MyContextListener.java:28 java.sql.DriverManager.getConnection()

```
26      try{
27       Class.forName(driver);//, jdbc,
28       Connection conn = DriverManager.getConnection(url, user, pass);
29       sc.setAttribute("conn", conn);
30       System.out.println("Connect Succeed!");
```

## 49: 01.11. (3 Issues)



### Abstract:

login.html 16 .

### Explanation:

CSRF(cross-site request forgery) .

1. .

2. HTTP .

Nonce . , CSRF ( ). . .

```
<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>
```

.

```
<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>
```

example.com . CSRF . . . .

ID URL ID CSRF .

CSRF 2007 OWASP Top 10 5.

### Recommendations:

. ID Nonce .

```
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));
```

ID . ID . ID ID CSRF . (: SSLv3 ) .

.

: CSRF CSRF .

- : CSRF . CAPTCHA, , .
HTTP Referer/Origin : CSRF . CSRF .
: ID ID CSRF . . ID .
: CSRF ID . .

XSS . CSRF XSS .

**Tips:**

1. Fortify Static Code Analyzer GET POST HTML XMLHttpRequest . CSRF .

## login.jsp, line 25 (Cross-Site Request Forgery)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |
| **Abstract:** | login.jsp 25 . | | |

**Sink:** login.jsp:25

```
      window.onload=show;
    </script>
    <form style="margin:auto;" action="login.do" method="POST">
        <center>ç¨æ·ç»å½</center><br/><br/>
        USER: <br/> <input type="text" name="user"></input><br/><br/>
```

## login.html, line 16 (Cross-Site Request Forgery)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |
| **Abstract:** | login.html 16 . | | |

**Sink:** login.html:16

```
        </div>    -->

        <form  style="margin: auto;" action="login.do" method="POST">
            <center></center><br/><br/>
            USER: <br/> <input type="text" name="user"></input><br/><br/>
```

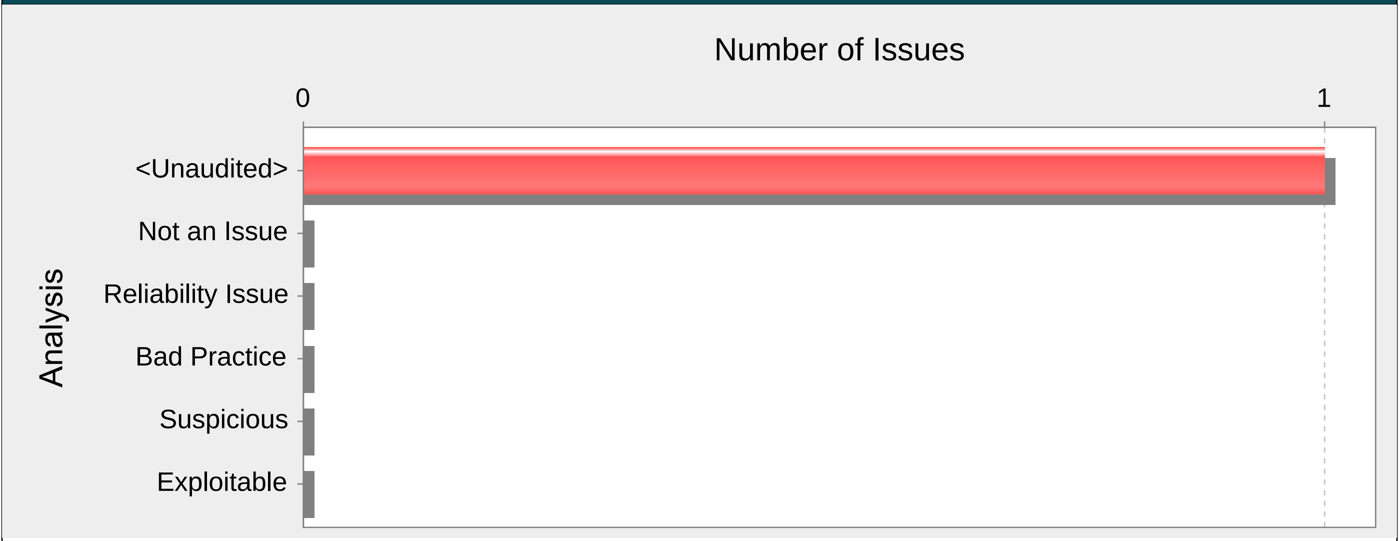
## register.html, line 36 (Cross-Site Request Forgery)

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Low | **Folder** | Low |
| **Kingdom:** | Encapsulation | | |
| **Abstract:** | register.html 36 . | | |

**Sink:** register.html:36

```
            <center><h1></h1></center><br/>
        <br/><br/>
        <form action="register.do" method="POST">
        <span>: <input onmouseout = "SnoCheck()"  id="Sno"  type="text" name="Sno" /><p
id="1"></p><br/><br/></span>
        <span>: <input onmouseout = "UserCheck()" id="User" type="text" name="LoginUser"
></input><p id="2"></p><br/><br/></span>
```

49: 01.04. (1 Issues)



**Abstract:**

login.jsp _jspService() 19 .

**Explanation:**

XSS(Cross-site scripting) .

1. . Persistent(Stored ) XSS , Reflected XSS .

2. .

JavaScript HTML, Flash . XSS , .

1: JSP ID .

```
<%...
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 %>
```

name . name . name . . Persistent( Stored) XSS . XSS "" . JavaScript .

2: JSP HTTP ID eid .

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

Example 1 eid . eid , HTTP .

. URL ? URL URL . . Reflected XSS .

Cross-Site Scripting . . . .

3: Android WebView JavaScript ( JavaScript ) Android .

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

...

url javascript: JavaScript WebView .

, XSS HTTP . XSS .

- Example 1 . . Persistent XSS . . . .

- Example 2 HTTP HTTP . XSS . . URL . URL URL . , .

- Example 3 .

(Struts Spring MVC ). , Fortify Fortify Static Code Analyzer . Context-Sensitive Ranking( ) . Fortify , Fortify Software Security Research Group .

**Recommendations:**

XSS .

XSS . ( ) . XSS .

SQL injection . XSS . XSS . . . , XSS .

XSS HTTP . , 0-9 . . HTML .

. . . HTML HTML . XSS . SEI(Software Engineering Institute) CERT(R) Coordination Center [1].

Block-level element ( ):

- "<" .

- "&" .

- ">" "<" .

.

- .

- .

- .

- "&" .

, URL . URL .

- , URL .

- "&" CGI .

- ASCII (, ISO-8859-1 127 ) URL .

- "%" HTTP . , "%" "%68%65%6C%6C%6F" "hello" .

<SCRIPT> </SCRIPT> :

- , , .

:

- (!) (") .

:

- UTF-7 "<" '+ADw-' . ( , UTF-7) .

XSS . , . . .

, . ISO 8859-1 HTML [2].

Cross-Site Scripting HTTP Cross-Site Scripting . . . .

**Tips:**

1. Fortify Secure Coding Rulepacks SQL Injection , XSS . , DATABASE . .

2. URL XSS , JavaScript DOM(Document Object Model) . Rulepacks Cross-Site Scripting URL . URL Fortify Cross-Site Scripting: Poor Validation .

**login.jsp, line 19 (Cross-Site Scripting: Persistent)**

| | | | |
|---|---|---|---|
| **Fortify Priority:** | Critical | **Folder** | Critical |

|  |  |
|---|---|
| **Kingdom:** | Input Validation and Representation |
| **Abstract:** | login.jsp _jspService()  19 . |
| **Source:** | LoginModel.java:9 java.sql.Statement.executeQuery() |

```
7    public String check(Connection conn, String user, String pass) throws SQLException{
8     state = conn.createStatement();
9     rs = state.executeQuery(sql);
10    while(rs.next()){
11     if(rs.getString(2).equals(user)){
```

**Sink:** login.jsp:19 javax.servlet.jsp.JspWriter.print()

```
17
18   %>
19   <p>Hello, <%= ((String)request.getAttribute("res")).substring(1) %></p>
20   <% }else{
21    %>
```

## Issue Count by Category

| Issues by 49 | |
|---|---|
| <none> | 16 |
| 04.01. | 7 |
| 02.09. | 1 |
| 01.11. | 3 |
| 01.04. | 1 |

## Issue Breakdown by Analysis

### Issues by Analysis